

z/OS  
3.2

*MVS Programming:  
Sysplex Services Reference*



**Note**

Before using this information and the product it supports, read the information in [“Notices” on page 1645](#).

This edition applies to IBM® z/OS® 3.2 (5655-ZOS) and to all subsequent releases and modifications until otherwise indicated in new editions.

Last updated: 2025-09-30

© **Copyright International Business Machines Corporation 1994, 2025.**

US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

---

# Contents

<b>Figures.....</b>	<b>xxi</b>
<b>Tables.....</b>	<b>xxiii</b>
<b>About this information.....</b>	<b>xxvii</b>
Who should use this information.....	xxvii
Where to find more information.....	xxvii
<b>How to provide feedback to IBM.....</b>	<b>xxix</b>
<b>Summary of changes.....</b>	<b>xxxix</b>
Summary of changes for z/OS 3.2.....	xxxix
Summary of changes for z/OS 3.1.....	xxxix
<b>Chapter 1. How to read syntax diagrams.....</b>	<b>1</b>
Symbols.....	1
Syntax items.....	1
Syntax examples.....	2
<b>Chapter 2. Specifying a Macro Version Number.....</b>	<b>5</b>
How to Request a Macro Version.....	5
General Considerations When Using PLISTVER.....	5
<b>Chapter 3. IHABLDP — Build Dump Parameter List Service.....</b>	<b>7</b>
Description.....	7
Environment.....	8
Programming Requirements.....	8
Restrictions.....	8
Input Register Information.....	9
Output Register Information.....	9
Performance Implications.....	9
Understanding IHABLDP Version Support.....	9
Syntax.....	10
Parameter Descriptions.....	10
ABEND Codes.....	14
Return and Reason Codes.....	14
<b>Chapter 4. IXCARM — Request Automatic Restart Management Services.....</b>	<b>15</b>
Description.....	15
Environment.....	15
Programming Requirements.....	15
Restrictions.....	16
Input Register Information.....	16
Output Register Information.....	16
Performance Implications.....	17
Understanding IXCARM Version Support.....	17
Syntax.....	17
Parameter Descriptions.....	19

ABEND Codes.....	25
Return and Reason Codes.....	25
Example.....	49
<b>Chapter 5. IXCCFCM — Coupling Facility Configuration Management.....</b>	<b>51</b>
Description.....	51
Recovery Manager Concepts.....	51
Environment.....	51
Restrictions.....	52
Input Register Information.....	52
Output Register Information.....	52
Programming Requirements.....	52
Performance Implications.....	53
Syntax Diagram.....	53
Parameter Descriptions.....	53
Return and Reason Codes.....	54
<b>Chapter 6. IXCCREAT — Define a Member to XCF.....</b>	<b>59</b>
Description.....	59
Environment.....	59
Programming Requirements.....	59
Restrictions.....	59
Input Register Information.....	60
Output Register Information.....	60
Performance Implications.....	60
Syntax Diagram.....	60
Parameter Descriptions.....	61
ABEND Codes.....	63
Return and Reason Codes.....	63
Example.....	69
<b>Chapter 7. IXCDELET — Change an XCF Member's State to Not-Defined.....</b>	<b>71</b>
Description.....	71
Environment.....	71
Programming Requirements.....	71
Restrictions.....	71
Input Register Information.....	71
Output Register Information.....	71
Performance Implications.....	72
Syntax Diagram.....	72
Parameter Descriptions.....	72
ABEND Codes.....	74
Return and Reason Codes.....	74
Example.....	76
<b>Chapter 8. IXCJOIN — Place an XCF Member in the Active State.....</b>	<b>79</b>
Description.....	79
Environment.....	79
Programming Requirements.....	80
Restrictions.....	80
Input Register Information.....	80
Output Register Information.....	80
Performance Implications.....	81
Understanding IXCJOIN Version Support.....	81
Syntax Diagram.....	81
Parameter Descriptions.....	82
ABEND Codes.....	90

Return and Reason Codes.....	90
Example.....	98
<b>Chapter 9. IXCLEAVE — Place an XCF Member in the Not-Defined State.....</b>	<b>101</b>
Description.....	101
Environment.....	101
Programming Requirements.....	101
Restrictions.....	101
Input Register Information.....	101
Output Register Information.....	102
Performance Implications.....	102
Syntax Diagram.....	102
Parameter Descriptions.....	103
ABEND Codes.....	104
Return and Reason Codes.....	104
Example.....	108
<b>Chapter 10. IXCMG — Obtain Tuning and Capacity Planning Data.....</b>	<b>109</b>
Description.....	109
Environment.....	109
Programming Requirements.....	110
Restrictions.....	110
Input Register Information.....	110
Output Register Information.....	110
Performance Implications.....	111
Understanding IXCMG version support.....	111
Syntax Diagram.....	111
Parameter Descriptions.....	112
ABEND Codes.....	116
Return and Reason Codes.....	116
Example.....	123
<b>Chapter 11. IXCMOD — Modify Status-Checking Interval.....</b>	<b>125</b>
Description.....	125
Environment.....	125
Programming Requirements.....	125
Restrictions.....	125
Input Register Information.....	125
Output Register Information.....	126
Performance Implications.....	126
Syntax Diagram.....	126
Parameter Descriptions.....	127
ABEND Codes.....	128
Return and Reason Codes.....	128
Example.....	131
<b>Chapter 12. IXCMMSGC — XCF Message Control.....</b>	<b>133</b>
Description.....	133
Environment.....	133
Programming Requirements.....	133
Restrictions.....	134
Input Register Information.....	134
Output Register Information.....	134
Understanding IXCMMSGC Version Support.....	134
Syntax Diagram.....	135
Parameter Descriptions.....	137
ABEND Codes.....	147

Return and Reason Codes.....	147
<b>Chapter 13. IXCMSGI — Receive a Message from Another Member in the XCF Group.....</b>	<b>155</b>
<b>Chapter 14. IXCMSGIX — Receive a Message from Another Member in the XCF Group.....</b>	<b>157</b>
Description.....	157
Environment.....	157
Programming Requirements.....	158
Restrictions.....	158
Input Register Information.....	159
Output Register Information.....	159
Performance Implications.....	159
Understanding IXCMSGIX Version Support.....	159
Syntax Diagram.....	160
Parameter Descriptions.....	162
ABEND Codes.....	167
Return and Reason Codes.....	167
Example.....	179
<b>Chapter 15. IXCMSGO — Send a Message to Another Member in the XCF Group... 181</b>	
<b>Chapter 16. IXCMSGOX — Send a Message to Another Member in the XCF Group. 183</b>	
Description.....	183
Environment.....	184
Programming Requirements.....	185
Restrictions.....	186
Input Register Information.....	186
Output Register Information.....	186
Performance Implications.....	187
Understanding IXCMSGOX Version Support.....	187
Syntax Diagram.....	188
Parameter Descriptions.....	191
ABEND Codes.....	204
Return and reason codes.....	204
Example.....	227
<b>Chapter 17. IXCNODE — XCF Note Pad Interface..... 229</b>	
Description.....	229
Environment.....	230
Programming Requirements.....	232
Restrictions.....	232
Input Register Information.....	233
Output Register Information.....	234
Performance Implications.....	234
Understanding IXCNODE Version Support.....	234
Syntax.....	234
Parameters.....	238
Parameters Common to All Requests.....	238
Parameters for REQUEST=NOTEPAD.....	241
Parameters for REQUEST=CONNECTION.....	246
Parameters for REQUEST=NOTE.....	252
Parameters for REQUEST=NOTES.....	256
ABEND Codes.....	259
Return and Reason Codes.....	259

Examples.....	277
<b>Chapter 18. IXCQUERY – Obtain XCF Information.....</b>	<b>279</b>
Description.....	279
Environment.....	280
Programming Requirements.....	281
Restrictions.....	281
Input Register Information.....	281
Output Register Information.....	281
Performance Implications.....	282
Understanding IXCQUERY Version Support.....	282
Syntax Diagram.....	282
Parameter descriptions.....	285
Return and reason codes.....	293
Example.....	301
<b>Chapter 19. IXCQUIES – Place an XCF Member in a Quiesced State.....</b>	<b>303</b>
Description.....	303
Environment.....	303
Programming Requirements.....	303
Restrictions.....	303
Input Register Information.....	303
Output Register Information.....	304
Performance Implications.....	304
Syntax Diagram.....	304
Parameter Descriptions.....	305
ABEND Codes.....	306
Return and Reason Codes.....	306
Example.....	311
<b>Chapter 20. IXCRECV– Receive Client/Server Information.....</b>	<b>313</b>
Description.....	313
Environment.....	313
Programming Requirements.....	313
Restrictions.....	314
Input Register Information.....	314
Output Register Information.....	314
Performance Implications.....	314
Understanding IXCRECV Version Support.....	314
Syntax.....	314
Parameters.....	315
ABEND Codes.....	320
Return and Reason Codes.....	320
Example.....	325
<b>Chapter 21. IXCREQ – Format a Request for the XCF Server.....</b>	<b>327</b>
Description.....	327
Environment.....	327
Programming Requirements.....	327
Restrictions.....	328
Input Register Information.....	328
Output Register Information.....	328
Performance Implications.....	328
Syntax.....	328
Parameters.....	329
ABEND Codes.....	331
Return and Reason Codes.....	331

Example.....	335
<b>Chapter 22. IXCSEND — Send Client/Server Requests and Responses.....</b>	<b>337</b>
Description.....	337
Environment.....	337
Programming Requirements.....	338
Restrictions.....	338
Input Register Information.....	338
Output Register Information.....	338
Performance Implications.....	339
Understanding IXCSEND Version Support.....	339
Syntax.....	339
Parameters.....	342
ABEND Codes.....	353
Return and Reason Codes.....	353
Examples.....	371
<b>Chapter 23. IXCSETUS — Update the User State Field.....</b>	<b>373</b>
Description.....	373
Environment.....	373
Programming Requirements.....	373
Restrictions.....	374
Input Register Information.....	374
Output Register Information.....	374
Performance Implications.....	374
Understanding IXCSETUS Version Support.....	374
Syntax Diagram.....	375
Parameter Descriptions.....	375
ABEND Codes.....	378
Return and Reason Codes.....	378
Example.....	383
<b>Chapter 24. IXCSRVR — Define a Server to XCF.....</b>	<b>385</b>
Description.....	385
Environment.....	385
Programming Requirements.....	386
Restrictions.....	386
Input Register Information.....	386
Output Register Information.....	387
Performance Implications.....	387
Syntax.....	387
Parameters.....	388
ABEND Codes.....	394
Return and Reason Codes.....	394
Example.....	403
<b>Chapter 25. IXCSYSCL — Notify the System that Cleanup Has Completed.....</b>	<b>405</b>
Description.....	405
Environment.....	405
Programming Requirements.....	405
Restrictions.....	405
Input Register Information.....	405
Output Register Information.....	406
Performance Implications.....	406
Understanding IXCSYSCL Version Support.....	406
Syntax Diagram.....	406
Parameter Descriptions.....	407



ABEND Codes.....	408
Return and Reason Codes.....	408
Example.....	412
<b>Chapter 26. IXCTERM — Terminate a Member of an XCF Group.....</b>	<b>421</b>
Description.....	421
Environment.....	421
Programming Requirements.....	422
Restrictions.....	422
Input Register Information.....	422
Output Register Information.....	422
Performance Implications.....	423
Syntax Diagram.....	423
Parameter Descriptions.....	423
ABEND Codes.....	424
Return and Reason Codes.....	424
Example.....	428
<b>Chapter 27. IXLADUPX — Synchronize an asynchronously duplexed structure....</b>	<b>429</b>
Description.....	429
Environment.....	429
Programming requirements.....	429
Restrictions.....	430
Input register information.....	430
Output register information.....	430
Performance implications.....	430
Understanding IXLADUPX version support.....	431
Syntax diagram.....	431
Parameter descriptions.....	431
Return and reason codes .....	433
<b>Chapter 28. IXLALTER — Alter a Coupling Facility Structure.....</b>	<b>437</b>
Description.....	437
Environment.....	438
Input Register Information.....	439
Output Register Information.....	439
Programming Requirements.....	439
Performance Implications.....	439
Understanding IXLALTER Version Support.....	439
Syntax Diagram.....	440
Parameter Descriptions.....	440
Return and reason codes.....	443
<b>Chapter 29. IXLAXISN.....</b>	<b>449</b>
Environment.....	449
Programming Requirements.....	449
Performance Implications.....	449
Understanding IXLAXISN Version Support.....	449
Syntax diagram.....	449
Input Register Information.....	450
Output Register Information.....	450
Parameter Descriptions.....	451
Return and reason codes.....	452
<b>Chapter 30. IXLCACHE — Cache Services.....</b>	<b>457</b>
Description.....	457
Environment.....	458

Programming Requirements.....	459
Restrictions.....	460
Input Register Information.....	460
Output Register Information.....	460
Performance Implications.....	461
Understanding IXLCACHE Version Support.....	461
<b>Chapter 31. IXLCACHE REQUEST=CASTOUT_DATA.....</b>	<b>463</b>
Description.....	463
Syntax Diagram.....	463
Parameter Descriptions.....	465
ABEND Codes.....	473
Return and Reason Codes.....	473
<b>Chapter 32. IXLCACHE REQUEST=CASTOUT_DATALIST.....</b>	<b>485</b>
Description.....	485
Syntax Diagram.....	486
Parameter Descriptions.....	488
ABEND Codes.....	495
Return and Reason Codes.....	495
<b>Chapter 33. IXLCACHE REQUEST=CROSS_INVAL.....</b>	<b>507</b>
Description.....	507
Syntax Diagram.....	507
Parameter Descriptions.....	508
ABEND Codes.....	514
Return and Reason Codes.....	514
<b>Chapter 34. IXLCACHE REQUEST=CROSS_INVALLIST.....</b>	<b>523</b>
Description.....	523
Syntax Diagram.....	523
Parameter Descriptions.....	525
ABEND Codes.....	531
Return and Reason Codes.....	531
<b>Chapter 35. IXLCACHE REQUEST=DELETE_NAME.....</b>	<b>543</b>
Description.....	543
Syntax Diagram.....	543
Parameter Descriptions.....	545
ABEND Codes.....	552
Return and Reason Codes.....	552
<b>Chapter 36. IXLCACHE REQUEST=DELETE_NAMELIST.....</b>	<b>561</b>
Description.....	561
Syntax Diagram.....	562
Parameter Descriptions.....	564
ABEND Codes.....	573
Return and Reason Codes.....	573
<b>Chapter 37. IXLCACHE REQUEST=PROCESS_REFLIST.....</b>	<b>587</b>
Description.....	587
Syntax Diagram.....	587
Parameter Descriptions.....	589
ABEND Codes.....	595
Return and Reason Codes.....	595

<b>Chapter 38. IXLCACHE REQUEST=READ_COCLASS.....</b>	<b>605</b>
Description.....	605
Syntax Diagram.....	605
Parameter Descriptions.....	607
ABEND Codes.....	615
Return and Reason Codes.....	615
<b>Chapter 39. IXLCACHE REQUEST=READ_COSTATS.....</b>	<b>629</b>
Description.....	629
Syntax Diagram.....	629
Parameter Descriptions.....	631
ABEND Codes.....	637
Return and Reason Codes.....	637
<b>Chapter 40. IXLCACHE REQUEST=READ_DATA.....</b>	<b>649</b>
Description.....	649
Syntax Diagram.....	649
Parameter Descriptions.....	651
ABEND Codes.....	660
Return and Reason Codes.....	661
<b>Chapter 41. IXLCACHE REQUEST=READ_DIRINFO.....</b>	<b>675</b>
Description.....	675
Syntax Diagram.....	675
Parameter Descriptions.....	677
ABEND Codes.....	685
Return and Reason Codes.....	685
<b>Chapter 42. IXLCACHE REQUEST=READ_STGSTATS.....</b>	<b>697</b>
Description.....	697
Syntax Diagram.....	697
Parameter Descriptions.....	698
ABEND Codes.....	701
Return and Reason Codes.....	701
<b>Chapter 43. IXLCACHE REQUEST=REG_NAMELIST.....</b>	<b>709</b>
Description.....	709
Syntax Diagram.....	709
Parameter Descriptions.....	711
ABEND Codes.....	716
Return and Reason Codes.....	716
<b>Chapter 44. IXLCACHE REQUEST=RESET_REFBIT.....</b>	<b>727</b>
Description.....	727
Syntax Diagram.....	727
Parameter Descriptions.....	729
ABEND Codes.....	733
Return and Reason Codes.....	733
<b>Chapter 45. IXLCACHE REQUEST=SET_RECLVCTR.....</b>	<b>743</b>
Description.....	743
Syntax Diagram.....	743
Parameter Descriptions.....	744
ABEND Codes.....	748
Return and Reason Codes.....	748

<b>Chapter 46. IXLCACHE REQUEST=UNLOCK_CASTOUT.....</b>	<b>757</b>
Description.....	757
Syntax Diagram.....	757
Parameter Descriptions.....	759
ABEND Codes.....	765
Return and Reason Codes.....	765
<b>Chapter 47. IXLCACHE REQUEST=UNLOCK_CO_NAME.....</b>	<b>777</b>
Description.....	777
Syntax Diagram.....	777
Parameter Descriptions.....	778
ABEND Codes.....	782
Return and Reason Codes.....	782
<b>Chapter 48. IXLCACHE REQUEST=WRITE_DATA.....</b>	<b>791</b>
Description.....	791
Syntax Diagram.....	792
Parameter Descriptions.....	795
ABEND Codes.....	808
Return and Reason Codes.....	808
<b>Chapter 49. IXLCACHE REQUEST=WRITE_DATALIST.....</b>	<b>821</b>
Description.....	821
Syntax Diagram.....	822
Parameter Descriptions.....	824
ABEND Codes.....	832
Return and Reason Codes.....	832
<b>Chapter 50. IXLCONN — Connect to a coupling facility structure.....</b>	<b>847</b>
Description.....	847
Environment.....	848
Restrictions.....	849
Input register information.....	849
Output register information.....	849
Programming requirements.....	850
Performance implications.....	850
Understanding IXLCONN version support.....	850
Syntax diagram.....	851
Parameter descriptions.....	855
Parameters common to all structure types.....	855
Parameters for TYPE=CACHE.....	871
Parameters for TYPE=LIST.....	874
Parameters for TYPE=LOCK.....	879
Return and reason codes for the IXLCONN macro.....	883
<b>Chapter 51. IXLCSP — XES Structure Computation Service.....</b>	<b>901</b>
Description.....	901
Environment.....	901
Programming Requirements.....	901
Input Register Information.....	902
Output Register Information.....	902
Performance Implications.....	902
Understanding IXLCSP version support.....	902
Syntax diagram.....	903
Parameter descriptions.....	906

ABEND Codes.....	915
Return and Reason Codes.....	915
<b>Chapter 52. IXLDISC – Disconnecting from a Coupling Facility Structure.....</b>	<b>923</b>
Description.....	923
Environment.....	923
Restrictions.....	923
Input Register Information.....	924
Output Register Information.....	924
Programming Requirements.....	924
Performance Implications.....	924
Understanding IXLDISC Version Support.....	924
Syntax Diagram.....	925
Parameter Descriptions.....	925
Return and Reason Codes.....	927
<b>Chapter 53. IXLEERSP – Responding to an Event.....</b>	<b>931</b>
Description.....	931
Environment.....	931
Restrictions.....	932
Input Register Information.....	932
Output Register Information.....	932
Programming Requirements.....	932
Performance Implications.....	932
Understanding IXLEERSP Version Support.....	932
Syntax Diagram.....	933
Parameter Descriptions.....	933
Return and Reason Codes.....	937
<b>Chapter 54. IXLFCOMP – Wait for Completion or Obtain Status of an IXLLIST or IXLCACHE Request.....</b>	<b>943</b>
Description.....	943
Environment.....	943
Programming Requirements.....	944
Restrictions.....	944
Input Register Information.....	944
Output Register Information.....	944
Performance Implications.....	945
Syntax Diagram.....	945
Parameter Descriptions.....	945
ABEND Codes.....	947
Return and Reason Codes.....	947
<b>Chapter 55. IXLFORCE – Deleting a Persistent Structure or Failed Connection....</b>	<b>951</b>
Description.....	951
Environment.....	952
Restrictions.....	952
Input Register Information.....	952
Output Register Information.....	952
Programming Requirements.....	953
Performance Implications.....	953
Syntax Diagram.....	953
Parameter Descriptions.....	954
Return and Reason Codes.....	956
<b>Chapter 56. IXLLIST – List Services.....</b>	<b>961</b>

Description.....	961
Environment.....	962
Programming Requirements.....	963
Restrictions.....	964
Input Register Information.....	964
Output Register Information.....	964
Performance Implications.....	964
Understanding IXLLIST Version Support.....	965
Coding Equivalent Functions.....	965
<b>Chapter 57. IXLLIST REQUEST=DELETE.....</b>	<b>967</b>
Description.....	967
Syntax Diagram.....	967
Parameter Descriptions.....	970
ABEND Codes.....	982
Return and Reason Codes.....	982
<b>Chapter 58. IXLLIST REQUEST=DELETE_ENTRYLIST.....</b>	<b>995</b>
Description.....	995
Syntax Diagram.....	995
Parameter Descriptions.....	997
ABEND Codes.....	1006
Return and Reason Codes.....	1006
<b>Chapter 59. IXLLIST REQUEST=DELETE_MULT.....</b>	<b>1017</b>
Description.....	1017
Syntax Diagram.....	1017
Parameter Descriptions.....	1019
ABEND Codes.....	1025
Return and Reason Codes.....	1025
<b>Chapter 60. IXLLIST REQUEST=DEQ_EVENTQ.....</b>	<b>1035</b>
Description.....	1035
Syntax Diagram.....	1035
Parameter Descriptions.....	1036
ABEND Codes.....	1041
Return and Reason Codes.....	1041
<b>Chapter 61. IXLLIST REQUEST=LOCK.....</b>	<b>1053</b>
Description.....	1053
Syntax Diagram.....	1053
Parameter Descriptions.....	1055
ABEND Codes.....	1060
Return and Reason Codes.....	1060
<b>Chapter 62. IXLLIST REQUEST=MONITOR_EVENTQ.....</b>	<b>1067</b>
Description.....	1067
Syntax Diagram.....	1067
Parameter Descriptions.....	1068
ABEND Codes.....	1072
Return and Reason Codes.....	1073
<b>Chapter 63. IXLLIST REQUEST=MONITOR_LIST.....</b>	<b>1081</b>
Description.....	1081
Syntax Diagram.....	1081
Parameter Descriptions.....	1082

ABEND Codes.....	1086
Return and Reason Codes.....	1087
<b>Chapter 64. IXLLIST REQUEST=MONITOR_SUBLIST.....</b>	<b>1095</b>
Description.....	1095
Syntax Diagram.....	1095
Parameter Descriptions.....	1096
ABEND Codes.....	1100
Return and Reason Codes.....	1100
<b>Chapter 65. IXLLIST REQUEST=MONITOR_SUBLISTS.....</b>	<b>1109</b>
Description.....	1109
Syntax Diagram.....	1109
Parameter Descriptions.....	1111
ABEND Codes.....	1117
Return and Reason Codes.....	1117
<b>Chapter 66. IXLLIST REQUEST=MOVE.....</b>	<b>1129</b>
Description.....	1129
Syntax Diagram.....	1130
Parameter Descriptions.....	1138
ABEND Codes.....	1153
Return and Reason Codes.....	1153
<b>Chapter 67. IXLLIST REQUEST=READ.....</b>	<b>1169</b>
Description.....	1169
Syntax Diagram.....	1169
Parameter Descriptions.....	1172
ABEND Codes.....	1183
Return and Reason Codes.....	1183
<b>Chapter 68. IXLLIST REQUEST=READ_EMCONTROLS.....</b>	<b>1197</b>
Description.....	1197
Syntax Diagram.....	1197
Parameter Descriptions.....	1198
ABEND Codes.....	1202
Return and Reason Codes.....	1202
<b>Chapter 69. IXLLIST REQUEST=READ_EQCONTROLS.....</b>	<b>1209</b>
Description.....	1209
Syntax Diagram.....	1209
Parameter Descriptions.....	1210
ABEND Codes.....	1213
Return and Reason Codes.....	1213
<b>Chapter 70. IXLLIST REQUEST=READ_LCONTROLS.....</b>	<b>1221</b>
Description.....	1221
Syntax Diagram.....	1221
Parameter Descriptions.....	1222
ABEND Codes.....	1227
Return and Reason Codes.....	1227
<b>Chapter 71. IXLLIST REQUEST=READ_LIST.....</b>	<b>1237</b>
Description.....	1237
Syntax Diagram.....	1237
Parameter Descriptions.....	1240

ABEND Codes.....	1250
Return and Reason Codes.....	1250
<b>Chapter 72. IXLLIST REQUEST=READ_MULT.....</b>	<b>1265</b>
Description.....	1265
Syntax Diagram.....	1265
Parameter Descriptions.....	1267
ABEND Codes.....	1276
Return and Reason Codes.....	1276
<b>Chapter 73. IXLLIST REQUEST=WRITE.....</b>	<b>1291</b>
Description.....	1291
Syntax Diagram.....	1291
Parameter Descriptions.....	1295
ABEND Codes.....	1308
Return and Reason Codes.....	1309
<b>Chapter 74. IXLLIST REQUEST=WRITE_LCONTROLS.....</b>	<b>1323</b>
Description.....	1323
Syntax Diagram.....	1323
Parameter Descriptions.....	1325
ABEND Codes.....	1330
Return and Reason Codes.....	1330
<b>Chapter 75. IXLLOCK Services.....</b>	<b>1339</b>
Description.....	1339
Environment.....	1339
Programming Requirements.....	1340
Restrictions and Limitations.....	1340
Input Register Information.....	1340
Output Register Information.....	1340
Performance Implications.....	1341
Understanding IXLLOCK version support.....	1341
Syntax diagram.....	1342
Parameter Descriptions.....	1344
REQUEST=OBTAIN.....	1346
REQUEST=ALTER.....	1352
REQUEST=RELEASE.....	1358
REQUEST=PROCESSMULT.....	1361
ABEND Codes.....	1362
Return and reason codes.....	1362
<b>Chapter 76. IXLLSTC – XES List Structure Control Services.....</b>	<b>1371</b>
Description.....	1371
Environment.....	1372
Programming Requirements.....	1372
Restrictions.....	1373
Input Register Information.....	1373
Output Register Information.....	1373
Performance Implications.....	1374
Understanding IXLLSTC Version Support.....	1374
Summary of Version-Dependent Parameter Functions.....	1375
Syntax Diagram.....	1375
Parameter Descriptions.....	1379
ABEND Codes.....	1402
Return and Reason Codes.....	1402



<b>Chapter 77. IXLLSTE – XES List Structure Single Entry Services .....</b>	<b>1417</b>
Description.....	1417
Environment.....	1417
Programming Requirements.....	1418
Restrictions.....	1418
Input Register Information.....	1418
Output Register Information.....	1418
Performance Implications.....	1419
Understanding IXLLSTE Version Support.....	1419
Summary of Version-Dependent Parameter Functions.....	1420
Syntax Diagram.....	1420
Parameter Descriptions.....	1430
ABEND Codes.....	1460
Return and Reason Codes.....	1460
 <b>Chapter 78. IXLLSTM – XES List Structure Multiple Entry Services.....</b>	 <b>1475</b>
Description.....	1475
Environment.....	1475
Programming Requirements.....	1476
Restrictions.....	1476
Input Register Information.....	1477
Output Register Information.....	1477
Performance Implications.....	1477
Understanding IXLLSTM Version Support.....	1477
Summary of Version-Dependent Parameter Functions.....	1478
Syntax Diagram.....	1479
Parameter Descriptions.....	1486
ABEND Codes.....	1502
Return and Reason Codes.....	1503
 <b>Chapter 79. IXLMG – Coupling facility measurement.....</b>	 <b>1521</b>
Description.....	1521
Environment.....	1521
Programming Requirements.....	1522
Restrictions.....	1522
Input Register Information.....	1522
Output Register Information.....	1522
Performance Implications.....	1523
Understanding IXLMG Version Support.....	1523
Syntax diagram.....	1523
Parameter descriptions.....	1524
ABEND Codes.....	1529
Return and Reason Codes.....	1529
 <b>Chapter 80. IXLPURGE – Purge Operations to a Coupling Facility.....</b>	 <b>1533</b>
Description.....	1533
Environment.....	1533
Programming Requirements.....	1533
Restrictions.....	1533
Input Register Information.....	1533
Output Register Information.....	1533
Performance Implications.....	1534
Syntax Diagram.....	1534
Parameter Descriptions.....	1534
ABEND Codes.....	1536
Return and Reason Codes.....	1536

<b>Chapter 81. IXLREBLD — Rebuilding or duplexing a structure.....</b>	<b>1539</b>
Description.....	1539
Environment.....	1540
Programming Requirements.....	1541
Restrictions.....	1541
Input Register Information.....	1541
Output Register Information.....	1541
Performance Implications.....	1541
Understanding IXLREBLD version support.....	1542
Syntax diagram.....	1542
Parameter descriptions.....	1544
Return and reason codes.....	1551
 <b>Chapter 82. IXLRT — Lock structure record data processing.....</b>	 <b>1565</b>
Description.....	1565
Environment.....	1565
Programming Requirements.....	1565
Restrictions and Limitations.....	1566
Input Register Information.....	1566
Output Register Information.....	1566
Performance Implications.....	1566
Understanding IXLRT Version Support.....	1566
Syntax Diagram.....	1567
Parameter descriptions.....	1569
ABEND Codes.....	1576
Return and Reason Codes.....	1576
 <b>Chapter 83. IXLSYNCH — Synchronous update to a lock structure.....</b>	 <b>1583</b>
Description.....	1583
Environment.....	1583
Programming Requirements.....	1583
Restrictions and Limitations.....	1583
Input Register Information.....	1583
Output Register Information.....	1583
Performance Implications.....	1584
Understanding IXLSYNCH Version Support.....	1584
Syntax Diagram.....	1584
Parameter descriptions.....	1585
ABEND Codes.....	1587
Return and Reason Codes.....	1587
 <b>Chapter 84. IXLUSYNC — Synchronizing Processing for User-Defined Events.....</b>	 <b>1593</b>
Description.....	1593
Environment.....	1593
Restrictions.....	1593
Input Register Information.....	1594
Output Register Information.....	1594
Programming Requirements.....	1594
Performance Implications.....	1594
Understanding IXLUSYNC Version Support.....	1594
Syntax Diagram.....	1595
Parameter Descriptions.....	1595
Return and Reason Codes.....	1599

<b>Chapter 85. IXLVECTR — Check or Modify Local Cache Vector or List Notification</b>	
<b>Vector.....</b>	<b>1605</b>
Description.....	1605
Environment.....	1605
Programming Requirements.....	1606
Restrictions.....	1606
Input Register Information.....	1606
Output Register Information.....	1606
Performance Implications.....	1607
Syntax Diagram.....	1607
Parameter Descriptions.....	1608
ABEND Codes.....	1611
Return Codes.....	1611
<b>Chapter 86. IXLZSTR — Coupling Facility Structure Data Access Service.....</b>	<b>1617</b>
Description.....	1617
Environment.....	1618
Programming Requirements.....	1619
Restrictions.....	1620
Input Register Information.....	1620
Output Register Information.....	1620
Performance Implications.....	1620
Syntax Diagram.....	1621
Parameter Descriptions.....	1623
ABEND Codes.....	1629
Return and Reason Codes.....	1629
Example.....	1631
<b>Appendix A. Accessibility.....</b>	<b>1643</b>
<b>Notices.....</b>	<b>1645</b>
Terms and conditions for product documentation.....	1646
IBM Online Privacy Statement.....	1647
Policy for unsupported hardware.....	1647
Minimum supported hardware.....	1647
Programming Interface Information.....	1648
Trademarks.....	1648
<b>Index.....</b>	<b>1649</b>



---

# Figures



---

# Tables

1. Syntax examples.....	2
2. Return and Reason Codes for the IHABLDP Macro.....	14
3. Return and Reason Codes for the IXCARM Macro.....	26
4. Return and Reason Codes for the IXCCFCM macro.....	55
5. Return and Reason Codes for the IXCCREAT Macro.....	64
6. Return and Reason Codes for the IXCDELET Macro.....	74
7. Return and Reason Codes for the IXCJOIN Macro.....	91
8. Return and Reason Codes for the IXCLEAVE Macro.....	105
9. Return and Reason Codes for the IXCMG Macro.....	116
10. Return and Reason Codes for the IXCMOD Macro.....	129
11. Return and Reason Codes for the IXCMSGC Macro.....	148
12. Return and Reason Codes for the IXCMSGIX Macro.....	168
13. Return and reason codes for the IXCMSGOX macro.....	204
14. Return and Reason Codes for the IXCNOTE Macro.....	260
15. Return and Reason Codes for the IXCQUERY Macro.....	293
16. Return and Reason Codes for the IXCQUIES Macro.....	307
17. Return and Reason Codes for the IXCRECV Macro.....	320
18. Return and Reason Codes for the IXCREQ Macro.....	331
19. Return and Reason Codes for the IXCSEND Macro.....	354
20. Return and Reason Codes for the IXCSETUS Macro.....	378
21. Return and Reason Codes for the IXCSRVR Macro.....	395
22. Return and Reason Codes for the IXCSYSCL Macro.....	409
23. Return and Reason Codes for the IXCTERM Macro.....	425

24. Return and reason codes for the IXLADUPX macro.....	433
25. Return and reason codes for the IXLALTER macro.....	444
26. Return and reason codes for the IXLALTER macro.....	453
27. Request Types for IXLCACHE.....	457
28. Mapping Macros for IXLCACHE.....	459
29. Return and Reason Codes for the IXLCACHE REQUEST=CASTOUT_DATA Macro.....	473
30. Return and Reason Codes for the IXLCACHE REQUEST=CASTOUT_DATA LIST Macro.....	496
31. Return and Reason Codes for the IXLCACHE REQUEST=CROSS_INVALID Macro.....	515
32. Return and Reason Codes for the IXLCACHE REQUEST=CROSS_INVALID LIST Macro.....	532
33. Return and Reason Codes for the IXLCACHE REQUEST=DELETE_NAME Macro.....	553
34. Return and Reason Codes for the IXLCACHE REQUEST=DELETE_NAME LIST Macro.....	574
35. Return and Reason Codes for the IXLCACHE REQUEST=PROCESS_REFLIST Macro.....	595
36. Return and Reason Codes for the IXLCACHE REQUEST=READ_COCLASS Macro.....	616
37. Return and Reason Codes for the IXLCACHE REQUEST=READ_COSTATS Macro.....	638
38. Return and Reason Codes for the IXLCACHE REQUEST=READ_DATA Macro.....	661
39. Return and Reason Codes for the IXLCACHE REQUEST=READ_DIRINFO Macro.....	686
40. Return and Reason Codes for the IXLCACHE REQUEST=READ_STGSTATS Macro.....	702
41. Return and Reason Codes for the IXLCACHE REQUEST=REG_NAME LIST Macro.....	717
42. Return and Reason Codes for the IXLCACHE REQUEST=RESET_REFBIT Macro.....	734
43. Return and Reason Codes for the IXLCACHE REQUEST=SET_RECLVCTR Macro.....	749
44. Return and Reason Codes for the IXLCACHE REQUEST=UNLOCK_CASTOUT Macro.....	766
45. Return and Reason Codes for the IXLCACHE REQUEST=UNLOCK_CO_NAME Macro.....	783
46. Return and Reason Codes for the IXLCACHE REQUEST=WRITE_DATA Macro.....	808
47. Return and Reason Codes for the IXLCACHE REQUEST=WRITE_DATA LIST Macro.....	833
48. Return and reason codes for the IXLCONN macro.....	884



49. Return and Reason Codes for the IXLCSP Macro.....	915
50. Return and Reason Codes for the IXLDISC Macro.....	928
51. Return and Reason Codes for the IXLEERSP Macro.....	937
52. Return and Reason Codes for the IXLFCOMP Macro.....	947
53. Return and Reason Codes for the IXLFORCE Macro.....	957
54. Request Types for IXLLIST.....	961
55. Mapping Macros for IXLLIST.....	963
56. IXLLIST Version Support.....	965
57. Return and Reason Codes for IXLLIST REQUEST=DELETE Macro.....	982
58. Return and Reason Codes for IXLLIST REQUEST=DELETE_ENTRYLIST Macro.....	1006
59. Return and Reason Codes for IXLLIST REQUEST=DELETE_MULT Macro.....	1026
60. Return and Reason Codes for IXLLIST REQUEST=DEQ_EVENTQ Macro.....	1042
61. Return and Reason Codes for IXLLIST REQUEST=LOCK Macro.....	1060
62. Return and Reason Codes for IXLLIST REQUEST=MONITOR_EVENTQ Macro.....	1073
63. Return and Reason Codes for IXLLIST REQUEST=MONITOR_LIST Macro.....	1087
64. Return and Reason Codes for IXLLIST REQUEST=MONITOR_SUBLIST Macro.....	1101
65. Return and Reason Codes for IXLLIST REQUEST=MONITOR_SUBLISTS Macro.....	1118
66. Return and Reason Codes for IXLLIST REQUEST=MOVE Macro.....	1153
67. Return and Reason Codes for IXLLIST REQUEST=READ Macro.....	1184
68. Return and Reason Codes for IXLLIST REQUEST=READ_EMCONTROLS Macro.....	1202
69. Return and Reason Codes for IXLLIST REQUEST=READ_EQCONTROLS Macro.....	1214
70. Return and Reason Codes for IXLLIST REQUEST=READ_LCONTROLS Macro.....	1228
71. Return and Reason Codes for IXLLIST REQUEST=READ_LIST Macro.....	1251
72. Return and Reason Codes for IXLLIST REQUEST=READ_MULT Macro.....	1277
73. Return and Reason Codes for IXLLIST REQUEST=WRITE Macro.....	1309

74. Return and Reason Codes for IXLLIST REQUEST=WRITE_LCONTROLS Macro.....	1331
75. Return and reason codes for the IXLLOCK macro.....	1363
76. Mapping Macros for IXLLSTC.....	1372
77. IXLLSTC Version Support.....	1375
78. Return and Reason Codes for IXLLSTC Macro.....	1403
79. Mapping Macros for IXLLSTE.....	1418
80. IXLLSTE Version Support.....	1420
81. Return and Reason Codes for IXLLSTE Macro.....	1460
82. Mapping Macros for IXLLSTM.....	1476
83. IXLLSTM Version Support.....	1478
84. Return and Reason Codes for IXLLSTM Macro.....	1503
85. Return and Reason Codes for the IXLMG Macro.....	1529
86. Return and Reason Codes for the IXPURGE Macro.....	1536
87. Return and reason codes for the IXLREBLD macro.....	1551
88. Return and Reason Codes for the IXLRT Macro.....	1577
89. Return and Reason Codes for the IXLSYNCH Macro.....	1587
90. Return and Reason Codes for the IXLUSYNC Macro.....	1599
91. Return Codes for the IXLVECTR Macro with the MODIFYVECTORSIZE Parameter.....	1612
92. Return Codes for the IXLVECTR Macro with the LTVECENTRIES Parameter.....	1613
93. Return Codes for the IXLVECTR Macro with the TESTLISTSTATE Parameter.....	1614
94. Return Codes for the IXLVECTR Macro with the TESTLOCALCACHE Parameter.....	1615
95. Answer Area Macros for IXLZSTR.....	1619
96. Return and Reason Codes for the IXLZSTR Macro.....	1630

# About this information

---

This publication supports z/OS (5694-A01).

This publication provides the coding information, such as syntax and parameter descriptions, for the MVS services that enable multisystem applications and subsystems to:

- Run in a sysplex
- Share status information
- Send and receive messages using the XCF signalling function.
- Share data using the coupling facility.

These sysplex services can be used by authorized assembler language programs. An authorized program meets one or more of the following requirements:

- Runs in supervisor state
- Runs under PSW key 0-7
- Resides in an APF-authorized library

## Who should use this information

---

This publication is for programmers designing or modifying a multisystem application or subsystem to run in a sysplex and take advantage of the communication and data sharing functions available to sysplex members.

Programmers using this publication should be extremely knowledgeable about the MVS operating system and assembler language programming.

## Where to find more information

---

Where necessary, this publication references information in other publications, using shortened versions of the publication title. For complete titles and order numbers of the publications for all products that are part of z/OS, see [z/OS Information Roadmap](#).

The following table lists the title and order number for a publication related to another product.

Short title used in this publication	Title	Order number
<i>PR/SM Planning Guide</i>	<i>Processor Resource/Systems Manager Planning Guide</i>	GA22-7123
<i>PR/SM Planning Guide</i>	<i>Processor Resource/Systems Manager Planning Guide</i> (S/390® processors only)	GA22-7236



## How to provide feedback to IBM

---

We welcome any feedback that you have, including comments on the clarity, accuracy, or completeness of the information. For more information, see [How to send feedback to IBM](#).



## Summary of changes

---

This information includes terminology, maintenance, and editorial changes. Technical changes or additions to the text and illustrations for the current edition are indicated by a vertical line to the left of the change.

**Note:** IBM z/OS policy for the integration of service information into the z/OS product documentation library is documented on the z/OS Internet Library under [IBM z/OS Product Documentation Update Policy](http://www.ibm.com/docs/en/zos/latest?topic=zos-product-documentation-update-policy) ([www.ibm.com/docs/en/zos/latest?topic=zos-product-documentation-update-policy](http://www.ibm.com/docs/en/zos/latest?topic=zos-product-documentation-update-policy)).

## Summary of changes for z/OS 3.2

---

The following content is new, changed, or no longer included in z/OS 3.2.

### New

The following content is new.

#### September 2025 release

- None.

### Changed

The following content is changed.

#### September 2025 release

- [“Understanding IXCSP version support” on page 902](#) is updated.
- The parameter 3 in [“Syntax diagram” on page 903](#) is updated.

### Deleted

The following content is deleted.

#### September 2025 release

- None.

## Summary of changes for z/OS 3.1

---

This information contains no technical changes for this release.





---

# Chapter 1. How to read syntax diagrams

This section describes how to read syntax diagrams. It defines syntax diagram symbols, items that may be contained within the diagrams (keywords, variables, delimiters, operators, fragment references, operands) and provides syntax examples that contain these items.

Syntax diagrams pictorially display the order and parts (options and arguments) that comprise a command statement. They are read from left to right and from top to bottom, following the main path of the horizontal line.





For users accessing the Information Center using a screen reader, syntax diagrams are provided in dotted decimal format.

## Symbols

---

The following symbols may be displayed in syntax diagrams:

Symbol	Definition
--------	------------

- |   |  |
|---|--|
|    | Indicates the beginning of the syntax diagram.                   |
|    | Indicates that the syntax diagram is continued to the next line. |
|    | Indicates that the syntax is continued from the previous line.   |
|  | Indicates the end of the syntax diagram.                         |

## Syntax items

---

Syntax diagrams contain many different items. Syntax items include:

- Keywords - a command name or any other literal information.
- Variables - variables are italicized, appear in lowercase, and represent the name of values you can supply.
- Delimiters - delimiters indicate the start or end of keywords, variables, or operators. For example, a left parenthesis is a delimiter.
- Operators - operators include add (+), subtract (-), multiply (\*), divide (/), equal (=), and other mathematical operations that may need to be performed.
- Fragment references - a part of a syntax diagram, separated from the diagram to show greater detail.
- Separators - a separator separates keywords, variables or operators. For example, a comma (,) is a separator.

**Note:** If a syntax diagram shows a character that is not alphanumeric (for example, parentheses, periods, commas, equal signs, a blank space), enter the character as part of the syntax.

Keywords, variables, and operators may be displayed as required, optional, or default. Fragments, separators, and delimiters may be displayed as required or optional.

Item type	Definition
-----------	------------

<b>Required</b>	Required items are displayed on the main path of the horizontal line.
-----------------	---

## Optional

Optional items are displayed below the main path of the horizontal line.

## Default

Default items are displayed above the main path of the horizontal line.

## Syntax examples

The following table provides syntax examples.

Table 1. Syntax examples

Item	Syntax example
Required item. Required items appear on the main path of the horizontal line. You must specify these items.	►► KEYWORD — required_item ►►
Required choice. A required choice (two or more items) appears in a vertical stack on the main path of the horizontal line. You must choose one of the items in the stack.	►► KEYWORD — <div>required_choice1 required_choice2</div> ►►
Optional item. Optional items appear below the main path of the horizontal line.	►► KEYWORD — <div>optional_item</div> ►►
Optional choice. An optional choice (two or more items) appears in a vertical stack below the main path of the horizontal line. You may choose one of the items in the stack.	►► KEYWORD — <div>optional_choice1 optional_choice2</div> ►►
Default. Default items appear above the main path of the horizontal line. The remaining items (required or optional) appear on (required) or below (optional) the main path of the horizontal line. The following example displays a default with optional items.	►► KEYWORD — <div>default_choice1 optional_choice2 optional_choice3</div> ►►
Variable. Variables appear in lowercase italics. They represent names or values.	►► KEYWORD — <i>variable</i> ►►
Repeatable item. An arrow returning to the left above the main path of the horizontal line indicates an item that can be repeated. A character within the arrow means you must separate repeated items with that character. An arrow returning to the left above a group of repeatable items indicates that one of the items can be selected, or a single item can be repeated.	►► KEYWORD — <div>repeatable_item</div> ►► ►► KEYWORD — <div>repeatable_item</div> ►►

Table 1. Syntax examples (continued)

Item	Syntax example
<p>Fragment.</p> <p>The fragment symbol indicates that a labelled group is described below the main syntax diagram. Syntax is occasionally broken into fragments if the inclusion of the fragment would overly complicate the main syntax diagram.</p>	<p>The diagram illustrates a fragment symbol and a choice structure. At the top, a box labeled 'fragment' is preceded by a double arrow and followed by a double arrow. Below this, the word 'fragment' is written. The main part of the diagram shows a choice structure starting with a double arrow. A horizontal line leads to a vertical line that branches into two paths. The top path is labeled ',required_choice1' and ends with a double arrow. The bottom path is labeled ',required_choice2' and leads to a vertical line that branches into two paths: ',default_choice' and ',optional_choice'. Both of these paths end with a double arrow, which then connects back to the top path's double arrow, completing the choice structure.</p>



---

## Chapter 2. Specifying a Macro Version Number

Often there is more than one version of a macro, differentiated by additional keywords or new or expanded function. For example, version 1 of the IXLCONN macro provides several new keywords in support of the structure alter function, while version 3 of the IXLCONN macro provides a new keyword to support a new resource name length attribute of a lock structure.

You can request a specific version of a macro based on the needs of your application, but you should also be attuned to the storage constraints of the installation. The version of a macro might affect the length of the parameter list generated when the macro is assembled. The size of the parameter list might grow from release to release of z/OS, perhaps affecting the amount of storage your program needs.

---

### How to Request a Macro Version

To request a version of a macro, use the PLISTVER keyword. PLISTVER is the only parameter allowed on the list form of a macro form (MF), and it determines which parameter list the system generates. PLISTVER is optional. If you omit it, the system generates a parameter list for the lowest version that will accommodate the keywords specified. This is the IMPLIED\_VERSION default.

You also have the option of coding a *specific* version number using *plistver*, or of specifying MAX.

- *plistver* allows you to code a decimal value corresponding to the version of the macro you require. The decimal value you provide determines the amount of storage allotted for the parameter list.
- MAX allows you to request that the system generate a parameter list for the highest version number currently available. The amount of storage allotted for the parameter list will depend on the level of the system on which the macro is assembled.

IBM recommends, if your program can tolerate possible additional growth, that you always specify PLISTVER=MAX when creating the list form parameter list. MAX ensures that the list form parameter list is always long enough to hold whatever parameters might be specified on the execute form.

---

### General Considerations When Using PLISTVER

There are some general considerations that you should keep in mind when specifying the version of a macro with PLISTVER:

1. Not all macros in z/OS have the same version numbers. The version numbers need not be contiguous.
2. If PLISTVER is omitted, the macro generates a parameter list of the **lowest** version that allows all the parameters to be processed.
3. If you code *plistver*='n' and then specify any version 'n+1' keywords, the macro will not assemble.
4. If you code *plistver*='n' and do not specify any version 'n' keywords, the macro will generate a version 'n' parameter list.

Each macro in *z/OS MVS Programming: Sysplex Services Reference* that has one or more versions contains the PLISTVER keyword in the syntax diagram and in the parameter descriptions. In each such macro description, there is a topic entitled “Understanding *macname* Version Support,” where *macname* is the macro name. That topic specifies the range of values for *plistver* and lists the keywords and functions applicable for each version of the macro.



## Chapter 3. IHABLDP — Build Dump Parameter List Service

### Description

The IHABLDP macro builds the STRLIST parameter list that is specified as input on the SDUMPX macro when you request coupling facility structure information. IHABLDP builds the STRLIST parameter list in a block of storage that you provide to the macro. The STRLIST parameter list is mapped by the macro IHASDSTR. The IHABLDP macro does not initiate a dump request.

Each time you invoke IHABLDP, you specify a TYPE parameter to indicate which operation is to be performed on the STRLIST parameter list. The TYPE option determines which entry is to be built in the STRLIST parameter list.

- TYPE=INITIAL initializes the storage to binary zeros and generates an initialized header for the dump parameter list. The length of the dump parameter list header is 24 bytes.
- TYPE=STRUCTURE generates a structure entry in the dump parameter list. The length of this entry is 48 bytes.
- TYPE=STRRNG generates a structure range entry in the dump parameter list. The length of this entry is 12 bytes.
- TYPE=STROPT generates a structure option entry in the dump parameter list. The length of this entry is 12 bytes.
- TYPE=ENDLIST completes the construction of the dump parameter list. No entries will be added to the dump parameter list.

The data is dumped in the order in which you requested it. Serialized ranges are dumped before unserialized ranges.

The amount of storage required for the STRLIST parameter list depends on the number and type of entries in the parameter list. You determine the size by adding the number of bytes for each entry.

- The minimum STRLIST parameter list consists of a header and one structure entry (TYPE=STRUCTURE); thus the minimum size is 72 bytes.
- The maximum STRLIST parameter list can consist of 47 structures and 6 ranges. For each structure less than 47, you can specify an additional 10 ranges. For example:

```
47 structures and 6 ranges
46 structures and 16 ranges
44 structures and 36 ranges
```

If you specify more than the maximum allowed in the STRLIST, the system truncates the extraneous entries and indicates that the dump is a partial one by setting the SDRSTRLE flag in IHASDRSN, SDUMP Partial Reason Codes.

IHABLDP processes requests with incorrect parameter list lengths in the following ways:

#### 1. Minimum storage not specified at initialization

If less than 72 bytes of storage is specified when TYPE=INITIAL is requested, IHABLDP returns a return code of 8 and a reason code of 4 to indicate that insufficient space is available. Subsequent IHABLDP invocations that reference the same STRLIST parameter list also fail.

#### 2. Insufficient storage not specified after initialization

If sufficient storage is not available in an initialized STRLIST parameter list to add another entry, IHABLDP returns a return code of 8 and a reason code of 4. Subsequent IHABLDP invocations that reference the same STRLIST parameter list also are unable to add entries to the parameter list

and receive the same return and reason codes. When you invoke TYPE=ENDLIST to complete the construction of this parameter list, IHABLDP sets the total length of the list to 16 bytes less than the block of storage that you initially provided to the macro. The SDUMPX macro must be responsible to check that all entries in the parameter list are processed.

To use the macro, follow these guidelines:

- Specify TYPE=INITIAL before any other invocation of IHABLDP.
- Specify TYPE=STRUCTURE for each structure desired in the dump before specifying TYPE=STRRNG or TYPE=STROPT for that structure.

Once TYPE=STRUCTURE is processed, you can specify one or more TYPE=STRRNG or TYPE=STROPT options until all the desired ranges for a requested structure are specified. IHABLDP allows you prioritization of information that you request for a specified structure. You can specify the STRRNG and STROPT options in any order to give you more flexibility in requesting the most information at the earliest time.

- On TYPE=STRRNG, the starting range value must not be greater than the ending range value. If the starting value is greater, IHABLDP does not allow the structure range entry to be added to the parameter list and returns a return code of 8 and a reason code of 8.
- Specify TYPE=ENDLIST to complete the building of the parameter list. Make sure that you have specified all of the requested structures and their requested ranges.
- Do not attempt to change the built parameter list once you have specified TYPE=ENDLIST.

## Environment

---

The following are the environment requirements for the caller.

Environment	Environment requirement
Authorization:	One of the following: 1. Problem or supervisor state 2. Any PSW key
Dispatchable unit mode:	Task or SRB
Cross memory mode:	PASN=HASN, any SASN.
AMODE:	31-bit
ASC mode:	Primary or access register (AR)
Interrupt status:	Enabled or disabled for I/O and external interrupts
Locks:	No locks held
Control parameters:	Control parameters must be in the primary address space or be in an address space that is addressable through a public entry on the caller's dispatchable unit access list (DU-AL).

## Programming Requirements

---

If the program is in AR mode, issue SYSSTATE ASCENV=AR before invoking IHABLDP. SYSSTATE ASCENV=AR tells the system to generate code appropriate for AR mode.

## Restrictions

---

None.



## Input Register Information

---

Before issuing the IHABLDP macro, the caller does not have to place any information into any register unless using it in register notation for a particular parameter, or using it as a base register.

## Output Register Information

---

When control returns to the caller of the IHABLDP macro, the general purpose registers (GPRs) contain:

### Register Contents

- 0**  
Reason code if GPR15 return code is non-zero
- 1**  
Used as work register by the system
- 2 - 13**  
Unchanged
- 14**  
Used as work register by the system
- 15**  
Return code

The access registers contain:

### Register Contents

- 0 - 1**  
Used as work registers by the system
- 2 - 13**  
Unchanged
- 14 -15**  
Used as work registers by the system

**For registers that the system changes**, a caller who depends on these registers containing the same value before and after issuing the macro must save these registers before issuing the macro and restore them after the system returns control.

## Performance Implications

---

None.

## Understanding IHABLDP Version Support

---

The IHABLDP macro supports versions in the range of 0 - 1.

- Keywords not specifically noted here are supported by all versions starting with version 0 and higher of the IHABLDP macro.
- The following keywords and functions are supported by all versions starting with version 1 and higher of the IHABLDP macro.

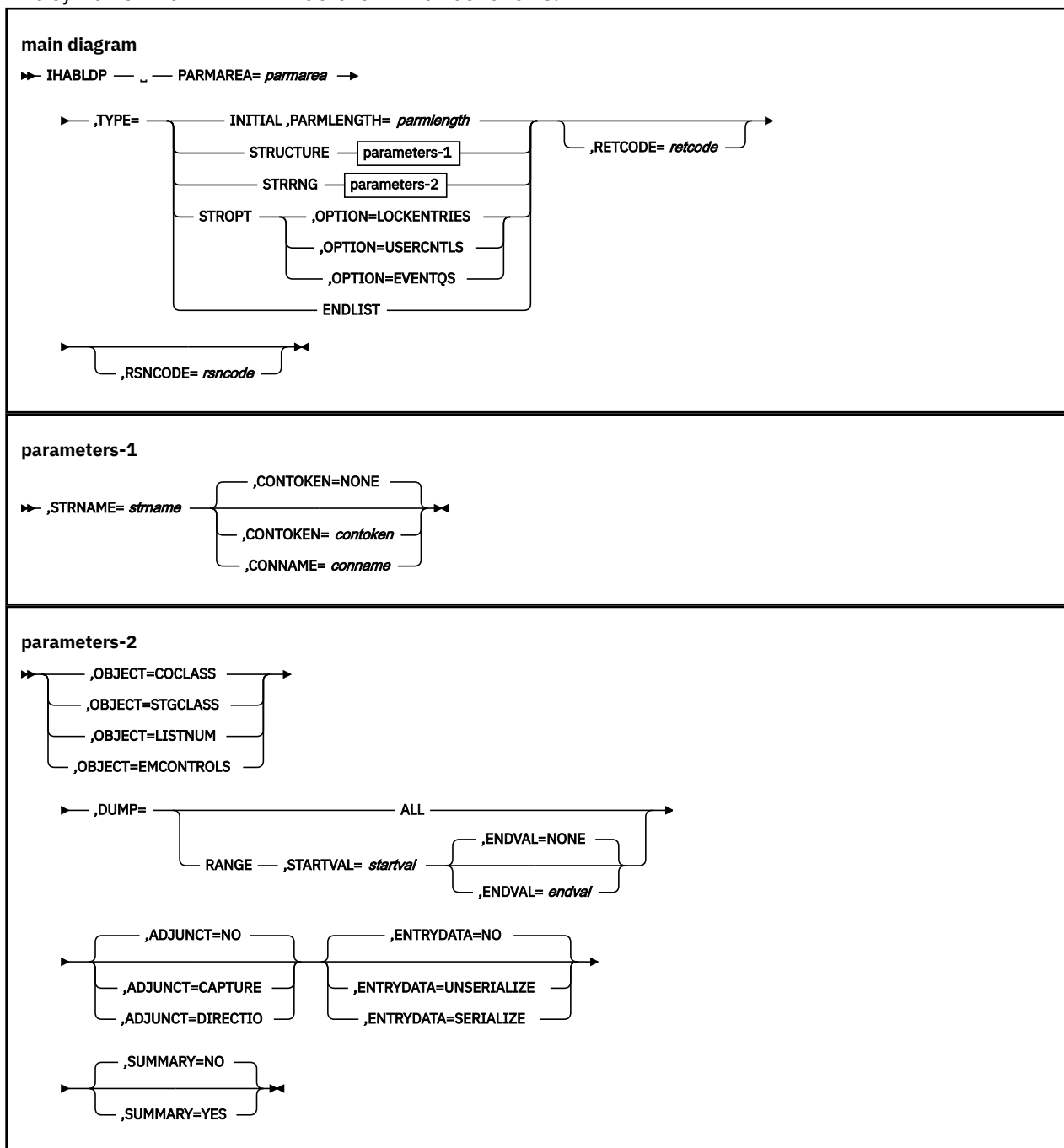
OBJECT=EMCONTROLS (only for keyed list structures allocated in a coupling facility with CFLEVEL=4 or higher)

OPTION=EVENTQS (only for keyed list structures allocated in a coupling facility with CFLEVEL=4 or higher)

Specify the version of the parameter list that you want generated with the PLISTVER keyword. See Chapter 2, “Specifying a Macro Version Number,” on page 5 for considerations when specifying the version of the parameter list with PLISTVER.

## Syntax

The syntax of the IHABLDP macro is written as follows:



## Parameter Descriptions

The parameter descriptions are listed in alphabetical order. Default values are underlined:

**,ADJUNCT=NO****,ADJUNCT=CAPTURE****,ADJUNCT=DIRECTIO**

Use this input parameter to specify whether or not the system should include adjunct data in the dump for all entries within the specified range. If the structure contains adjunct data, ADJUNCT specifies whether the adjunct data should be dumped.

**NO**

Do not dump adjunct data.

**CAPTURE**

While serialization is held, capture the adjunct data along with the entry controls.

**DIRECTIO**

After serialization is released, dump the adjunct data via direct I/O to the dump data set after the controls are captured.

**,CONNAME=conname**

Use this input parameter to specify the name of the connected user. When this keyword is specified for a cache structure, the VECTORINDEX with which the indicated user has registered interest in each entry that is dumped, will be included in the dump along with the directory entry information. When this keyword is specified for a list structure, it is ignored.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-character field that contains the name of the connected user.

**,CONTOKEN=NONE****,CONTOKEN=contoken**

Use this input parameter to specify the connect token that was returned by the IXLCONN service. The CONTOKEN uniquely identifies a user's connection to a structure. When this keyword is specified for a cache structure, the VECTORINDEX with which the indicated user has registered interest in each entry that is dumped, will be included in the dump along with the directory entry information. When this keyword is specified for a list structure, it is ignored.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-character field that contains the connect token returned by the IXLCONN service.

**,DUMP=ALL****,DUMP=RANGE**

Use this input parameter to specify the ranges to be dumped of a specified object.

**ALL**

All values of the specified object are to be dumped. If you also specify SUMMARY=NO (the default), the system dumps all the object controls and their associated entry controls. If you specify SUMMARY=YES, then only the object controls are dumped. For example:

- If you specify OBJECT=COCLASS with DUMP=ALL and SUMMARY=NO, the system dumps all of the object controls and their associated entries in all of the cast-out classes.
- If you specify OBJECT=COCLASS with DUMP=ALL and SUMMARY=YES, the system dumps only the object controls for all cast-out classes.

**RANGE**

A range of values of the specified object are to be dumped. If you also specify SUMMARY=NO (the default), the system dumps all the object controls and their associated entry controls. For example:

- If you specify OBJECT=COCLASS with DUMP=RANGE and SUMMARY=NO, the system dumps the object controls for the range of cast-out classes and also the cast-out classes' entry controls.
- If you specify OBJECT=COCLASS with DUMP=RANGE and SUMMARY=YES, the system dumps all of the object controls for the range of cast-out classes and no entries for any of the cast-out classes.

**,ENDVAL=NONE****,ENDVAL=endval**

Use this input parameter to specify the ending range value. If you do not specify ENDVAL or if you specify ENDVAL=NONE, then only the value specified for STARTVAL is dumped for this range.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword field that contains the ending range value.

**,ENTRYDATA=NO****,ENTRYDATA=UNSERIALIZE****,ENTRYDATA=SERIALIZE**

Use this input parameter to specify whether or not entry data should be included in the dump for all the entries in the specified range.

**NO**

Do not dump the entry data.

**UNSERIALIZE**

Release structure dump serialization before writing entry data to the dump data set.

**SERIALIZE**

Keep structure dump serialization until after the entry data has been written to the dump data set.

**,OBJECT=COCLASS****,OBJECT=STGCLASS****,OBJECT=LISTNUM****,OBJECT=EMCONTROLS**

Use this input parameter to specify what type range should be dumped.

**COCLASS**

The requested range is a range of cast-out classes. Use COCLASS only if the requested structure is a cache structure. If the requested structure is not a cache structure, the system dumps nothing for this entry.

**STGCLASS**

The requested range is a range of storage classes. Use STGCLASS only if the requested structure is a cache structure. If the requested structure is not a cache structure, the system dumps nothing for this entry.

**Note:** If a data entry appears in both a castout class and a storage class and you request that both classes are to be dumped in two IHABLDP parameter list entries, the system dumps the entry twice.

**LISTNUM**

The requested range is a range of list numbers. Use LISTNUM only if the requested structure is a list structure. If the requested structure is not a list structure, the system dumps nothing for this entry.

**EMCONTROLS**

The requested range is a range of list numbers for which event monitor controls should be dumped. Use EMCONTROLS only if the requested structure is a keyed list structure allocated in a coupling facility with CFLEVEL=4 or higher. If the requested structure is not such a list structure, the system dumps nothing for this entry.

**,OPTION=LOCKENTRIES****,OPTION=USERCNTLS****,OPTION=EVENTQS**

Use this input parameter to specify the option you want in the dump.

**LOCKENTRIES**

Include the lock table entries associated with the requested structure in the dump. Use LOCKENTRIES only when the requested structure is a list structure.

**USERCNTLS**

Include the user attach controls in the dump.

**EVENTQS**

Include in the dump the user event queues for a list structure allocated in a coupling facility with CFLEVEL=4 or higher.

**PARMAREA=*parmarea***

Use this output parameter to specify the name of the storage area that is to be used for building the dump parameter list. This storage area eventually will be passed as input to the SDUMPX macro.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the storage area that is to be used for building the dump parameter list.

**,PARMLENGTH=*parmlength***

Use this input parameter to specify the length of the storage area used for the dump parameter list.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword field containing the length, **in bytes**, of the storage area used for the dump parameter list.

**,RETCODE=*retcode***

Use this output parameter to specify the location in which the system is to copy the return code from GPR 15.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword field to contain the return code.

**,RSNCODE=*rsncode***

Use this output parameter to specify the location in which the system is to copy the reason code from GPR 0.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword field to contain the reason code.

**,STARTVAL=*startval***

Use this input parameter to specify the starting range value.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword field that contains the starting range value.

**,STRNAME=*strname***

Use this input parameter to specify the name of the cache or list structure for which information is being requested.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-character field that contains the name of the cache or list structure for which information is being requested.

**,SUMMARY=NO****,SUMMARY=YES**

Use this input parameter to specify whether or not the dump should include only a summary of the object requested. If SUMMARY=YES, the entry control information is not included; only structure and class information is included in the dump.

**NO**

Do not do a summary dump of the requested object; include entry controls.

**YES**

Do a summary dump of the requested object; do not include entry controls.

**,TYPE=INITIAL****,TYPE=STRUCTURE****,TYPE=STRNG****,TYPE=STROPT****,TYPE=ENDLIST**

Use this input parameter to specify the type of operation to be performed on the dump parameter list.

**INITIAL**

Initialize the storage to binary zeros and generate an initialized header for the STRLIST dump parameter list. The header for the dump parameter list is 24 bytes long.

**STRUCTURE**

Generate a structure entry in the STRLIST dump parameter list. The structure entry is 48 bytes long.

**STRRNG**

Generate a structure range entry in the STRLIST dump parameter list. The structure range entry is 12 bytes long.

**STROPT**

Generate a structure option entry in the STRLIST dump parameter list. The structure option entry is 12 bytes long.

**ENDLIST**

End building the STRLIST dump parameter list. You must specify ENDLIST when you complete building the STRLIST dump parameter list.

## ABEND Codes

---

None.

## Return and Reason Codes

---

When the system returns control to the caller, GPR 15 (and *retcode*, if you coded RETCODE) contains the return code and GPR 0 (and *rsncode*, if you coded RSNCODE) contains the reason code.

Table 2. Return and Reason Codes for the IHABLDP Macro		
Hexadecimal Return Code	Hexadecimal Reason Code	Meaning and Action
00	None	<b>Meaning:</b> IHABLDP completed successfully. <b>Action:</b> None
08	04	<b>Meaning:</b> There is insufficient space in the dump parameter list to add the requested entry. <b>Action:</b> Ensure that you have correctly calculated the amount of storage required for the STRLIST parameter list.
08	08	<b>Meaning:</b> The range entry was not added to the dump parameter list because the starting range value was greater than the ending range value. <b>Action:</b> Verify that you specified the correct starting and ending range values on the TYPE=STRRNG request.

## Chapter 4. IXCARM – Request Automatic Restart Management Services

### Description

Use the IXCARM macro to request the following services from the automatic restart management function of XCF:

- Register as an element of the automatic restart manager (REGISTER parameter)
- Wait until predecessor elements have been restarted, if applicable (WAITPRED parameter)
- Mark an element as ready to accept work (READY parameter)
- Deregister from the automatic restart manager (DEREGISTER parameter)
- Associate an element with another element (ASSOCIATE parameter).

For more information about the services performed by the IXCARM macro, see [z/OS MVS Programming: Sysplex Services Guide](#).

### Environment

The following are the environment requirements for the caller.

Environment	Environment requirement
Minimum authorization:	Supervisor state or PKM allowing key 0 - 7  Problem state and non-PKM system key is supported for all request types if the caller has SAF authority. SAF authorization to the entity IXCARM.elemtype.elementname is required. If elemtype is not specified, then the entity defaults to IXCARM.DEFAULT.elementname.
Dispatchable unit mode:	Task
Cross memory mode:	Any PASN, any HASN, any SASN
AMODE:	31-bit
ASC mode:	Primary or access register (AR)
Interrupt status:	Enabled for I/O and external interrupts
Locks:	No locks held
Control parameters:	Must be in the primary address space or be in an address/data space that is addressable through a public entry in the caller's dispatchable unit access list (DU-AL)

### Programming Requirements

If the program is in AR mode, issue the SYSSTATE ASCENV=AR macro before IXCARM. SYSSTATE ASCENV=AR tells the system to generate code appropriate for AR mode.

The IXCYARAA mapping macro provides the format of the area pointed to by the ANSAREA parameter. If you intend to use that area, include the IXCYARAA mapping macro in your program.

The areas specified by STARTTXT, EVENTEXITPL, and ANSAREA must be in the primary address space or be in an address/data space that is addressable through a public entry in the caller's dispatchable unit access list (DU-AL).

Include the IXCARM mapping macro in your program. This macro provides a list of equate symbols for users of IXCARM.

Initialize the area that contains the event exit parameter list (EVENTEXITPL) before issuing the IXCARM macro with the REQUEST=REGISTER parameter.

## Restrictions

The caller cannot have any enabled, unlocked task (EUT) FRRs established.

In general, all request types must be issued from the same home address space from which the IXCARM REQUEST=REGISTER request was issued. However, IXCARM requests for elements that represent abstract resources (ELEMBIND=CURSIS is specified) can be issued from any address space. IXCARM DEREGISTER requests for elements that represent jobs or started tasks (ELEMBIND=CURJOB is specified) can be issued from the master address space or the same home address space from which the REGISTER request was issued.

Registration for ELEMBIND=CURJOB elements can only be done from started tasks that were initiated with the START command. Started tasks initiated by TSO/E logins or the ASCRE (address space create) macro in a program will not be restartable by ARM.

The security administrator may control the use of automatic restart management by unauthorized applications through the use of RACF® or another security product. Unauthorized applications can use only element names that are registered as ELEMBIND=CURJOB elements. Ensure that you are authorized to issue the IXCARM macro for such unauthorized applications.

To define profiles that control unauthorized applications' use of automatic restart management, the security administrator can:

1. Define resource profile IXCARM.elemtype.elemname in the FACILITY class.
2. Specify the users who have access to automatic resource management services using the RACF PERMIT command.
3. Make sure the FACILITY class is active and generic profile checking is in effect. If in-storage profiles are maintained for the FACILITY class, refresh them.

For example, if a user wants to permit an unauthorized application with an *elemtype* of xxxxxxxx and an *elemname* of yyyyyyyyyyyyyyyy to use automatic restart management services, the security administrator can use the following commands:

```
RDEFINE FACILITY IXCARM.xxxxxxx.yyyyyyyyyyyyyyyy UACC(NONE)
PERMIT IXCARM.xxxxxxx.yyyyyyyyyyyyyyyy CLASS(FACILITY)
      ID(userid) ACCESS(UPDATE)
SETOPTS CLASSACT(FACILITY)
```

For information about RACF, see [z/OS Security Server RACF Security Administrator's Guide](#).

## Input Register Information

Before issuing the IXCARM macro, the caller does not have to place any information into any register unless using it in register notation for a particular parameter, or using it as a base register.

## Output Register Information

When control returns to the caller, the general purpose registers (GPRs) contain:

### Register Contents



**0**

If GPR 15 contains a 0, GPR 0 is used as a work register by the system; otherwise, GPR 0 contains a reason code.

**1**

Used as a work register by the system.

**2-14**

Unchanged.

**15**

Return code.

When control returns to the caller, the access registers (ARs) contain:

#### **Register Contents**

**0-1**

Used as work registers by the system

**2-14**

Unchanged

**15**

Used as a work register by the system

Some callers depend on register contents remaining the same before and after issuing a service. If the system changes the contents of registers on which the caller depends, the caller must save them before issuing the service, and restore them after the system returns control.

## Performance Implications

---

None.

## Understanding IXCARM Version Support

---

The IXCARM macro supports version 1 keywords and functions.

Specify the version of the parameter list that you want generated with the PLISTVER keyword. See [Chapter 2, “Specifying a Macro Version Number,” on page 5](#) for considerations when specifying the version of the parameter list with PLISTVER.

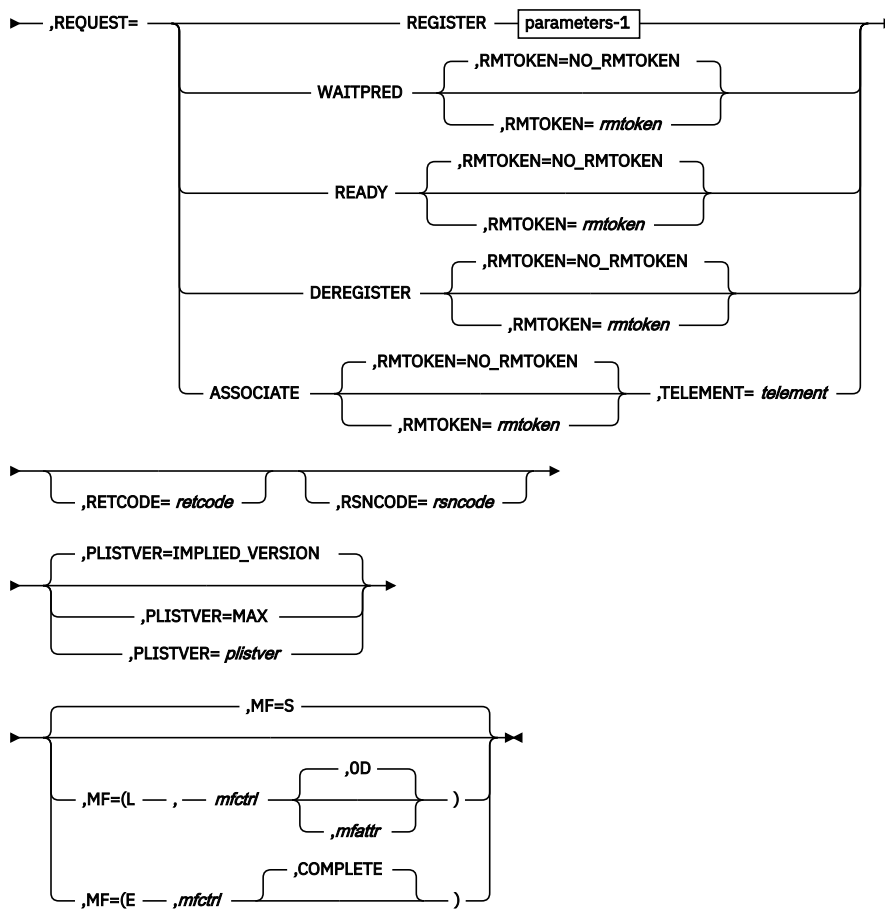
## Syntax

---

The syntax of the IXCARM macro is as follows:

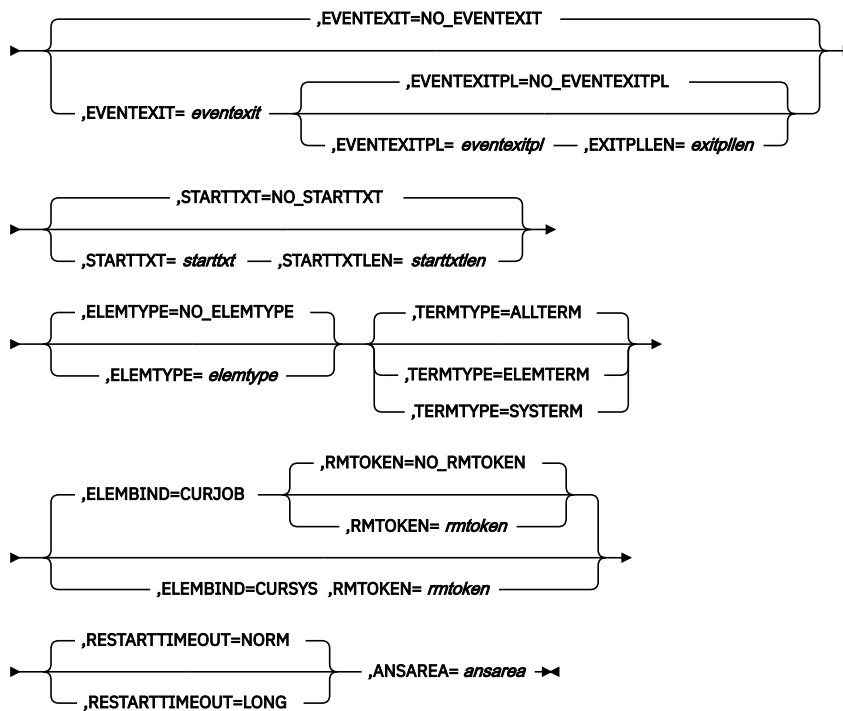
## main diagram

► IXCARM — b —►



**parameters-1**

➔ ,ELEMENT= *element* ➔



## Parameter Descriptions

The parameter descriptions are listed in alphabetical order. Default values are underlined.

**REQUEST=REGISTER****REQUEST=WAITPRED****REQUEST=READY****REQUEST=DEREGISTER****REQUEST=ASSOCIATE**

Use this input parameter to specify the type of request:

**REGISTER**

Requests registration (or re-registration) of a user with automatic restart manager. The user can be a batch job, started task, or an abstract resource that automatic restart manager is to restart when the work fails or the system on which the work is currently registered fails.

**WAITPRED**

Requests that automatic restart manager suspend the processing of the issuing program during restart processing for the element until any predecessor elements in the same restart group that are also being restarted have indicated that they are ready. The predecessors to the element that has issued this WAITPRED request are identified from “level” specifications in the active automatic restart management policy.

**READY**

Requests that automatic restart manager mark a registered user as READY to accept work. If a WAITPRED request has not previously been issued by the user, the READY request will perform an implicit WAITPRED to wait for predecessor elements to initialize. The system will suspend the current task until all predecessor elements have indicated that they are ready to accept work.

**DEREGISTER**

Requests deregistration of a registered user. The DEREGISTER request must be issued from the same address space that issued the REGISTER request, with the following exceptions:

- If the REGISTER request specified ELEMBIND=CURSYS, the Deregister may be done from any address space.
- Any element may be Deregistered from an address space termination resource manager running in the MASTER address space, if the RMTOKEN returned on the REGISTER is supplied. The application should only issue this request from the master address space while running under a resource manager. If the address space under which the REGISTER request was issued is not terminating and the application issues any IXCARM requests from that address space after a Deregister from the master address space, results are unpredictable.

**ASSOCIATE**

Requests that a registered user be associated with another element for takeover or restart processing purposes. This “association” suppresses all automatic restart manager restarts of the element identified in the TELEMENT parameter of this macro. The element issuing the ASSOCIATE request must be in the “available” state (that is, have issued the IXCARM REQUEST=READY macro).

If the issuer of the ASSOCIATE request terminates or deregisters from automatic restart manager, then the association is broken and the element specified in TELEMENT again becomes eligible for automatic restart manager restarts.

**,ANSAREA=ansarea**

Use this output area to specify an answer area to contain registration information on return from an IXCARM REQUEST=REGISTER request. This area must be 32 bytes long. Its format is described in the IXCYARAA mapping macro. Data returned in this area is valid only upon successful completion of this request (return code of 0 or 4), and if the validity flag is on (field ARAAREGTYPE is not zero).

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 32-byte area where the system will put the registration information.

**,ELEMBIND=CURJOB****,ELEMBIND=CURSYS**

Use this input parameter to specify the relationship between the element and the system. The element bind specifies the minimum bind that must be broken for automatic restart management to take action for the element. The ELEMBIND keyword indicates the type of resource the element represents.

**CURJOB**

The element has a minimum bind to the batch job or started task under which the element is registering. Specify ELEMBIND=CURJOB if the batch job or started task represented by the element needs to be restarted if it fails. The IXCARM request must be issued under a batch job or started task. There can be only one automatic restart management element registered with ELEMBIND=CURJOB per batch job or started task.

**CURSYS**

The element has a minimum bind to the system on which the element is registering. The element represents an abstract resource. An abstract resource is a program or a set of programs that is only associated with (or has a bind to) the system on which it is running. Specify ELEMBIND=CURSYS if the application registering with automatic restart management needs to be restarted only when the system fails. No address space, batch job, or started task failure will cause the element to be restarted. A REGISTER request that specifies ELEMBIND=CURSYS can be issued from any address space.

There can be more than one automatic restart management element registered with ELEMBIND=CURSYS per batch job, started task, or address space because the element is not associated with any of these units of work. There is no persistent restart text to restart the element. Therefore, the text of the command to restart this element must be specified either by the application when it registers, in the automatic restart management policy, or by an installation-written element restart exit. If the system fails and restart command text is not provided, the element will be deregistered.

ELEMBIND=CURSYS cannot be specified with TERMTYPE=ELEMTERM.

**,ELEMENT=element**

Use this input parameter to specify the name of the element that the automatic restart manager is to register. The rules for specifying an element name are:

- Valid characters are:
  - Uppercase alphabetic characters
  - The numbers 0 through 9
  - \$, #, @, and underscore (\_).
- The first character may not be a number.
- The name must be 16 characters long, padded on the right with blanks
- The name must be unique across the sysplex

Element names that start with A through I and SYS are reserved for use by IBM.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-byte field that contains the name of the element.

**,ELEMTYPE=NO\_ELEMTYPE****,ELEMTYPE=elementtype**

Use this input parameter to specify the type of element to be associated with this user. The element type is used:

- To facilitate the assigning of a “level” that is used to sequence the restarts of multiple elements during automatic restart management restarts after a system failure.
- To communicate the element type to listeners of the automatic restart management ENF Code 38 for events that pertain to this element.

The rules for specifying an element type are:

- Valid characters are:
  - Uppercase alphabetic characters
  - The numbers 0 through 9
  - \$, #, and @
- The first character may not be a number
- The type must be 8 characters long, padded on the right with blanks
- The element type does not have to be unique.

Element types that start with A through I and SYS are reserved for use by IBM. See [z/OS MVS Programming: Sysplex Services Guide](#) for a list of the assigned ELEMTYPE values.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-byte field that contains the type of element being registered.

**,EVENTEXIT=NO\_EVENTEXIT****,EVENTEXIT=eventexit**

Use this input parameter to specify the name of the event exit routine that is to be given control when certain events occur to the element. The event exit can be used to perform specialized processing for the element.

The event exit must be contained in the authorized linklib concatenation or LPA (not steplib or tasklib) of both the system where the REGISTER request was invoked and any potential target restart systems. The event exit will be invoked with a BALR/BAKR sequence of instructions.

The event exit name must be 8 characters long, padded on the right with blanks.

An unauthorized application cannot specify an event exit routine when registering with automatic restart management.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-byte field that contains the name of the event exit routine.

**,EVENTEXITPL=NO\_EVENTEXITPL****,EVENTEXITPL=eventexitpl**

Use this input parameter to specify the name of the event exit parameter list. If the IXCARM invocation is an AR ASC mode, this parameter list can be in either the primary address space or in a address/data space that is addressable through a public entry on the caller's DU-AL.

Automatic restart management makes a copy of this parameter list to pass to the event exit. The parameter list should not contain data that is MVS-image dependent (such as addresses) because the exit may run on another MVS image (an automatic restart manager may restart the element on another image).

If this parameter is specified, you must specify a value for EXITPLLEN.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the parameter list to be used by the event exit.

**,EXITPLLEN=exitplen**

Use this input parameter to specify the length of the event exit parameter list. The parameter list may be from 0 through 255 bytes in length. This parameter is required when EVENTEXITPL is specified.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword field that contains the length of the event exit parameter list.

**,MF=S****,MF=(L,mfctrl)****,MF=(L,mfctrl,mfattr)****,MF=(L,mfctrl,0D)****,MF=(M,mfctrl)****,MF=(M,mfctrl,COMPLETE)****,MF=(M,mfctrl,NOCHECK)****,MF=(E,mfctrl)****,MF=(E,mfctrl,COMPLETE)****,MF=(E,mfctrl,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

**,mfctrl**

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

**,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

**,COMPLETE****,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

**COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if `SMILE=var` were an optional parameter and the default is `SMILE=NO_SMILE` then it would not be documented. However, if the default was `SMILE=-:-`, then it would be documented because a value would be the default.

**NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

**,PLISTVER=IMPLIED\_VERSION**

**,PLISTVER=MAX**

**,PLISTVER=plistver**

Use this input parameter to specify the version of the macro. See “[Understanding IXCARM Version Support](#)” on page 17 for a description of the options available with PLISTVER.

**,RESTARTTIMEOUT=NORM**

**,RESTARTTIMEOUT=LONG**

Use this input parameter to indicate how long automatic restart management should wait for a restarted element to re-register. This parameter is applicable only when the element's restart timeout is determined by automatic restart manager's default value. If the installation's active ARM policy specifies a `RESTART_TIMEOUT` value for this element, the `RESTARTTIMEOUT` specification on a `REGISTER` request is ignored.

**NORM**

Automatic restart management is to wait 5 minutes (the normal time-out value) for this restarted element to issue the `IXCARM REQUEST=REGISTER` request.

**LONG**

Automatic restart management is to wait up to 6 hours (the long time-out value) for this restarted element to issue the `IXCARM REQUEST=REGISTER` request.

**,RETcode=retcode**

Use this output parameter to specify a field to contain the return code. (The return code is also returned in register 15.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the return code when the request has completed.

**,RMTOKEN=NO\_RMTOKEN**

**,RMTOKEN=rmtoken**

Use this input and output parameter to specify the 16-byte field that contains a restart manager token.

- The restart manager token is returned by an `IXCARM REQUEST=REGISTER` request. The contents of the restart token must not be modified by the application registering the element.
- When registering with `ELEMBIND=CURJOB`, applications can specify the restart manager token on subsequent `REQUEST=DEREGISTER` requests issued from the master address space. Applications cannot specify the restart manager token on `READY`, `WAITPRED`, `ASSOCIATE`, or `DEREGISTER` requests from the address space in which the element is registered.
- When registering with `ELEMBIND=CURSYS`, applications must specify the restart manager token on subsequent `READY`, `WAITPRED`, `ASSOCIATE`, and `DEREGISTER` requests to identify the element. The applications can specify the restart manager token on these requests from any address space.

If the IXCARM invocation is in AR ASC mode, the area containing the restart manager token can be in either the primary address space or in an address or data space that is addressable through a public entry on the caller's DU-AL.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the 16-byte field containing the restart manager token.

**,RSNCODE=rsncode**

Use this output parameter to specify a field to contain the reason code returned, if applicable. (The reason code is also returned in register 0.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the reason code (if any) when the request has completed.

**,STARTTXT=NO\_STARTTXT**

**,STARTTXT=starttxt**

Use this input parameter to specify the text of the command to be used to restart this element. When command text is specified, STARTTXTLEN must contain the length of the command text.

If the IXCARM invocation is in AR ASC mode, the text can be in either the primary address space or in an address/data space that is addressable through a public entry on the caller's DU-AL.

If this parameter is specified, it will take precedence over the automatic restart manager's reuse of the START command that most recently started this element (persistent restart text). A command specified with the STARTTXT parameter can be overridden by a RESTART\_METHOD specified for the element in the active ARM policy, unless the RESTART\_METHOD defaults to or specifies PERSIST.

The STARTTXT parameter is valid only for started tasks. If specified with a REGISTER request from a batch job, the registration request will be rejected.

**Note:**

1. This parameter is valid only for started tasks.
2. A command provided through the STARTTXT parameter cannot contain symbolic substitution parameters (such as &SYSNAME).
3. An unauthorized application cannot specify the STARTTXT parameter.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the text of the command to be used to restart this element.

**,STARTTXTLEN=starttxtlen**

Use this input parameter to specify the length of the command specified in the STARTTXT parameter. This command may be from 0 through 126 bytes in length.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword field that contains the length of the command text.

**,TELEMENT=telement**

Use this input parameter to specify the name of the element that should be associated with the element issuing this IXCARM request. The element name must be 16 characters long, padded on the right with blanks. This name must meet all of the requirements described under the ELEMENT parameter.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-byte field that contains the name of the element to be associated with the element issuing the IXCARM macro.

**,TERMTYPE=ALLTERM**

**,TERMTYPE=ELEMTERM**

**,TERMTYPE=SYSTEM**

Use this input parameter to specify the type of termination for which this element may be restarted. A specification of TERMTYPE for this element in the active ARM policy will override this keyword, except when the policy specifies TERMTYPE=ELEMTERM and the IXCARM macro specifies ELEMBIND=CURSYS.

**ALLTERM**

Automatic restart manager is to restart this element for all unexpected failures as appropriate.

The ELEMBIND keyword determines which types of failures are appropriate to restart the element.



**ELEMTERM**

Automatic restart manager is to restart the element when the element fails but not when the system on which the element is registered is removed from the sysplex or unexpectedly terminates. TERMTYPE=ELEMTERM cannot be specified with ELEMbind=CURSYS.

**SYSTEM**

Automatic restart manager is to restart this element when the system on which the element is registered is removed from the sysplex or unexpectedly terminates, but not when the element unexpectedly terminates.

**Note:** The installation can override this parameter by specifying TERMTYPE for this element in an installation-written policy.

## ABEND Codes

---

None.

## Return and Reason Codes

---

When control returns from IXCARM:

- GPR 15 (and *retcode*, if you coded the RETCODE parameter) contains a return code.
- GPR 0 (and *rsncode*, if you coded the RSNCODE parameter) contains a reason code, if GPR 15 contains a non-zero return code.

The IXCARM macro provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

**0**

IXCARMRC0

**4**

IXCARMRC4

**8**

IXCARMRC8

**C**

IXCARMRC12

**10**

IXCARMRC16

Table 3. Return and Reason Codes for the IXCARM Macro

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
00	None.	<p><b>Equate Symbol:</b> None.</p> <p><b>Meaning:</b> IXCARM completed successfully. If this was a REGISTER request, this was the first time the element registered with automatic restart management.</p> <p>On a Deregister request, the request was either performed successfully, or the caller attempted to deregister an element that was either not valid or no longer registered.</p> <p><b>Note:</b> This return code could also be received if a timeout occurred during restart processing. Automatic restart management considers this the initial registration of this element.</p> <p><b>Action:</b> None. However, if this element had been deregistered due to an error during restart processing, some cleanup may be necessary.</p>
04	104	<p><b>Equate Symbol:</b> IXCARMPERJCL</p> <p><b>Meaning:</b> IXCARM REQUEST=REGISTER was issued by an element that is being restarted with the same JCL or command text that was used for the previous start of this job.</p> <p><b>Action:</b> None; however, the element might need to perform some cleanup to continue processing.</p>
04	108	<p><b>Equate Symbol:</b> IXCARMNEWJCL</p> <p><b>Meaning:</b> IXCARM REQUEST=REGISTER was issued by an automatic restart management element that is being restarted with JCL, command text that was provided by an automatic restart management exit or by the automatic restart manager policy, or start text that was specified on the original IXCARM REQUEST=REGISTER.</p> <p><b>Action:</b> None; however, the element might need to perform some cleanup to continue processing.</p>
04	204	<p><b>Equate Symbol:</b> IXCAMPREDTIMEOUT</p> <p><b>Meaning:</b> IXCARM REQUEST=WAITPRED was issued but the predecessor element did not issue an IXCARM REQUEST=READY within its specified time interval.</p> <p><b>Action:</b> None required. However, if your program cannot run without the predecessor element, then some installation-defined action may be necessary.</p>

Table 3. Return and Reason Codes for the IXCARM Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
04	304	<p><b>Equate Symbol:</b> IXCARMREADYTIMEOUT</p> <p><b>Meaning:</b> The IXCARM REQUEST=READY completed but a predecessor of this element did not issue an IXCARM REQUEST=READY within its specified time interval. This reason code is issued only if the caller did not issue IXCARM REQUEST=WAITPRED previously, but has predecessor elements defined in the automatic restart manager policy.</p> <p><b>Action:</b> None required. However, if your program cannot run without the predecessor element, then some installation-defined action may be necessary.</p>
08	14	<p><b>Equate Symbol:</b> IXCARMNOTREG</p> <p><b>Meaning:</b> Program error. Either the caller is not a registered element of the automatic restart manager or the element represents an abstract resource (the element registered with ELEMBIND=CURSYS) and did not specify a valid RMTOKEN on an IXCARM READY, WAITPRED, ASSOCIATE, or DEREGISTER request. Note that for elements that represent jobs or started tasks (the element registered with ELEMBIND=CURJOB), specifying RMTOKEN on a READY, WAITPRED, and ASSOCIATE request is not allowed and results in this reason code. Specifying RMTOKEN on a DEREGISTER request is not allowed and results in this reason code unless the request is issued from the master address space.</p> <p>A registered element could have been deregistered for any of the following reasons:</p> <ul style="list-style-type: none"> <li>• IXCARM DEREGISTER request issued for the element.</li> <li>• An internal ARM error occurred processing a prior IXCARM request. The prior IXCARM request would have completed with IXCARMRC16/IXCARMARMERR.</li> <li>• The element was deregistered via the SETXCF FORCE,ARMDEREGISTER operator command.</li> </ul> <p><b>Action:</b> A request other than IXCARM REQUEST=REGISTER was issued out of sequence. Ensure that your program issues an IXCARM REQUEST=REGISTER prior to requesting any other functions from the automatic restart manager.</p>

Table 3. Return and Reason Codes for the IXCARM Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
08	18	<p><b>Equate Symbol:</b> IXCARMINVANSADDR</p> <p><b>Meaning:</b> Program error. The system cannot access the answer area provided with this request. The element was successfully registered, but the answer data could not be returned.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The answer area address has not been overlaid.</li> <li>• If IXCARM was called in AR mode: <ul style="list-style-type: none"> <li>– The SYSSTATE ASCENV=AR macro was issued prior to this macro.</li> <li>– If the answer area address was specified using explicit register notation, the corresponding access register was updated accordingly.</li> <li>– This area is either in the primary address space or in an address/data space that is addressable through a public entry on the caller's DU-AL.</li> </ul> </li> </ul>
08	1C	<p><b>Equate Symbol:</b> IXCARMINVANSALET</p> <p><b>Meaning:</b> Program error. The ALET that qualifies the address of the answer area is not associated with a valid entry on the DU-AL.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The SYSSTATE ASCENV=AR macro was issued prior to this macro.</li> <li>• If the answer area address was specified using explicit register notation, the corresponding access register was updated accordingly.</li> <li>• This area is either in the primary address space or in an address/data space that is addressable through a public entry on the caller's DU-AL.</li> </ul>
08	20	<p><b>Equate Symbol:</b> IXCARMINVRMTADDR</p> <p><b>Meaning:</b> Program or environmental error. The system cannot access the RMTOKEN value.</p> <p><b>Action:</b> Ensure that</p> <ul style="list-style-type: none"> <li>• The token area has not been overlaid.</li> <li>• If IXCARM was called in AR mode: <ul style="list-style-type: none"> <li>– The SYSSTATE ASCENV=AR macro was issued prior to this macro.</li> <li>– If the token address was specified using explicit register notation, the corresponding access register was updated accordingly.</li> <li>– This area is either in the primary address space or in an address/data space that is addressable through a public entry on the caller's DU-AL.</li> </ul> </li> </ul>

Table 3. Return and Reason Codes for the IXCARM Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
08	24	<p><b>Equate Symbol:</b> IXCARMINVRTMALET</p> <p><b>Meaning:</b> Program or environmental error. The ALET that qualifies the address of the RMTOKEN is not associated with a valid entry on the DU-AL.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The SYSSTATE ASCENV=AR macro was issued prior to this macro.</li> <li>• If the answer area address was specified using explicit register notation, the corresponding access register was updated accordingly.</li> <li>• This area is either in the primary address space or in an address/data space that is addressable through a public entry on the caller's DU-AL.</li> </ul>
08	2C	<p><b>Equate Symbol:</b> IXCARMINVELEMNAME</p> <p><b>Meaning:</b> Program error. For IXCARM REQUEST=REGISTER, the name specified for ELEMENT was not valid. For IXCARM REQUEST=ASSOCIATE, the name specified for TELEMENT was not valid.</p> <p><b>Action:</b> Verify that the element name was not inadvertently overlaid. The element name must adhere to the rules listed under the description of the ELEMENT parameter.</p> <p>For REQUEST=ASSOCIATE, TELEMENT must be the name of an element that is already registered with automatic restart management.</p>
08	30	<p><b>Equate Symbol:</b> IXCARMREQUESTOVERLAP</p> <p><b>Meaning:</b> Program error. An IXCARM request for an element registered with or defaulted to ELEMBIND=CURJOB is already being processed for this address space. The IXCARM request cannot be issued until a previous IXCARM request from this address space has been completed.</p> <p><b>Action:</b> An IXCARM request cannot be issued until all previous IXCARM requests from this address space have completed. Check your protocol to determine how two requests could be outstanding at the same time.</p>
08	34	<p><b>Equate Symbol:</b> IXCARMAMODE24</p> <p><b>Meaning:</b> Program error. The IXCARM macro was issued in 24-bit addressing mode.</p> <p><b>Action:</b> IXCARM must be invoked in 31-bit addressing mode. Correct the problem and rerun the program.</p>

Table 3. Return and Reason Codes for the IXCARM Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
08	40	<p><b>Equate Symbol:</b> IXCARMRSVNOT0</p> <p><b>Meaning:</b> Program error. A reserved field in the parameter list is not zero.</p> <p><b>Action:</b> Verify that:</p> <ul style="list-style-type: none"> <li>• The parameter list was not inadvertently overlaid.</li> <li>• The parameter list was initialized.</li> <li>• Your program was assembled with the correct macro library for the release of MVS your program is running on.</li> <li>• The PLISTVER specified is correct.</li> </ul>
08	A0	<p><b>Equate Symbol:</b> IXCARMINVR0</p> <p><b>Meaning:</b> System error.</p> <p><b>Action:</b> Make sure your program was assembled with the correct macro library for the release of MVS your program is running on. If the macro version is correct, retry the request at least once. If the problem persists, record the return and reason code, and supply them to the appropriate IBM support personnel.</p>
08	A4	<p><b>Equate Symbol:</b> IXCARMR0TYPECONFL</p> <p><b>Meaning:</b> System error.</p> <p><b>Action:</b> Verify that:</p> <ul style="list-style-type: none"> <li>• The parameter list was not inadvertently overlaid.</li> <li>• The parameter list was initialized.</li> <li>• Your program was assembled with the correct macro library for the release of MVS your program is running on.</li> <li>• If IXCARM was called in AR mode: <ul style="list-style-type: none"> <li>– The SYSSTATE ASCENV=AR macro was issued prior to this macro.</li> <li>– If the parameter list address was specified using explicit register notation, the corresponding access register was updated accordingly.</li> <li>– This area is either in the primary address space or in an address/data space that is addressable through a public entry on the caller's DU-AL.</li> </ul> </li> </ul> <p>If everything is verified, retry the request at least once. If the problem persists, record the return and reason code, and supply them to the appropriate IBM support personnel.</p>

Table 3. Return and Reason Codes for the IXCARM Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
08	100	<p><b>Equate Symbol:</b> IXCARMINVPLISTALET</p> <p><b>Meaning:</b> Program error. The ALET that qualifies the address of the parameter list is not associated with a valid DU-AL entry.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• Your program is not intended to run in primary ASC mode.</li> <li>• The SYSSTATE ASCENV=AR macro was issued prior to this macro.</li> <li>• If the parameter list address was specified using explicit register notation, the corresponding access register was updated accordingly.</li> <li>• The parameter list is either in the primary address space or in an address/data space that is addressable through a public entry on the caller's DU-AL.</li> </ul>
08	104	<p><b>Equate Symbol:</b> IXCARMBADVERSION</p> <p><b>Meaning:</b> Program error. The version number specified in the IXCARM parameter list is not valid for the release of MVS this program is running on.</p> <p><b>Action:</b> Ensure that your program:</p> <ul style="list-style-type: none"> <li>• Specified the correct version on the PLISTVER parameter.</li> <li>• Did not overlay the parameter list storage.</li> <li>• Was assembled with the correct macro library for the release of MVS your program is running on.</li> </ul>
08	108	<p><b>Equate Symbol:</b> IXCARMBADREQUEST</p> <p><b>Meaning:</b> Program error. The function specified on the REQUEST parameter of the IXCARM macro is not valid.</p> <p><b>Action:</b> Verify that:</p> <ul style="list-style-type: none"> <li>• The parameter list was not inadvertently overlaid.</li> <li>• Your program was assembled with the correct macro library for the release of MVS your program is running on.</li> </ul>

Table 3. Return and Reason Codes for the IXCARM Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
08	10C	<p><b>Equate Symbol:</b> IXCARM ParmErr</p> <p><b>Meaning:</b> Program error. An error occurred when the system tried to access the parameter list.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• Ensure that the parameter list address has not been overlaid.</li> <li>• Ensure that the correct parameter list storage area was specified.</li> <li>• If your program is running in AR ASC mode: <ul style="list-style-type: none"> <li>– Ensure that you specified SYSSTATE ASCENV=AR before issuing the IXCARM macro.</li> <li>– If the parameter list address was specified using explicit register notation, the corresponding access register was updated accordingly.</li> <li>– This area must be either in the primary address space or in an address/data space that is addressable through a public entry on the caller's DU-AL.</li> </ul> </li> <li>• Ensure that the parameter list storage area was not inadvertently freed by your program.</li> <li>• Ensure that your program was assembled with the correct macro library for the release of MVS on which your program is running.</li> </ul>
08	110	<p><b>Equate Symbol:</b> IXCARM StartErr</p> <p><b>Meaning:</b> Program error. An error occurred when the system tried to access the start text.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The start text address has not been overlaid.</li> <li>• The correct STARTTXT address was specified.</li> <li>• The STARTTXT storage area was not inadvertently freed by your program.</li> <li>• If IXCARM was called in AR mode: <ul style="list-style-type: none"> <li>– The SYSSTATE ASCENV=AR macro was issued prior to this macro.</li> <li>– If the start text address was specified using explicit register notation, the corresponding access register was updated accordingly.</li> <li>– The start text is either in the primary address space or in an address/data space that is addressable through a public entry on the caller's DU-AL.</li> </ul> </li> </ul>



Table 3. Return and Reason Codes for the IXCARM Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
08	114	<b>Equate Symbol:</b> IXCARMSTARTLEN <b>Meaning:</b> Program error. The length of the start text is not valid. <b>Action:</b> Ensure that the parameter list was not inadvertently overlaid. Valid lengths for the start text are from 0 through 126.
08	118	<b>Equate Symbol:</b> IXCARMNOTTASKMODE <b>Meaning:</b> Program error. The caller is not running in task mode. <b>Action:</b> Correct your program so that it issues IXCARM only while in task mode.
08	11C	<b>Equate Symbol:</b> IXCARMNOTENABLED <b>Meaning:</b> The caller is not enabled. <b>Action:</b> Correct your program so that it does not issue IXCARM while it is disabled.
08	120	<b>Equate Symbol:</b> IXCARMHASLOCK <b>Meaning:</b> Program error. The caller of IXCARM holds a lock. <b>Action:</b> Correct your program so that it does not issue IXCARM while it is holding a lock.
08	124	<b>Equate Symbol:</b> IXCARMHASEUTFRR <b>Meaning:</b> Program error. The caller of IXCARM has an EUT FRR established. <b>Action:</b> Correct your program so that it does not issue IXCARM while it has an EUT FRR established. An ESTAE should be used if recovery needs to be established for this program.
08	128	<b>Equate Symbol:</b> IXCARMJESERR <b>Meaning:</b> Program error. A REGISTER request failed because of an error in JES. <b>Action:</b> Analyze problem with JES.
08	12C	<b>Equate Symbol:</b> IXCARMJOURNAL <b>Meaning:</b> Program error. The caller is a candidate for checkpoint/restart and is not eligible to be restarted by automatic restart management. <b>Action:</b> If you want to use automatic restart management to restart your jobs, you cannot use a checkpoint/restart.
08	130	<b>Equate Symbol:</b> IXCARMINVELEMTYPE <b>Meaning:</b> Program error. The name specified for the element type was not valid. <b>Action:</b> Make sure the element type specified adheres to the rules outlined under the description of the ELEMTYPE parameter.

Table 3. Return and Reason Codes for the IXCARM Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
08	134	<p><b>Equate Symbol:</b> IXCARMWRONGCALLERTYPE</p> <p><b>Meaning:</b> Program error. The caller was neither a started task nor a batch job, but specified a bind to the current batch job or started task.</p> <p><b>Action:</b> Register the element with a bind to the current system.</p>
08	138	<p><b>Equate Symbol:</b> IXCARMCANCELLED</p> <p><b>Meaning:</b> Environmental error. A register request was received from a program that is in an address space that is being cancelled. The CANCEL command was issued without the ARMRESTART parameter, therefore, the registration of this element is not allowed.</p> <p><b>Action:</b> None; however, you should make sure that your program is not issuing the IXCARM macro with the REQUEST=REGISTER parameter from a recovery routine or resource manager.</p>
08	13C	<p><b>Equate Symbol:</b> IXCARMRACRFAIL</p> <p><b>Meaning:</b> Environmental error. The issuer of the IXCARM macro does not have the required SAF authorization.</p> <p><b>Action:</b> Determine why the program issuing the IXCARM macro does not have the proper SAF authorization. If this program was restarted with new command text (specified in the policy, on the STARTTXT parameter of the IXCARM macro, or in the element-restart installation exit), verify that the command text did not cause this problem.</p>
08	140	<p><b>Equate Symbol:</b> IXCARMINVTERMTYPE</p> <p><b>Meaning:</b> Program error. The TERMTYPE value specified is not valid.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The parameter list address has not been overlaid.</li> <li>• If your program is running in AR ASC mode: <ul style="list-style-type: none"> <li>– Your program specified SYSSTATE ASCENV=AR before issuing the IXCARM macro.</li> <li>– If the parameter list address was specified using explicit register notation, the corresponding access register was updated accordingly.</li> <li>– This area is either in the primary address space or in an address/data space that is addressable through a public entry on the caller's DU-AL.</li> </ul> </li> <li>• The parameter list storage area was not inadvertently freed by your program.</li> <li>• Your program was assembled with the correct macro library for the release of MVS your program is running on.</li> </ul>

Table 3. Return and Reason Codes for the IXCARM Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
08	144	<p><b>Equate Symbol:</b> IXCARMINVRESTTIMEOUT</p> <p><b>Meaning:</b> Program error. The RESTARTTIMEOUT value specified is not valid.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The parameter list address has not been overlaid.</li> <li>• If your program is running in AR ASC mode: <ul style="list-style-type: none"> <li>– Your program specified SYSSTATE ASCENV=AR before issuing the IXCARM macro.</li> <li>– If the parameter list address was specified using explicit register notation, the corresponding access register was updated accordingly.</li> <li>– This area is either in the primary address space or in an address/data space that is addressable through a public entry on the caller's DU-AL.</li> </ul> </li> <li>• The parameter list storage area was not inadvertently freed by your program.</li> <li>• Your program was assembled with the correct macro library for the release of MVS your program is running on.</li> </ul>
08	148	<p><b>Equate Symbol:</b> IXCARMSAVEFAIL</p> <p><b>Meaning:</b> Environmental error. Register request prohibited by JES.</p> <p><b>Action:</b> This element cannot be restarted by automatic restart management.</p>
08	14C	<p><b>Equate Symbol:</b> IXCARMBATCHSTARTTXT</p> <p><b>Meaning:</b> Program error. An element registering under a batch job with a bind to the job specified STARTTXT with its register request.</p> <p><b>Action:</b> Do one of the following:</p> <ul style="list-style-type: none"> <li>• If you have a batch job that needs different JCL if it is restarted, then use the policy to point to this information.</li> <li>• Code an automatic restart manager element restart exit that will point to the different JCL if it is restarted.</li> </ul>

Table 3. Return and Reason Codes for the IXCARM Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
08	150	<p><b>Equate Symbol:</b> IXCARMELEMNAMEINUSE</p> <p><b>Meaning:</b> Program error. The element name specified for this REGISTER request is already registered.</p> <p><b>Action:</b> Element names must be unique across a sysplex. To determine what element names are already in use, you could issue the IXCQUERY macro and request ARMSTATUS. This will provide you with a list of element names that are already in use. A system operator may want to issue the DISPLAY XCF,ARMSTATUS command to verify that the name is really being used. If the element has not been deregistered by ARM when the bind to the element has been broken, an operator can issue the SETXCF FORCE,ARMDEREGISTER command to deregister the element and allow a new registration.</p>
08	154	<p><b>Equate Symbol:</b> IXCARMADDRSPACEDUP</p> <p><b>Meaning:</b> Program error. An element with a bind to the batch job or started task is already registered with the automatic restart manager. Only one element per batch job or started task can register with a bind specification of CURJOB.</p> <p><b>Action:</b> Ensure that the application is not registering more than once or use the ELEMBIND=CURSYS keyword option if appropriate.</p>
08	158	<p><b>Equate Symbol:</b> IXCARMEXITPARM</p> <p><b>Meaning:</b> Program error or environmental error. The system cannot access the event exit parameter list specified in the EVENTEXITPL parameter.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The parameter list address has not been overlaid.</li> <li>• The correct parameter list address was specified.</li> <li>• The parameter list storage area was not inadvertently freed by your program.</li> <li>• If IXCARM was called in AR mode: <ul style="list-style-type: none"> <li>– The SYSSTATE ASCENV=AR macro was issued prior to this macro.</li> <li>– If the parameter list address was specified using explicit register notation, the corresponding access register was updated accordingly.</li> <li>– The event exit parameter list is either in the primary address space or in an address/data space that is addressable through a public entry on the caller's DU-AL.</li> </ul> </li> </ul>

Table 3. Return and Reason Codes for the IXCARM Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
08	15C	<p><b>Equate Symbol:</b> IXCARMEXITLEN</p> <p><b>Meaning:</b> Program error. The value specified in EXITPLEN has exceeded the maximum length.</p> <p><b>Action:</b> See the description of the EXITPLEN parameter for the limits on this length.</p> <p>Ensure that:</p> <ul style="list-style-type: none"> <li>• The parameter list length has not been overlaid.</li> <li>• The correct parameter list length was specified.</li> <li>• If IXCARM was called in AR mode: <ul style="list-style-type: none"> <li>– The SYSSTATE ASCENV=AR macro was issued prior to this macro.</li> <li>– If the parameter list address was specified using explicit register notation, the corresponding access register was updated accordingly.</li> <li>– The event exit parameter list is either in the primary address space or in an address/data space that is addressable through a public entry on the caller's DU-AL.</li> </ul> </li> </ul>
08	160	<p><b>Equate Symbol:</b> IXCARMEXITNAME</p> <p><b>Meaning:</b> Program error or environmental error. The system cannot access the name of the event exit.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The correct event exit name was specified.</li> <li>• The event exit is available on every system in the sysplex that could be the target of a restart by automatic restart management.</li> <li>• If IXCARM was called in AR mode: <ul style="list-style-type: none"> <li>– The SYSSTATE ASCENV=AR macro was issued prior to this macro.</li> <li>– If the parameter list address was specified using explicit register notation, the corresponding access register was updated accordingly.</li> <li>– The event exit parameter list is either in the primary address space or in an address/data space that is addressable through a public entry on the caller's DU-AL.</li> </ul> </li> </ul>
08	164	<p><b>Equate Symbol:</b> IXCARMINVEVENTEXIT</p> <p><b>Meaning:</b> Program error. The name specified for the event exit routine is not a valid MVS load module name.</p> <p><b>Action:</b> Make sure the event exit name specified adheres to the rules listed under the EVENTEXIT parameter description.</p>

Table 3. Return and Reason Codes for the IXCARM Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
08	168	<p><b>Equate Symbol:</b> IXCARMINVASYNCREQ</p> <p><b>Meaning:</b> Program error. An IXCARM request requiring asynchronous processing is invalid in the issuer's address space.</p> <p><b>Action:</b> Check your program to ensure that you are issuing the IXCARM request in a valid environment. For example, do not issue IXCARM in an installation restart exit.</p>
08	16C	<p><b>Equate Symbol:</b> IXCARMINVELEMBIND</p> <p><b>Meaning:</b> Program error. The ELEMBIND value specified is not valid, or ELEMBIND=CURSYS was specified with TERMTYPE=ELEMTERM.</p> <p><b>Action:</b> Check your program to ensure that it specified a valid value for ELEMBIND and did not specify ELEMBIND=CURSYS with TERMTYPE=ELEMTERM.</p>
08	1A8	<p><b>Equate Symbol:</b> IXCARMRSVREGFDS</p> <p><b>Meaning:</b> Program error. A REGISTER request contained fields that did not apply to REGISTER and were not zero.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The parameter list address has not been overlaid.</li> <li>• The correct parameter list address was specified.</li> <li>• The parameter list storage area was not inadvertently freed by your program.</li> <li>• If IXCARM was called in AR mode: <ul style="list-style-type: none"> <li>– The SYSSTATE ASCENV=AR macro was issued prior to this macro.</li> <li>– If the parameter list address was specified using explicit register notation, the corresponding access register was updated accordingly.</li> <li>– The parameter list is either in the primary address space or in an address/data space that is addressable through a public entry on the caller's DU-AL.</li> </ul> </li> <li>• Your program was assembled with the correct macro library for the release of z/OS on which your program is running.</li> </ul> <p>Retry the request at least once. If the problem persists, record the return and reason code, and supply them to the appropriate IBM support personnel. Also supply IBM with the IXCARM macro invocation and/or generated parameter list to diagnose the problem.</p>

Table 3. Return and Reason Codes for the IXCARM Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
08	204	<p><b>Equate Symbol:</b> IXCARMBADWAITPRED</p> <p><b>Meaning:</b> Program or environmental error. An IXCARM REQUEST=WAITPRED was received from an element that is already in an Available or Available-To state. An element is placed in an Available state when the element has successfully completed an IXCARM REQUEST=READY request. An element is placed in an Available-To state when the element was restarted by ARM, reregistered, but did not complete an IXCARM REQUEST=READY within the Automatic Restart Manager policy-defined READY_TIMEOUT value.</p> <ul style="list-style-type: none"> <li>• This is a programming error if the element had already successfully completed an IXCARM REQUEST=READY request.</li> <li>• This is an environmental error if the ARM policy did not allot enough time for the element to complete its initialization and an IXCARM REQUEST=READY request.</li> </ul> <p><b>Action:</b> Continue processing. ARM already considers the element to be in an available state. As such, any predecessor elements would be available as well. Note that elements that are at a higher level than your element might think your services are available prematurely.</p> <p>Check your program to determine if it had already issued a READY request. Use the IXCQUERY service to determine the current state of the element (AVAILABLE or AVAILABLE-TO). You can also use the DISPLAY XCF,ARM command for manual debugging. If the element has truly timed out (AVAILABLE-TO), consider changing your logic to continue processing.</p>

Table 3. Return and Reason Codes for the IXCARM Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
08	2A8	<p><b>Equate Symbol:</b> IXCARMRSVWTFDS</p> <p><b>Meaning:</b> Program error. An IXCARM WAITPRED request contained fields not applying to WAITPRED that were not zero.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The parameter list address has not been overlaid.</li> <li>• The correct parameter list address was specified.</li> <li>• The parameter list storage area was not inadvertently freed by your program.</li> <li>• If IXCARM was called in AR mode: <ul style="list-style-type: none"> <li>– The SYSSTATE ASCENV=AR macro was issued prior to this macro.</li> <li>– If the parameter list address was specified using explicit register notation, the corresponding access register was updated accordingly.</li> <li>– The parameter list is either in the primary address space or in an address/data space that is addressable through a public entry on the caller's DU-AL.</li> </ul> </li> <li>• Your program was assembled with the correct macro library for the release of OS/390® on which your program is running.</li> </ul> <p>Retry the request at least once. If the problem persists, record the return and reason code, and supply them to the appropriate IBM support personnel. Also supply IBM with the IXCARM macro invocation and/or generated parameter list to diagnose the problem.</p>
08	304	<p><b>Equate Symbol:</b> IXCARMBADREADY</p> <p><b>Meaning:</b> Program error. An IXCARM REQUEST=READY was received from an element that had already issued an IXCARM REQUEST=READY.</p> <p><b>Action:</b> Check your program to determine why the READY request was issued more than once.</p>



Table 3. Return and Reason Codes for the IXCARM Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
08	3A8	<p><b>Equate Symbol:</b> IXCARMRSVRDYFDS</p> <p><b>Meaning:</b> System error. An IXCARM READY request contained required fields that were not zero.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The parameter list address has not been overlaid.</li> <li>• The correct parameter list address was specified.</li> <li>• The parameter list storage area was not inadvertently freed by your program.</li> <li>• If IXCARM was called in AR mode: <ul style="list-style-type: none"> <li>– The SYSSTATE ASCENV=AR macro was issued prior to this macro.</li> <li>– If the parameter list address was specified using explicit register notation, the corresponding access register was updated accordingly.</li> <li>– The parameter list is either in the primary address space or in an address/data space that is addressable through a public entry on the caller's DU-AL.</li> </ul> </li> <li>• Your program was assembled with the correct macro library for the release of OS/390 on which your program is running.</li> </ul> <p>Retry the request at least once. If the problem persists, record the return and reason code, and supply them to the appropriate IBM support personnel. IBM will also need the IXCARM macro invocation and/or generated parameter list to diagnose the problem.</p>
08	404	<p><b>Equate Symbol:</b> IXCARM DUPASSOC1</p> <p><b>Meaning:</b> Program error. An IXCARM REQUEST=ASSOCIATE was issued by an element that was already associated with an element.</p> <p><b>Action:</b> An element may be associated only with one other element. Check your program to determine why the ASSOCIATE request was issued more than once from this element.</p>
08	408	<p><b>Equate Symbol:</b> IXCARM BADTARGETELEM</p> <p><b>Meaning:</b> Program error. The element specified by the TELEMENT parameter on an IXCARM REQUEST=ASSOCIATE was not a registered element.</p> <p><b>Action:</b> The TELEMENT specified in an ASSOCIATE request must be an element that is already registered with the automatic restart manager. Make sure the element name is correct. You can issue the IXCQUERY macro with the ARMSTATUS parameter for a list of all the elements that have registered with the automatic restart manager.</p>

Table 3. Return and Reason Codes for the IXCARM Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
08	40C	<p><b>Equate Symbol:</b> IXCARM DUPASSOC2</p> <p><b>Meaning:</b> Program error. The element specified by the TELEMENT parameter on an IXCARM REQUEST=ASSOCIATE was already associated with another element.</p> <p><b>Action:</b> An element may be associated only with one other element. Check your program to determine why the ASSOCIATE request was issued more than once for this element.</p>
08	414	<p><b>Equate Symbol:</b> IXCARM SELFASSOC</p> <p><b>Meaning:</b> Program error. The element specified by the TELEMENT parameter in an IXCARM REQUEST=ASSOCIATE is the same element that issued the ASSOCIATE request.</p> <p><b>Action:</b> An element cannot be associated with itself. Correct the TELEMENT value and reissue the request.</p>
08	4A8	<p><b>Equate Symbol:</b> IXCARM RSVASSFDS</p> <p><b>Meaning:</b> System error. An IXCARM ASSOCIATE request contained required fields that were not zero.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The parameter list address has not been overlaid.</li> <li>• The correct parameter list address was specified.</li> <li>• The parameter list storage area was not inadvertently freed by your program.</li> <li>• If IXCARM was called in AR mode: <ul style="list-style-type: none"> <li>– The SYSSTATE ASCENV=AR macro was issued prior to this macro.</li> <li>– If the parameter list address was specified using explicit register notation, the corresponding access register was updated accordingly.</li> <li>– The parameter list is either in the primary address space or in an address/data space that is addressable through a public entry on the caller's DU-AL.</li> </ul> </li> <li>• Your program was assembled with the correct macro library for the release of OS/390 on which your program is running.</li> </ul> <p>Retry the request at least once. If the problem persists, record the return and reason code, and supply them to the appropriate IBM support personnel. IBM will also need the IXCARM invocation and/or generated parameter list to diagnose the problem.</p>

Table 3. Return and Reason Codes for the IXCARM Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
08	5A8	<p><b>Equate Symbol:</b> IXCARMRSVDRGFDS</p> <p><b>Meaning:</b> System error. An IXCARM DEREGISTER request contained required fields that were not zero.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The parameter list address has not been overlaid.</li> <li>• The correct parameter list address was specified.</li> <li>• The parameter list storage area was not inadvertently freed by your program.</li> <li>• If IXCARM was called in AR mode: <ul style="list-style-type: none"> <li>– The SYSSTATE ASCENV=AR macro was issued prior to this macro.</li> <li>– If the parameter list address was specified using explicit register notation, the corresponding access register was updated accordingly.</li> <li>– The parameter list is either in the primary address space or in an address/data space that is addressable through a public entry on the caller's DU-AL.</li> </ul> </li> <li>• Your program was assembled with the correct macro library for the release of OS/390 on which your program is running.</li> </ul> <p>Retry the request at least once. If the problem persists, record the return and reason code, and supply them to the appropriate IBM support personnel. IBM will also need the IXCARM invocation and/or generated parameter list to diagnose the problem.</p>

Table 3. Return and Reason Codes for the IXCARM Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
08	5B0	<p><b>Equate Symbol:</b> IXCARMWRONGELEMNREREG</p> <p><b>Meaning:</b> The element that has attempted to register has done so in an address space that was created for the restart of another element. Only the restarted element can re-register in the current address space. ARM rejects the registration because as part of restarting the related job or started task address space, ARM propagates resources related to the member from the previous instance of the member. Currently the system symbolic substitution table is the only resource that is propagated. Allowing the element to register with ARM might cause an incorrect resource to be used by the element.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• You register with ARM as soon as possible so the condition can be identified.</li> <li>• You use the same ELEMENT name when the application is restarted anywhere in the sysplex. For example, using a system name as part of the element name would not work because the element name then would not match when the element was restarted on another system. If the element name was not correct, re-attempt the IXCARM registration with the correct name.</li> <li>• You terminate your application and inform the operator that your application element was incorrectly started or restarted by a method other than ARM.</li> </ul>

Table 3. Return and Reason Codes for the IXCARM Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
08	5B4	<p><b>Equate Symbol:</b> IXCARMWRONGADDRONREREG</p> <p><b>Meaning:</b> The element that has attempted to re-register has done so in an address space other than the one that was created for the re-registering element. The element can only re-register in the address space in which the override restart start text was issued. ARM rejects the registration because as part of restarting the related job or started task address space, ARM propagates resources related to the member from the previous instance of the member. Currently the system symbolic substitution table is the only resource that is propagated. Allowing the element to register with ARM might cause an incorrect resource to be used by the element.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• You register with ARM as soon as possible so the condition can be identified.</li> <li>• You use the same ELEMENT name when the application is restarted anywhere in the sysplex. For example, using a system name as part of the element name would not work because the element name then would not match when the element was restarted on another system. If the element name was not correct, re-attempt the IXCARM registration with the correct name.</li> <li>• You terminate your application and inform the operator that your application element was incorrectly started or restarted by a method other than ARM.</li> </ul>

Table 3. Return and Reason Codes for the IXCARM Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
08	5B8	<p><b>Equate Symbol:</b> IXCARMREREGAFTEERTIMOUT</p> <p><b>Meaning:</b> The element that has attempted to register has done so in an address space that was created for the restart of another element. However, the element for which the address space was initially created is no longer known to ARM. This is probably due to the restart of the element having timed out. ARM rejects the registration because as part of restarting the related job or started task address space, ARM propagates resources related to the member from the previous instance of the member. Currently the system symbolic substitution table is the only resource that is propagated. Allowing the element to register with ARM might cause an incorrect resource to be used by the element.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• You register with ARM as soon as possible so the condition can be identified.</li> <li>• You use the same ELEMENT name when the application is restarted anywhere in the sysplex. For example, using a system name as part of the element name would not work because the element name then would not match when the element was restarted on another system. If the element name was not correct, re-attempt the IXCARM registration with the correct name.</li> <li>• You terminate your application and inform the operator that your application element was incorrectly started or restarted by a method other than ARM.</li> </ul>
08	5BC	<p><b>Equate Symbol:</b> IXCARMUNAUTHEVENTEXIT</p> <p><b>Meaning:</b> Users who are both in problem state and problem key cannot specify an event exit keyword on registration.</p> <p><b>Action:</b> Remove the EVENTEXIT parameter from the IXCARM invocation and resubmit.</p>
08	5C0	<p><b>Equate Symbol:</b> IXCARMUNAUTHSTARTTXT</p> <p><b>Meaning:</b> Users who are both in problem state and problem key cannot specify a start text keyword on registration.</p> <p><b>Action:</b> Remove the STARTTXT parameter from the IXCARM invocation and resubmit.</p>
08	5C4	<p><b>Equate Symbol:</b> IXCARMUNAUTHRMTOKEN</p> <p><b>Meaning:</b> Program error. Users who are both in problem state and problem key cannot specify the RMTOKEN keyword on any request.</p> <p><b>Action:</b> Remove the RMTOKEN keyword, or switch to supervisor state or system key before using it.</p>

Table 3. Return and Reason Codes for the IXCARM Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0C	04	<p><b>Equate Symbol:</b> IXCARMNOARM</p> <p><b>Meaning:</b> Environmental error. The IXCARM macro was issued on a system that does not support the automatic restart manager.</p> <p><b>Action:</b> Issue the IXCARM macro only on systems that have MVS SP5.2.0 (or higher) and either JES2 SP5.2.0 (or higher), or JES3 SP5.2.1 (or higher), and have access to an ARM couple data set. Consult your system programmer to determine the level of the system you are running on and which systems have connectivity to the ARM couple data set.</p>
0C	0C	<p><b>Equate Symbol:</b> IXCARMNOESTAE</p> <p><b>Meaning:</b> System error. The request is not processed.</p> <p><b>Action:</b> Retry the request at least once. If the problem persists, record the return and reason code, and supply them to the appropriate IBM support personnel.</p>
0C	C0	<p><b>Equate Symbol:</b> IXCARMFDSERR1</p> <p><b>Meaning:</b> Environmental error. Internal error while trying to access the automatic restart manager's couple data set.</p> <p><b>Action:</b> Retry the request at least once. If the problem persists, record the return and reason code, and supply them to the appropriate IBM support personnel.</p>
0C	C4	<p><b>Equate Symbol:</b> IXCARMFDSERR2</p> <p><b>Meaning:</b> System error. Internal error while trying to access the automatic restart manager's couple data set.</p> <p><b>Action:</b> Retry the request at least once. If the problem persists, record the return and reason code, and supply them to the appropriate IBM support personnel.</p>
0C	C8	<p><b>Equate Symbol:</b> IXCARMFDSERR3</p> <p><b>Meaning:</b> System error. Internal error while trying to access the automatic restart manager's couple data set.</p> <p><b>Action:</b> Retry the request at least once. If the problem persists, record the return and reason code, and supply them to the appropriate IBM support personnel.</p>
0C	CC	<p><b>Equate Symbol:</b> IXCARMBADTESTART</p> <p><b>Meaning:</b> System error.</p> <p><b>Action:</b> Retry the request at least once. If the problem persists, record the return and reason code, and supply them to the appropriate IBM support personnel.</p>

Table 3. Return and Reason Codes for the IXCARM Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0C	104	<p><b>Equate Symbol:</b> IXCARMMAXUSERS</p> <p><b>Meaning:</b> Environmental error. The maximum number of automatic restart management users has been reached.</p> <p><b>Action:</b> Retry the request at least once. If the problem persists, consult your system programmer to determine if the maximum number of elements defined in the automatic restart management couple data set should be increased or if the limit on the number of elements has been reached. Your application should cover cases in which the automatic restart management couple data set has no available room. Before running or installing your application, you should make the system programmer aware of your element's requirements.</p>
0C	160	<p><b>Equate Symbol:</b> IXCARMNOCDS</p> <p><b>Meaning:</b> Environmental error. The system on which the IXCARM macro was issued does not have access to an automatic restart management couple data set.</p> <p><b>Action:</b> Either contact the system programmer to determine which systems have a couple data set for the automatic restart manager, or retry the request at a later time. The system will issue ENF signal 38 (ENFPC038 ArenQualifier = ArenEventCdsConnect) when the ARM couple data set becomes available.</p>
0C	164	<p><b>Equate Symbol:</b> IXCARMBADJOB</p> <p><b>Meaning:</b> Environmental error. JES could not support automatic restart manager requests for this job. A unit of work other than a normal batch job or started task has attempted to register with automatic restart manager without specifying ELEMBIND=CURSYS. The registration was rejected.</p> <p><b>Action:</b> Register the element with a bind to the current system.</p>
0C	168	<p><b>Equate Symbol:</b> IXCARMSAFNOTDEFINED</p> <p><b>Meaning:</b> Environmental error. Problem state and problem key users cannot use IXCARM without having a security profile.</p> <p><b>Action:</b> Ensure that the proper IXCARM.elemtype.elemname resource profile for the unauthorized application is defined to RACF or another security product.</p>
0C	16C	<p><b>Equate Symbol:</b> IXCARMNOSAFAUTH</p> <p><b>Meaning:</b> Environmental error. The installed security product indicated that the user does not have authorized access to the IXCARM facility or the secure entity. The entity is made up of the element type and element name.</p> <p><b>Action:</b> Ensure that the unauthorized application has UPDATE access to the IXCARM.elemtype.elemname resource profile in the FACILITY class.</p>



Table 3. Return and Reason Codes for the IXCARM Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
10	04	<p><b>Equate Symbol:</b> IXCARMARMERR</p> <p><b>Meaning:</b> System error. The element was deregistered due to an internal error.</p> <p><b>Action:</b> Retry the request at least once. If the problem persists, record the return and reason code, and supply them to the appropriate IBM support personnel.</p>
10	08	<p><b>Equate Symbol:</b> IXCARMUNKERR</p> <p><b>Meaning:</b> System error. An unexpected error occurred. Automatic restart management will attempt to deregister this element.</p> <p><b>Action:</b> Retry the request at least once. If it fails again, try to deregister the element or issue the IXCQUERY macro to determine the state of this element.</p> <p>If the problem persists, record the return and reason code, and supply them to the appropriate IBM support personnel.</p>
10	A0	<p><b>Equate Symbol:</b> IXCARMPCERROR</p> <p><b>Meaning:</b> System error. An unexpected error occurred. Automatic restart management will attempt to deregister this element.</p> <p><b>Action:</b> Retry the request at least once. If it fails again, try to deregister the element or issue the IXCQUERY macro to determine the state of this element.</p> <p>If the problem persists, record the return and reason code, and supply them to the appropriate IBM support personnel.</p>

## Example

See automatic restart management in [z/OS MVS Programming: Sysplex Services Guide](#) for an example of this macro.



## Chapter 5. IXCCFCM – Coupling Facility Configuration Management

### Description

The IXCCFCM service is for use by a recovery manager to communicate the ACTIVE or INACTIVE states of the recovery manager. IXCCFCM can be used to update the CFRM active policy with the status of the recovery manager (for example, Graphically Dispersed Parallel Sysplex® (GDPS®)) and the name of the recovery site. CFRM will use the SITE information from the CFRM policy, along with the status of the recovery manager and recovery site, to assist in making the appropriate CF duplexing failover decisions.

**Note:** With APAR OA31601, the recovery site provided to XCF through IXCCFCM is ignored. The recovery site does not affect which structure is kept when coupling facility structure duplexing is stopped. IXCQUERY or the DISPLAY XCF command does not provide the recovery site. Because using the recovery site might cause you to lose duplexed coupling facility structure data in the event of a coupling facility failure at the recovery site, disabling use of the recovery site with OA31601 helps you avoid the problem.

### Recovery Manager Concepts

A recovery manager (for example, GDPS) can use the IXCCFCM service to provide recovery status information to CFRM. To implement this solution, the CFRM policy must contain SITE information that has been added by the Administrative Data Utility. The recovery manager uses the IXCCFCM service to inform CFRM of the site being used as the recovery site. When a recovery manager is active, CFRM uses the provided recovery site information along with the site information from the CFRM active policy to assist in making CF structure duplexing failover decisions.

Recovery managers can use the IXCQUERY service to obtain SITE information for a coupling facility.

For additional information about using GDPS as a recovery manager, see [GDPS \(www.ibm.com/systems/z/advantages/gdps\)](http://www.ibm.com/systems/z/advantages/gdps).

If you are using the GDPS/PPRC product, see the following publication for GDPS specific requirements:

- *GDPS/PPRC Installation and Customization Guide, ZG246703*

### Environment

The following are the environment requirements for the caller.

Environment	Environment requirement
Authorization:	Supervisor state or PKM keys 0 - 7
Dispatchable unit mode:	Task
Cross memory mode:	PASN=HASN, any SASN
AMODE:	31-bit
ASC mode:	Primary or access register (AR)
Interrupt status:	Enabled for I/O and external interrupts
Locks:	No locks held, with no enabled, unlocked task (EUT) FRRs established

Environment	Environment requirement
Control parameters:	Must be in the primary address space or be in an address/data space that is addressable through a public entry on the caller's dispatchable unit access list (DU-AL)

## Restrictions

---

None.

## Input Register Information

---

Before issuing the IXCCFCM macro, the caller does not have to place any information into any register unless using it in register notation for a particular parameter, or using it as a base register.

## Output Register Information

---

When control returns to the caller of the IXCCFCM macro, the general purpose registers (GPRs) contain:

### Register

#### Contents

**0**

Reason code if applicable

**1**

Used as work register by the system

**2-13**

Unchanged

**14**

Used as work register by the system

**15**

Return code

When control returns to the caller of the IXCCFCM macro, the access registers (ARs) contain:

### Register

#### Contents

**0-1**

Used as work registers by the system

**2-13**

Unchanged

**14-15**

Used as work registers by the system

**For registers that the system changes,** a caller who depends on these registers containing the same value before and after issuing the macro must save these registers before issuing the macro and restore them after the system returns control.

## Programming Requirements

---

If the program is in AR mode, issue the SYSSTATE ASCENV=AR macro before IXCCFCM. SYSSTATE ASCENV=AR tells the system to generate code appropriate for AR mode.

The use of the IXCCFCM service must be controlled through the z/OS Security Server, which includes RACF, or your installation's security package. The security administrator must define a resource profile for the resource name 'MVSADMIN.XCF.CFRM' in the FACILITY class. This profile should already be defined for using the IXCMIAPU utility on an active CFRM couple data set. UPDATE access authority must be given

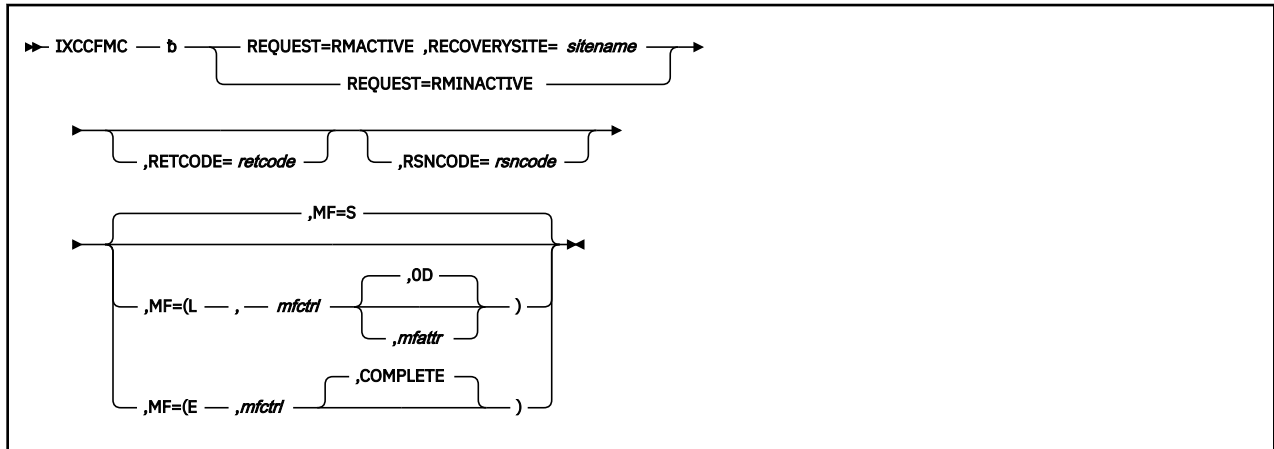
to the recovery manager to allow the recovery manager to use IXCCFCM. Any attempt to use IXCCFCM will fail if the profile is not defined or UPDATE authority is not granted.

## Performance Implications

None.

## Syntax Diagram

The syntax of the IXCCFCM macro is as follows:



## Parameter Descriptions

The parameter descriptions are listed in alphabetical order. Default values are underlined:

**,MF=S**

**,MF=(L,mfctrl)**

**,MF=(L,mfctrl,mfattr)**

**,MF=(L,mfctrl,OD)**

**,MF=(M,mfctrl)**

**,MF=(M,mfctrl,COMPLETE)**

**,MF=(M,mfctrl,NOCHECK)**

**,MF=(E,mfctrl)**

**,MF=(E,mfctrl,COMPLETE)**

**,MF=(E,mfctrl,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

**,mfctrl**

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

**,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

**,COMPLETE****,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

**COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if *SMILE=var* were an optional parameter and the default is *SMILE=NO\_SMILE* then it would not be documented. However, if the default was *SMILE=-*, then it would be documented because a value would be the default.

**NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

**RECOVERYSITE=sitenam**

Use this input parameter to specify the name of the site being used as the recovery site. The name must be 8 characters long, padded on the right with blanks. The name must be 'SITE1 ' or 'SITE2 '.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-byte field that contains the name of the site being used as the recovery site.

**REQUEST=RMACTIVE****REQUEST=RMINACTIVE**

Use this input parameter to specify what information the requestor wants.

**RMACTIVE**

Informs XCF that the recovery manager is currently active. The name of the site being used as the recovery site must be supplied. CFRM will use this information to enhance "duplexing failover" decisions to keep the structure instance at the recovery site for potential site failures.

Use this request to inform XCF that the recovery manager has become active, or to inform XCF that the currently active recovery manager has changed the recovery site.

**RMINACTIVE**

Informs XCF that the recovery manager is currently inactive. CFRM will no longer use the recovery site information when making "duplexing failover" decisions.

**,RETCODE=retcode**

Use this output parameter to specify a field to contain the return code. (The return code is also returned in register 15.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the return code when the request has completed.

**,RSNCODE=rsncode**

Use this output parameter to specify a field to contain the reason code returned, if applicable. (The reason code is also returned in register 0.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the reason code (if any) when the request has completed.

## Return and Reason Codes

---

When the IXCCFCM returns control to your program:

- GPR 15 (and *retcode*, if you coded RETCODE) contains a return code.
- GPR 0 (and *rsncode*, if you coded RSNCODE) contains a reason code, if applicable.

Macro IXCYCON provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

**0**

IXCRETCODEOK

**8**

IXLRETCODEPARMERROR

**C**

IXCRETCODEENVERROR

**10**

IXCRETCODECOMPERROR

The following table identifies the hexadecimal return and reason codes and the equate symbol associated with each reason code.

Table 4. Return and Reason Codes for the IXCCFCM macro		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0	None	<b>Meaning:</b> IXCCFCM request successful. <b>Action:</b> None required.
8	xxxx0801	<b>Equate Symbol:</b> IXCRSNCODEBADPARMLIST <b>Meaning:</b> Program error. The IXCCFCM parameter list is not valid. <b>Action:</b> Ensure that the parameter list is addressable and accessible in your program's key.
8	xxxx0802	<b>Equate Symbol:</b> IXCRSNCODEBADPARMLISTALET <b>Meaning:</b> Program error. The IXCCFRM parameter list ALET is not valid. <b>Action:</b> Ensure that macro usage meets environmental and programming requirements for control parameters.
8	xxxx0804	<b>Equate Symbol:</b> IXCRSNCODEBADVERSIONNUM <b>Meaning:</b> Program error. The level of z/OS on which the caller is running does not support the specified IXCCFCM parameter list version. <b>Action:</b> Reissue the request specifying only keywords that are supported by the level of z/OS on which your program is running.
8	xxxx0806	<b>Equate Symbol:</b> IXCRSNCODENOTTASKMODE <b>Meaning:</b> Program error. The requestor is not in task mode. <b>Action:</b> Ensure that macro usage meets environmental requirements.
8	xxxx0807	<b>Equate Symbol:</b> IXCRSNCODENOTENABLED <b>Meaning:</b> Program error. The caller is not in an enabled state. <b>Action:</b> Ensure that macro usage meets environmental requirements.

Table 4. Return and Reason Codes for the IXCCFCM macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0809	<b>Equate Symbol:</b> IXCRSNCODEPRIMARYNOTHOME <b>Meaning:</b> Program error. Issuer's primary address space is not the home address space. <b>Action:</b> Ensure that macro usage meets environmental requirements.
8	xxxx084C	<b>Equate Symbol:</b> IXCRSNCODEBADNOSAFAUTH <b>Meaning:</b> Program error. The caller does not have the required SAF authorization. <b>Action:</b> Determine why the installation has not allowed the proper SAF authorization for updating the policy.
8	xxxx084D	<b>Equate Symbol:</b> IXCRSNCODELOCKED <b>Meaning:</b> Program error. The caller holds a lock. <b>Action:</b> Ensure that macro usage meets environmental requirements.
8	xxxx0850	<b>Equate Symbol:</b> IXCRSNCODEBADPLISTRSD <b>Meaning:</b> Program error. A reserved field in the parameter list is not valid. <b>Action:</b> Use the IBM-supported IXCCFCM macro interface to build the parameter list and execute the MVS service code. Only specify keywords that are supported on the level of z/OS on which your program is running.
8	xxxx0857	<b>Equate Symbol:</b> IXCRSNCODEFRR <b>Meaning:</b> Program error. The caller has an FRR defined. <b>Action:</b> Ensure that macro usage restrictions are not violated.
8	xxxx088A	<b>Equate Symbol:</b> IXCRSNCODEBADASCMode <b>Meaning:</b> Program error. The ASC mode of the caller is not valid. <b>Action:</b> Ensure that macro usage meets environmental requirements.
8	xxxx088F	<b>Equate Symbol:</b> IXCRSNCODEBADSITE <b>Meaning:</b> Program error. The value of RecoverySite is not valid. Only 'SITE1' and 'SITE2' are valid values. <b>Action:</b> Ensure a valid site name is specified for RecoverySite.
C	xxxx0C29	<b>Equate Symbol:</b> IXCRSNCODENOCFRM <b>Meaning:</b> Environmental error. The CFRM function is not active or is not available. <b>Action:</b> Reissue the request when the CFRM function becomes available.



*Table 4. Return and Reason Codes for the IXCCFCM macro (continued)*

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
10	xxxx10xx	<b>Meaning:</b> Failure in CFRM processing. The state of the CFRM policy is unpredictable. <b>Action:</b> Save the reason code information and contact the IBM support center.



## Chapter 6. IXCCREAT – Define a Member to XCF

### Description

The IXCCREAT macro defines a member to the cross-system coupling facility (XCF). Use it to define the group name and member name, and optionally, place a value in the member's user state field. IXCCREAT also:

- Places the member in the created state.
- Assigns a member token to the new member.
- Activates permanent status recording for the member.
- If active members have a group user-routine, notifies those members about the existence of the created member.

For more information on XCF see *z/OS MVS Programming: Sysplex Services Guide*.

Before the member can use the signalling or monitoring services of XCF, it must issue the IXCJOIN macro with the LASTING=YES parameter. Other active members in the specified group can issue the IXCQUERY macro to access the name, status, and user state field of the created member. They can issue the IXCSETUS macro to change the user state of a created member.

### Environment

The following are the environment requirements for the caller.

Environment	Environment requirement
Minimum authorization:	Supervisor state or PKM allowing key 0-7
Dispatchable unit mode:	Task
Cross memory mode:	Any PASN, any HASN, any SASN
AMODE:	31-bit
ASC mode:	Primary or access register (AR)
Interrupt status:	Enabled for I/O and external interrupts
Locks:	No locks held
Control parameters:	Must be in the primary address space or in an address/data space that is addressable through a public entry on the caller's dispatchable unit access list (DU-AL)

### Programming Requirements

If the program is in AR mode, issue the SYSSTATE ASCENV=AR macro before IXCCREAT. SYSSTATE ASCENV=AR tells the system to generate code appropriate for AR mode.

The IXCYQUAA mapping macro provides the format of the area that the ANSAREA parameter points to. If you intend to use that area, include that mapping macro in your program.

### Restrictions

The caller can have no enabled, unlocked task (EUT) FRRs established.

Do not issue the IXCCREAT macro on a system that is in XCF-local mode. In XCF-local mode, XCF does not support permanent status recording.

## Input Register Information

---

Before issuing the IXCCREAT macro, the caller does not have to place any information into any register unless using it in register notation for a particular parameter, or using it as a base register.

## Output Register Information

---

When control returns to the caller, the general purpose registers (GPRs) contain:

### **Register Contents**

#### **0**

If GPR 15 contains a zero or X'10', GPR 0 is used as a work register by the system; otherwise, GPR 0 contains a reason code.

#### **1**

Used as a work register by the system.

#### **2-13**

Unchanged.

#### **14**

Used as a work register by the system.

#### **15**

Return code.

When control returns to the caller, the access registers (ARs) contain:

### **Register Contents**

#### **0-1**

Used as work registers by the system

#### **2-13**

Unchanged

#### **14-15**

Used as work registers by the system

Some callers depend on register contents remaining the same before and after issuing a service. If the system changes the contents of registers on which the caller depends, the caller must save them before issuing the service, and restore them after the system returns control.

## Performance Implications

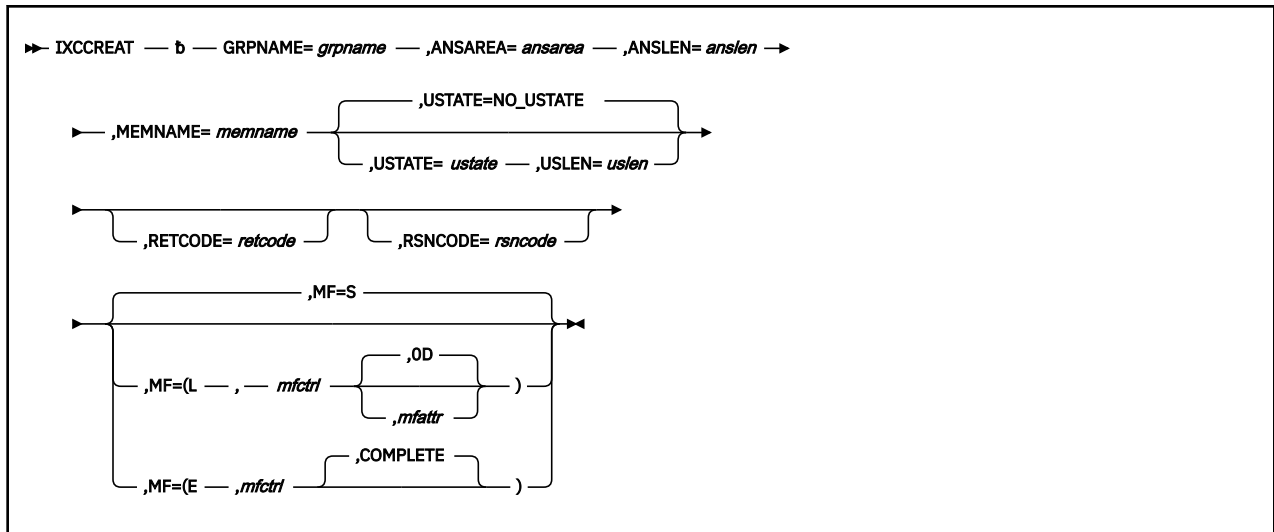
---

None.

## Syntax Diagram

---

The syntax of the IXCCREAT macro is as follows:



## Parameter Descriptions

The parameter descriptions are listed in alphabetical order. Default values are underlined:

### **,ANSAREA=ansarea**

Use this output parameter to specify a storage area to contain information from the request. Member information includes the group name, member name, member token, and any data you place in the user state field specified on USTATE. You will use the member token as input to other XCF macros.

The IXCYQUAA mapping macro provides the format of the member information area. The member information area can be in the primary address space or be addressable through a public entry on the caller's dispatchable unit access list (DU-AL).

Use the ANSLEN parameter to specify the length of the area.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an area (with a length of ANSLEN) where the member information will go.

### **,ANSLEN=anslen**

Use this input parameter to specify the size of the answer area specified by ANSAREA. The size of the area must be greater than or equal to the length of the data area for a single member record returned by IXCQUERY plus the length of the user state field. The field QUAMLEN in the IXCYQUAA mapping macro contains this value (the length of the member record plus the length of the user state field).

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 4-byte field that contains the length of the member information area (ANSAREA).

### **GRPNAME=grpname**

Use this input parameter to specify the name of the group to which the member is to belong. The group name must be eight characters long, padded on the right with blanks if necessary; the valid characters are A-Z, 0-9, and national characters (\$, # and @). To avoid using the names IBM uses for its XCF groups, do not begin group names with the letters A through I or the character string **SYS**. Also, do not use the name UNDESIG, which is reserved for use by the system programmer in your installation.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-byte field that contains the name of the XCF group.

### **,MEMNAME=memname**

Use this input parameter to specify the name you chose for the member. The name must be 16 characters long, padded on the right with blanks if necessary; the valid characters are A-Z, 0-9, and national characters (\$, # and @).

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-byte field that contains the XCF member name.

**,MF=S**  
**,MF=(L,mfctrl)**  
**,MF=(L,mfctrl,mfattr)**  
**,MF=(L,mfctrl,0D)**  
**,MF=(M,mfctrl)**  
**,MF=(M,mfctrl,COMPLETE)**  
**,MF=(M,mfctrl,NOCHECK)**  
**,MF=(E,mfctrl)**  
**,MF=(E,mfctrl,COMPLETE)**  
**,MF=(E,mfctrl,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

#### **,mfctrl**

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

#### **,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

#### **,COMPLETE**

#### **,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

#### **COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if SMILE=*var* were an optional parameter and the default is SMILE=NO\_SMILE then it would not be documented. However, if the default was SMILE=:-), then it would be documented because a value would be the default.

#### **NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

#### **,RETCODE=retcode**

Use this output parameter to specify a field to contain the return code. (The return code is also returned in register 15.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the return code when the request has completed.

**,RSNCODE=*rsncode***

Use this output parameter to specify a field to contain the reason code returned, if applicable. (The reason code is also returned in register 0.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the reason code (if any) when the request has completed.

**,USLEN=*uslen***

Use this input parameter to specify the length in bytes of the user state data that you provide on the USTATE parameter. The length must be from 1 to 32 bytes. If you specify less than 32 bytes, XCF pads the remainder of the user state field, up to 32 bytes, on the right with zeros. IXCQUERY always returns the full 32 bytes, and group user-routines always receive the full 32 bytes. If you code USTATE, USLEN is required.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 4-byte field that contains the length of the user state data (USTATE).

**,USTATE=NO\_USTATE****,USTATE=*ustate***

Use this input parameter to specify the data that you want XCF to place in the user state field associated with the member. Use the USLEN parameter to specify the length of the user state data.

If you do not specify USTATE, or if you specify USTATE=NO\_USTATE, XCF sets the user state field to zeros.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the area (with a length of USLEN) that contains the user state information.

## ABEND Codes

---

None.

## Return and Reason Codes

---

When the IXCCREAT macro returns control to your program:

- GPR 15 (and *retcode*, if you coded RETCODE) contains a return code.
- GPR 0 (and *rsncode*, if you coded RSNCODE) contains a reason code, if applicable.

Macro IXCYCON provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

**0**

IXCRETCODEOK

**4**

IXCRETCODEWARNING

**8**

IXCRETCODEPARMERROR

**C**

IXCRETCODEENVERROR

**10**

IXCRETCODECOMPERROR

The following table identifies the hexadecimal return and reason codes and the equate symbol associated with each reason code.

Table 5. Return and Reason Codes for the IXCCREAT Macro

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
00	None.	<b>Meaning:</b> IXCCREAT completed successfully; XCF places member information in the ANSAREA. <b>Action:</b> None.
04	00	<b>Equate Symbol:</b> IXCCREATRSNFIRSTMEMBER <b>Meaning:</b> This is the first member XCF is defining in the group; XCF places member information in the ANSAREA. <b>Action:</b> None.
08	04	<b>Equate Symbol:</b> IXCCREATRSNALREADYCREATED <b>Meaning:</b> Program error. The member already exists in a created state. <b>Action:</b> None required. However, if your program did not expect this return code, you might want to do one of the following: <ul style="list-style-type: none"> <li>• Choose a different unique name for your new member.</li> <li>• Use the IXCDELET service to delete the current instance of the member, and then reissue the IXCCREAT service to create a new instance of the member. XCF will assign a unique MEMTOKEN to the new instance.</li> <li>• Use the IXCSETUS service to update the member's user state.</li> </ul>
08	08	<b>Equate Symbol:</b> IXCCREATRSNISACTIVE <b>Meaning:</b> Program error. The member already exists in an active state. <b>Action:</b> None required. However, if you did not expect this return code, you might want to do one of the following: <ul style="list-style-type: none"> <li>• Choose a different unique name for your new member.</li> <li>• Use the IXCLEAVE service to remove the current instance of the member, and then reissue the IXCCREAT service to create a new instance of the member. XCF will assign a new unique MEMTOKEN to the new instance.</li> </ul>
08	0C	<b>Equate Symbol:</b> IXCCREATRSNISQUIESCED <b>Meaning:</b> Program error. The member already exists in a quiesced state. <b>Action:</b> None required. However, if you did not expect this return code, you might want to do one of the following: <ul style="list-style-type: none"> <li>• Choose a different unique name for your new member.</li> <li>• Use the IXCDELET service to delete the current instance of the member, and then reissue the IXCCREAT service to create a new instance of the member. XCF will assign a new unique MEMTOKEN to the new instance.</li> </ul>



Table 5. Return and Reason Codes for the IXCCREAT Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
08	10	<p><b>Equate Symbol:</b> IXCCREATRSNISFAILED</p> <p><b>Meaning:</b> Program error. The member already exists in a failed state.</p> <p><b>Action:</b> None required. However, if you did not expect this return code, you might want to do one of the following:</p> <ul style="list-style-type: none"> <li>• Choose a different unique name for your new member.</li> <li>• Use the IXCDELET service to delete the current instance of the member, and then issue the IXCCREAT service again to create a new instance of the member. XCF will assign a new unique MEMTOKEN to the new instance.</li> </ul>
08	14	<p><b>Equate Symbol:</b> IXCCREATRSNGRPNAMEBAD</p> <p><b>Meaning:</b> Program error. The group name is not valid.</p> <p><b>Action:</b> Correct the group name, and retry the request.</p>
08	18	<p><b>Equate Symbol:</b> IXCCREATRSNMEMNAMEBAD</p> <p><b>Meaning:</b> Program error. The member name is not valid.</p> <p><b>Action:</b> Correct the member name, and retry the request.</p>
08	3C	<p><b>Equate Symbol:</b> IXCCREATRSNANSAREAINCOMPLETE</p> <p><b>Meaning:</b> Program error. The caller specified ANSAREA or ANSLEN incorrectly. Even though the member might have been placed in a created state, the ANSAREA was nonaddressable or the ANSLEN was not large enough.</p> <p><b>Action:</b> Check the two high-order bytes of this reason code fullword xxyy003C for the return code xx (either 00 or 04) and reason code yy the caller would have received if the caller had coded those parameters correctly. Take action as described by the corresponding return and reason code. You might want to abnormally end your program or take some other action that will record the problem. You should correct your program to ensure that the ANSAREA is addressable and that the ANSLEN is large enough. See the description of these keywords for their requirements.</p> <p>If the return and reason code indicate that the member was placed in a created state, you can use the IXCQUERY service to get the information that would have been returned in the ANSAREA.</p> <p><b>Note:</b> You should specify a member name or a unique user state value on the IXCCREAT invocation. If you request to have XCF define the member name, you might not be able to use the IXCQUERY macro to determine which member was created on your behalf.</p>

Table 5. Return and Reason Codes for the IXCCREAT Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
08	40	<p><b>Equate Symbol:</b> IXCCREATRSNPLISTRSDNOTVALID</p> <p><b>Meaning:</b> Program error or environmental error. A reserved field in the control parameter list is not zero.</p> <p><b>Action:</b> Check to see if your program inadvertently overlaid the parameter list storage, and that it was assembled with the correct macro library for the release of MVS your program is running on.</p>
08	100	<p><b>Equate Symbol:</b> IXCCREATRSNPLISTBADALET</p> <p><b>Meaning:</b> Program error. Your program is not running in primary ASC mode, and the ALET that qualifies the address of the control parameter list is neither zero nor associated with a valid public entry on the caller's DU-AL.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• Your program was not intended to run in primary ASC mode.</li> <li>• You specified SYSSTATE ASCENV=AR prior to issuing the IXCCREAT macro.</li> <li>• The ALET for the parameter list is a valid public entry on the DU-AL or zero (primary address space ALET).</li> </ul>
08	104	<p><b>Equate Symbol:</b> IXCCREATRSNPLISTVERSIONNOTVALID</p> <p><b>Meaning:</b> Program error or environmental error. The version number in the control parameter list is not valid.</p> <p><b>Action:</b> Check to see if your program inadvertently overlaid the parameter list storage, and that it was assembled with the correct macro library for the release of MVS your program is running on.</p>
08	108	<p><b>Equate Symbol:</b> IXCCREATRSNPLISTBADFUNCTION</p> <p><b>Meaning:</b> Program error or environmental error. The function code in the control parameter list is not valid.</p> <p><b>Action:</b> Check to see if your program inadvertently overlaid the parameter list storage, and that it was assembled with the correct macro library for the release of MVS your program is running on.</p>

Table 5. Return and Reason Codes for the IXCCREAT Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
08	10C	<p><b>Equate Symbol:</b> IXCCREATRSNPLISTBADSTG</p> <p><b>Meaning:</b> Program error. XCF could not access the control parameter list.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• Ensure that the correct parameter list storage area was specified.</li> <li>• If your program is running in AR mode: <ul style="list-style-type: none"> <li>– Ensure that you specified SYSSTATE ASCENV=AR before issuing the IXCCREAT macro.</li> <li>– Ensure that the parameter list ALET is correct.</li> </ul> </li> <li>• Ensure that the parameter list storage area was not inadvertently freed by your program.</li> </ul>
08	110	<p><b>Equate Symbol:</b> IXCCREATRSNUSTATEBADSTG</p> <p><b>Meaning:</b> Program error. XCF could not access the USTATE value.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• Ensure that the correct USTATE address was specified.</li> <li>• If your program is running in AR mode: <ul style="list-style-type: none"> <li>– Ensure that you specified SYSSTATE ASCENV=AR before issuing the IXCCREAT macro.</li> <li>– Ensure that the USTATE ALET is correct.</li> </ul> </li> <li>• Ensure that the USTATE storage area was not inadvertently freed by your program.</li> </ul>
08	114	<p><b>Equate Symbol:</b> IXCCREATRSNUSLENBADVALUE</p> <p><b>Meaning:</b> Program error. The USLEN value is less than 1 or greater than 32.</p> <p><b>Action:</b> Correct the USLEN, and retry the request.</p>
08	118	<p><b>Equate Symbol:</b> IXCCREATRSNNOTTASKMODE</p> <p><b>Meaning:</b> Program error. The caller is not in task mode.</p> <p><b>Action:</b> Correct your program so that it issues IXCCREAT only while in task mode.</p>
08	11C	<p><b>Equate Symbol:</b> IXCCREATRSNNOTENABLED</p> <p><b>Meaning:</b> Program error. The caller is not enabled.</p> <p><b>Action:</b> Correct your program so that it issues IXCCREAT only while enabled.</p>

Table 5. Return and Reason Codes for the IXCCREAT Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0C	04	<p><b>Equate Symbol:</b> IXCCREATRSNMAXGROUPS</p> <p><b>Meaning:</b> Environmental error. Your request would have created the first member in a new XCF group. The new group could not be created because the maximum number of groups as defined by the current couple data set already exists.</p> <p><b>Action:</b> Retry the request at least once. If the problem persists, consult your system programmer to determine if the number of groups defined in the couple data set should be increased, or if the XCF limit on the number of groups has been reached. Your application should cover cases in which the couple data set has no available room. Before running or installing your application, you should make the system programmer aware of your XCF group and member resource requirements.</p>
0C	08	<p><b>Equate Symbol:</b> IXCCREATRSNMAXMEMBERS</p> <p><b>Meaning:</b> Environmental error. The maximum number of members in the group already exists. The maximum number of members per group is defined by the installation when the couple data set is formatted.</p> <p><b>Action:</b> Retry the request at least once. If the problem persists, consult your system programmer to determine if the maximum number of members defined in the couple data set should be increased, or if the XCF limit on the number of members has been reached. Your application should cover cases in which no room is available within the group. Before running or installing your application, you should make the system programmer aware of your XCF group and member resource requirements.</p>
0C	10	<p><b>Equate Symbol:</b> IXCCREATRSNPARTITIONING</p> <p><b>Meaning:</b> Environmental error. The system is being removed from the sysplex, and all IXCCREAT requests are permanently suspended.</p> <p><b>Action:</b> The action required depends on your application. You might want to prepare your application for system termination. (See “Using the Cross-System Coupling Facility” in <i>z/OS MVS Programming: Sysplex Services Guide</i>.)</p>

Table 5. Return and Reason Codes for the IXCCREAT Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0C	14	<b>Equate Symbol:</b> IXCCREATRSNXCFLOCALMODE <b>Meaning:</b> Environmental error. IXCCREAT is not allowed when the system is in XCF-local mode. By default, a member in a created state has permanent status recording. Permanent status recording requires a couple data set in which XCF can retain the member's status. XCF-local mode does not support a couple data set. <b>Action:</b> The action required depends on your application. If XCF-local mode is not supported by your application, you must make the system programmer aware of this requirement. The system programmer will have to ensure that the system is not IPLed in XCF-local mode. Before running or installing your application, you should make the system programmer aware of your XCF group and member resource requirements. See <i>z/OS MVS Programming: Sysplex Services Guide</i> .
0C	18	<b>Equate Symbol:</b> IXCCREATRSNTASKABENDED <b>Meaning:</b> Environmental error. While the issuing task was suspended for XCF processing, the task was abended; that is, another unit of work attempted to abnormally terminate this task. The state of the IXCCREAT request is unpredictable. <b>Action:</b> Determine why this task was being abended.
10	None	<b>Meaning:</b> System error. XCF processing failed. <b>Action:</b> Retry the request at least once. If the problem persists, record the return code, and supply it to the appropriate IBM support personnel.

## Example

Create a member MEMB2 in a group MYGROUP with a user state of X'11'. MYAREA points to the area where XCF is to return member information. XCF is to store the return code and reason code into the variables RETURN and REASON. The code is as follows:

```

IXCCREAT GRPNAME=MYGROUP,ANSAREA=MYAREA,ANSLEN=AREALEN,      X
          MEMNAME=MEMB2,USTATE=STATE,USLEN=LEN,              X
          RETCODE=RETURN,RSNCODE=REASON,MF=S

RETURN    DS      1F      RETURN CODE
REASON    DS      1F      REASON CODE
MYAREA    DS      CL124    OUTPUT AREA TO CONTAIN DATA      X
                               RETURNED BY IXCCREAT
MYGROUP   DC      CL8'MYGROUP '  GROUP NAME
MEMB2     DC      CL16'MEMB2    '  MEMBER NAME
STATE     DC      X'11'        USER STATE VALUE FOR THIS      X
                               MEMBER
LEN        DC      F'1'        LENGTH OF USER STATE FIELD
AREALEN    DC      F'124'      LENGTH OF OUTPUT AREA

```



## Chapter 7. IXCDELET – Change an XCF Member's State to Not-Defined

### Description

The IXCDELET macro allows an authorized routine to change the state of a failed, quiesced, or created cross-system coupling facility (XCF) member to the not-defined state. XCF does not recognize the existence of a member in the not-defined state. Therefore, all future requests against this member will generate a return code indicating that the member no longer exists. For created members, XCF notifies the active members of the group, through their group user-routines, about the change in the member's state.

### Environment

The following are the environment requirements for the caller.

Environment	Environment requirement
Minimum authorization:	Supervisor state or PKM allowing key 0-7
Dispatchable unit mode:	Task
Cross memory mode:	Any PASN, any HASN, any SASN
AMODE:	31-bit
ASC mode:	Primary or access register (AR)
Interrupt status:	Enabled or disabled for I/O and external interrupts
Locks:	No locks held
Control parameters:	Must be in the primary address space or in an address/data space that is addressable through a public entry in the caller's dispatchable unit access list (DU-AL)

### Programming Requirements

If the program is in AR mode, issue SYSSTATE ASCENV=AR before IXCDELET. SYSSTATE ASCENV=AR tells the system to generate code appropriate for AR mode.

### Restrictions

The caller can have no enabled, unlocked task (EUT) FRRs established.

### Input Register Information

Before issuing the IXCDELET macro, the caller does not have to place any information into any register unless using it in register notation for a particular parameter, or using it as a base register.

### Output Register Information

When control returns to the caller, the general purpose registers (GPRs) contain:

**Register  
Contents**

- 0**  
If GPR 15 contains a zero or X'10', GPR 0 is used as a work register by the system; otherwise, GPR 0 contains a reason code.
- 1**  
Used as a work register by the system.
- 2-13**  
Unchanged.
- 14**  
Used as a work register by the system.
- 15**  
Return code.

When control returns to the caller, the access registers (ARs) contain:

**Register  
Contents**

- 0-1**  
Used as work registers by the system
- 2-13**  
Unchanged
- 14-15**  
Used as work registers by the system

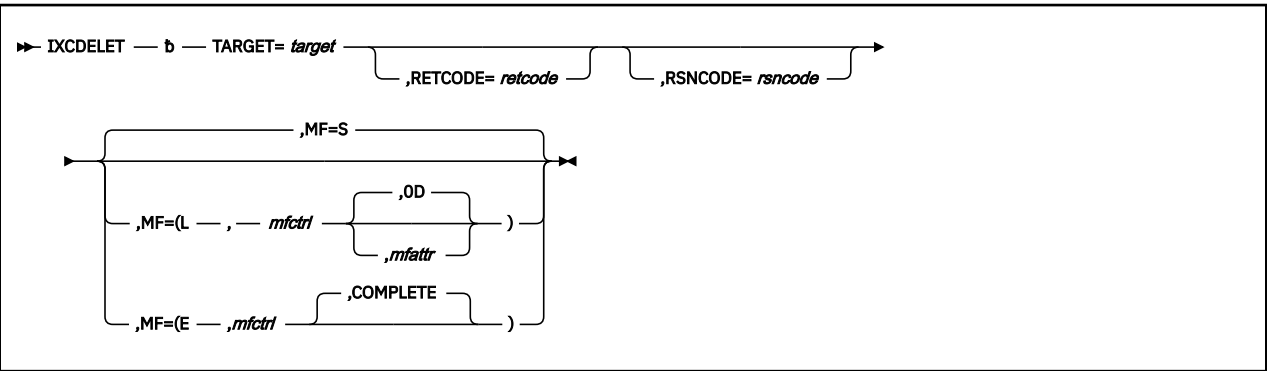
Some callers depend on register contents remaining the same before and after issuing a service. If the system changes the contents of registers on which the caller depends, the caller must save them before issuing the service, and restore them after the system returns control.

Performance Implications

None.

Syntax Diagram

The syntax of the IXCDELET macro is as follows:



Parameter Descriptions

The parameter descriptions are listed in alphabetical order. Default values are underlined:



**,MF=S**  
**,MF=(L,mfctrl)**  
**,MF=(L,mfctrl,mfattr)**  
**,MF=(L,mfctrl,0D)**  
**,MF=(M,mfctrl)**  
**,MF=(M,mfctrl,COMPLETE)**  
**,MF=(M,mfctrl,NOCHECK)**  
**,MF=(E,mfctrl)**  
**,MF=(E,mfctrl,COMPLETE)**  
**,MF=(E,mfctrl,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

#### **,mfctrl**

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

#### **,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

#### **,COMPLETE**

#### **,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

#### **COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if SMILE=*var* were an optional parameter and the default is SMILE=NO\_SMILE then it would not be documented. However, if the default was SMILE=:), then it would be documented because a value would be the default.

#### **NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

#### **,RETCODE=retcode**

Use this output parameter to specify a field to contain the return code. (The return code is also returned in register 15.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the return code when the request has completed.

**,RSNCODE=rsncode**

Use this output parameter to specify a field to contain the reason code returned, if applicable. (The reason code is also returned in register 0.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the reason code (if any) when the request has completed.

**TARGET=target**

Use this input parameter to specify the 64-bit token of the created, quiesced, or failed member that XCF is to place in the not-defined state. IXCJOIN (or IXCCREAT) provided this token when it activated (or created) the member.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-byte field that contains the token.

## ABEND Codes

---

None.

## Return and Reason Codes

---

When the IXCDELET macro returns control to your program:

- GPR 15 (and *retcode*, if you coded RETCODE) contains a return code.
- GPR 0 (and *rsncode*, if you coded RSNCODE) contains a reason code, if applicable.

Macro IXCYCON provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

**0**

IXCRETCODEOK

**4**

IXCRETCODEWARNING

**8**

IXCRETCODEPARMERROR

**C**

IXCRETCODEENVERROR

**10**

IXCRETCODECOMPERROR

The following table identifies the hexadecimal return and reason codes and the equate symbol associated with each reason code.

Table 6. Return and Reason Codes for the IXCDELET Macro		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
00	None.	<b>Meaning:</b> IXCDELET completed successfully; XCF places the member in the not-defined state. <b>Action:</b> None.
08	04	<b>Equate Symbol:</b> IXCDELETRSNNOTDEFINED <b>Meaning:</b> Program error. The target member does not exist. <b>Action:</b> None required. However, if the target member is expected to exist, make sure TARGET MEMTOKEN is correct, and retry the request.

Table 6. Return and Reason Codes for the IXCDELET Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
08	08	<p><b>Equate Symbol:</b> IXCDELETRSNINAPPROPRIATESTATE</p> <p><b>Meaning:</b> Program error. The target member is not in a created, failed, or quiesced state.</p> <p><b>Action:</b> The action required depends on your application. If the member is expected to be in a created, failed, or quiesced state, ensure that the TARGET MEMTOKEN is correct, and retry the request.</p> <p>If the target member can be in another state, use the IXCQUERY service to determine what state the member is in. If the member is not defined to the XCF group, no action is required. If the member is in an active state and should be placed in a not-defined state, you can use the IXCLEAVE or IXCTERM service to place the member in a not-defined state.</p>
08	40	<p><b>Equate Symbol:</b> IXCDELETRSNPLISTRSDNOTVALID</p> <p><b>Meaning:</b> Program error or environmental error. A reserved field in the control parameter list is not zero.</p> <p><b>Action:</b> Check to see if your program inadvertently overlaid the parameter list storage, and that it was assembled with the correct macro library for the release of MVS your program is running on.</p>
08	100	<p><b>Equate Symbol:</b> IXCDELETRSNPLISTBADALET</p> <p><b>Meaning:</b> Program error. Your program is not running in primary ASC mode, and the ALET that qualifies the address of the control parameter list is neither zero nor associated with a valid public entry on the caller's DU-AL.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• Your program is not intended to run in primary ASC mode.</li> <li>• You specified SYSSTATE ASCENV=AR before issuing the IXCDELET macro.</li> <li>• The ALET for the parameter list is on the DU-AL or is zero (primary address space ALET).</li> </ul>
08	104	<p><b>Equate Symbol:</b> IXCDELETRSNPLISTVERSIONNOTVALID</p> <p><b>Meaning:</b> Program error. The version number in the control parameter list is not valid.</p> <p><b>Action:</b> Check to see if your program inadvertently overlaid the parameter list storage.</p>
08	108	<p><b>Equate Symbol:</b> IXCDELETRSNPLISTBADFUNCTION</p> <p><b>Meaning:</b> Program error. The function code in the control parameter list is not valid.</p> <p><b>Action:</b> Check to see if your program inadvertently overlaid the parameter list storage.</p>

Table 6. Return and Reason Codes for the IXCDELET Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
08	10C	<b>Equate Symbol:</b> IXCDELETRSNNPLISTBADSTG <b>Meaning:</b> Program error. XCF could not access the control parameter list. <b>Action:</b> Take one or more of the following actions: <ul style="list-style-type: none"> <li>• Ensure that the correct parameter list storage area was specified.</li> <li>• If your program is running in AR ASC mode:               <ul style="list-style-type: none"> <li>– Ensure that you specified SYSSTATE ASCENV=AR before issuing the IXCDELET macro.</li> <li>– Ensure that the parameter list ALET is correct.</li> </ul> </li> <li>• Ensure that the parameter list storage area was not inadvertently freed by your program.</li> </ul>
08	118	<b>Equate Symbol:</b> IXCDELETRSNNOTTASKMODE <b>Meaning:</b> Program error. The caller is not in task mode. <b>Action:</b> Correct your program so that it issues IXCDELET only while in task mode.
08	11C	<b>Equate Symbol:</b> IXCDELETRSNNOTENABLED <b>Meaning:</b> Program error. The caller is not enabled. <b>Action:</b> Correct your program so that it issues IXCDELET only while enabled.
0C	18	<b>Equate Symbol:</b> IXCDELETRSNTASKABENDED <b>Meaning:</b> Environmental error. While the issuing task was suspended for XCF processing, the task was abended (i.e. another unit of work attempted to abnormally terminate this task). The state of the IXCDELET request is unpredictable. <b>Action:</b> Determine why this task was being abended.
10	None	<b>Meaning:</b> System error. XCF processing failed. <b>Action:</b> Retry the request at least once. If the problem persists, record the return code, and supply it to the appropriate IBM support personnel.

## Example

Delete a member from a group. XCF is to store the return code and reason code into the variables RETURN and REASON. The code is as follows:

```

IXCDELET TARGET=TOKEN3, RETCODE=RETURN, RSNCODE=REASON, MF=S

TOKEN3  DS    CL8                TOKEN OF MEMBER TO BE PLACED    X
      IN NOT-DEFINED STATE
RETURN  DS    1F                RETURN CODE
REASON  DS    1F                REASON CODE

```

You can obtain the member token from the QUAMTKN field in the area returned by IXCCREAT, IXCJOIN, or IXCQUERY, and mapped by the IXCYQUAA mapping macro.



## Chapter 8. IXCJOIN – Place an XCF Member in the Active State

### Description

The IXCJOIN macro places a cross-system coupling facility (XCF) member in the active state, associating it with an XCF group. In the active state, the member can use the monitoring and signalling services of XCF. Additionally, IXCJOIN:

- Returns a member token. (Each member, regardless of its previous state, receives a **new** member token.)
- Notifies the active group members that have a group user-routine that the joining member is in an active state.

The member might previously have been in the failed, quiesced, not-defined, or created state.

The required parameters ANSAREA and ANSLLEN specify the area where XCF returns member information, including a member token.

With the optional parameters USTATE and USLEN, you can place a value in the user state field. The data in the user state field is provided to the group user-routines.

Optional parameters on IXCJOIN allow you to:

- Associate the joining member with a task, job step task, or address space (MEMASSOC).
- Establish permanent status recording (LASTING=YES).
- Specify the address of the message user-routine (MSGEXIT).
- Specify the address of the group user-routine (GRPEXIT).
- Specify the address of the status user-routine (STATEXIT).
- Specify whether cleanup will be performed when a system leaves the sysplex (SYSCLEANUPMEM).

Additional optional parameters on version 1 of IXCJOIN allow you to:

- Specify whether the member can participate in a message response collection protocol (CANREPLY).
- Specify the address of the message notify user routine (NOTIFYEXIT).
- Specify whether the member supports sending or receiving messages greater than 61K bytes in length (up to 128M bytes) (GT61KMSG).

Status monitoring requires a status field (STATFLD) and an interval value (INTERVAL) that sets the status-checking interval. The interval determines how long the status field can remain unchanged before XCF schedules the status user-routine.

For information about the user-routines, see the chapter on XCF in [z/OS MVS Programming: Sysplex Services Guide](#).

### Environment

The following are the environment requirements for the caller.

Environment	Environment requirement
Minimum authorization:	Supervisor state or PKM allowing key 0 - 7
Dispatchable unit mode:	Task
Cross memory mode:	PASN=HASN, any HASN, any SASN

Environment	Environment requirement
AMODE:	31-bit
ASC mode:	Primary or access register (AR)
Interrupt status:	Enabled for I/O and external interrupts
Locks:	No locks held
Control parameters:	Must be in the primary address space or be in an address/data space that is addressable through a public entry in the caller's dispatchable unit access list (DU-AL)

## Programming Requirements

If the program is in AR mode, issue the SYSSTATE ASCENV=AR macro before IXCJOIN. SYSSTATE ASCENV=AR tells the system to generate code appropriate for AR mode.

The IXCYQUAA mapping macro provides the format of the area that the ANSAREA parameter points to (the QUAMEM section maps the member information). If you intend to use that area, include the IXCYQUAA mapping macro in your program.

## Restrictions

The caller can have no enabled, unlocked task (EUT) FRRs established.

## Input Register Information

Before issuing the IXCJOIN macro, the caller does not have to place any information into any register unless using it in register notation for a particular parameter, or using it as a base register.

## Output Register Information

When control returns to the caller, the general purpose registers (GPRs) contain:

### Register

#### Contents

**0**

Reason code, if GPR 15 contains a non-zero return code that has applicable reason codes.

**1**

Used as a work register by the system.

**2-13**

Unchanged.

**14**

Used as a work register by the system.

**15**

Return code.

When control returns to the caller, the access registers (ARs) contain:

### Register

#### Contents

**0-1**

Used as work registers by the system

**2-13**

Unchanged



**14-15**

Used as work registers by the system

Some callers depend on register contents remaining the same before and after issuing a service. If the system changes the contents of registers on which the caller depends, the caller must save them before issuing the service, and restore them after the system returns control.

## Performance Implications

---

None.

## Understanding IXCJOIN Version Support

---

The IXCJOIN macro supports versions 0, 1 and 2.

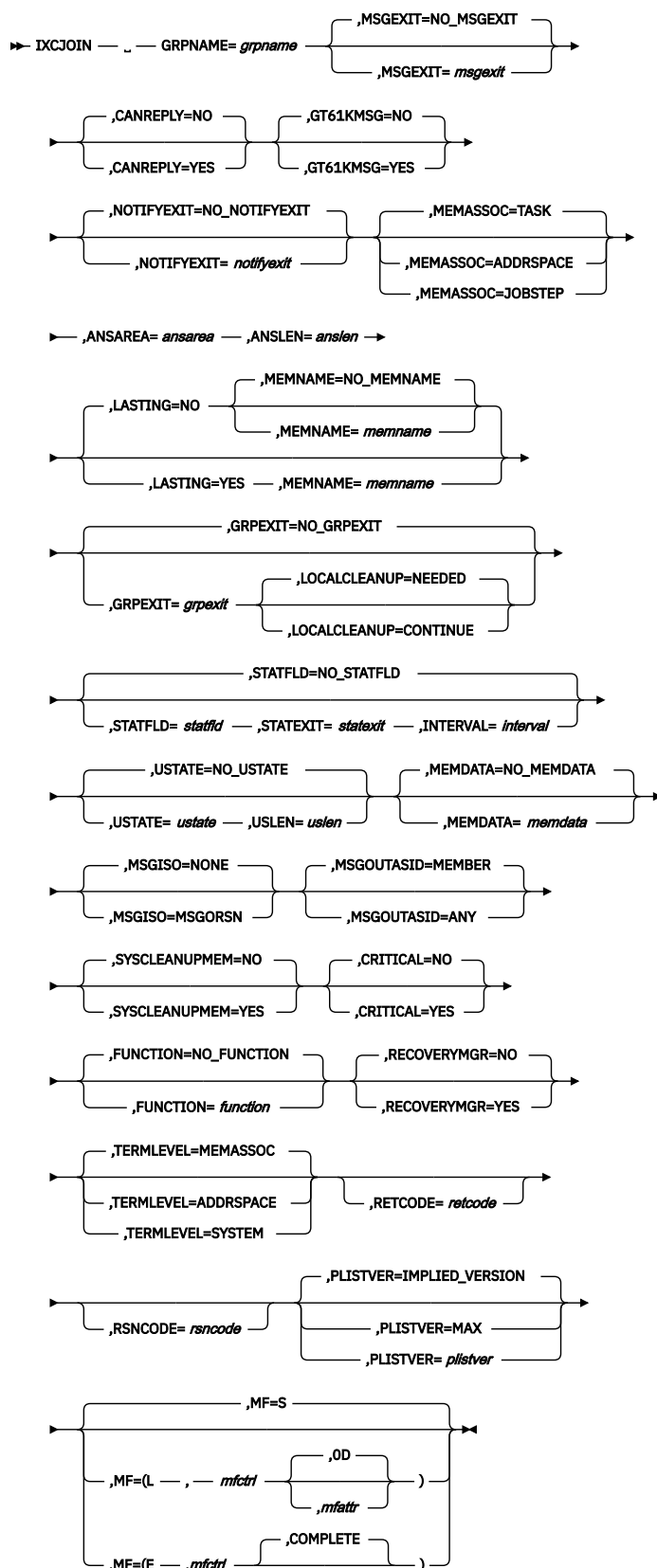
- Keywords not specifically noted here are supported by version 0 and subsequent versions of the IXCJOIN macro.
- The following keywords are supported by version 1 and subsequent versions of the IXCJOIN macro.
  - CANREPLY
  - GT61KMSG
  - NOTIFYEXIT
- The following keywords are supported by version 2 and subsequent versions of the IXCJOIN macro.
  - CRITICAL
  - FUNCTION
  - MSGISO
  - RECOVERYMGR
  - TERMLEVEL

Specify the version of the parameter list that you want generated with the PLISTVER keyword. See Chapter 2, [“Specifying a Macro Version Number,”](#) on page 5 for considerations when specifying the version of the parameter list with PLISTVER.

## Syntax Diagram

---

The syntax of the IXCJOIN macro is as follows:



## Parameter Descriptions

The parameter descriptions are listed in alphabetical order. Default values are underlined:

**,ANSAREA=ansarea**

Use this output parameter to specify a storage area to contain member information returned from the request. Member information includes the group name, member name, member token, and any data you place in the user state field specified on USTATE. You will use the member token as input to other XCF macros.

The IXCYQUAA mapping macro, in the QUAMEM section, provides the format of the member information area. The member information area can be in the primary address space or be addressable through a public entry on the caller's dispatchable unit access list (DU-AL).

Use the ANSLEN parameter to specify the length of the area.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an area (with a length of ANSLEN) where the system will put the member information.

**,ANSLEN=anslen**

Use this input parameter to specify the size of the answer area specified by ANSAREA. The size of the area must be greater than or equal to the length of the data area for a single member record returned by IXCQUERY plus the length of the user state field. The field QUAMLEN in the IXCYQUAA mapping macro contains this value (the length of the member record plus the length of the user state field).

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 4-byte field that contains the length of the member information area (ANSAREA).

**,CANREPLY=NO****,CANREPLY=YES**

Use this input parameter to specify whether the member can participate in the protocol for XCF-managed response collection for response-required signals.

**NO**

The member does not participate in XCF-managed response collection. A response-required message will be sent to the member even if it does not participate in the XCF-managed response collection protocol. XCF does not expect the member to recognize or respond to a message that requires an XCF-managed response. XCF response management does not wait for responses from this member.

**YES**

The member participates in XCF-managed response collection. The member can recognize and respond to a message that requires an XCF-managed response. The member detects when an XCF-managed response is required and sends the response with the IXCMSGO SENDTO=ORIGINATOR service.

The message exit parameter list, mapped by IXCYMEPL, the message notification exit parameter list, mapped by IXCYMNPL, or data, mapped by IXCYMQAA, returned by the Message Control Query Message service (IXCMSGC REQUEST=QUERYMSG) provide the information needed to allow the member to participate in this protocol.

**,CRITICAL=NO****,CRITICAL=YES**

Use this input parameter to indicate whether the member is a critical member. A critical member is one whose function is so critical to the normal operation of the group (and probably the system) that the member should be terminated if it appears to be impaired.

**NO**

The member does not designate itself as a critical member.

**YES**

The member designates itself as a critical member. XCF will monitor the health of the member. If the member becomes impaired for too long, XCF will terminate the member according to the TERMLEVEL specification.

To determine whether the member is impaired, XCF can use two techniques to monitor the health of a critical member:

- XCF can use the status supplied by the member.

- XCF can monitor the member's ability to process its XCF related work.

If the member requests status monitoring by coding the STATFLD, STATEXIT, and INTERVAL keywords, XCF will use both monitoring methods. If the member does not request status monitoring at join time, XCF will use the second method only.

**,FUNCTION=NO FUNCTION**

**,FUNCTION=function**

Use this input parameter to describe the function, service, or application associated with the member. This description will appear in various XCF messages that provide information about the member.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 24-character input field that contains the description of the function, service, or application associated with the member. The string can contain any alphanumeric (A-Z), national (@,#,\$), or special character (underscore or blank). Leading blanks and all blank descriptors are not permitted.

**Note:** The FUNCTION keyword is required if the IXCJOIN macro uses a parameter list of version 2 or higher.

**,GRPEXIT=NO GRPEXIT**

**,GRPEXIT=grpexit**

Use this input parameter to specify the address of the optional group user-routine. This routine executes in the primary address space of the issuer of the IXCJOIN macro, in 31-bit addressing mode and in SRB mode. The routine receives control when there is a change in the operational state of other members in the group or systems in the sysplex.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 4-byte field that contains the address of the group user-routine.

**GRPNAME=grpname**

Use this input parameter to specify the name you assign to the group that the member joins. The group name must be eight characters long, padded on the right with blanks if necessary; the valid characters are A-Z, 0-9, and national characters (\$, # and @). To avoid using the names IBM uses for its XCF groups, do not begin group names with the letters A through I or the character string **SYS**. Also, do not use the name UNDESIG, which is reserved for use by the system programmer in your installation.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-byte field that contains the name of the XCF group.

**,GT61KMSG=NO**

**,GT61KMSG=YES**

Use this input parameter to specify whether the member supports sending or receiving messages greater than 61K bytes in length (62,464 bytes). If so, the member is allowed to use the XCF Message Services for messages up to 128M bytes in length (134,217,728 bytes).

Data returned by the IXCQUERY service indicates whether a member is permitted to send or receive messages greater than 61K bytes.

**NO**

Messages for this member are limited to a maximum length of 61K bytes. XCF will not allow the member to use the XCF Message-out Service (IXCMSGO) to send messages longer than 61K bytes. Messages sent to this member will be rejected by the XCF Message-out Service if they exceed 61K bytes.

**YES**

Messages for this member are limited to a maximum length of 128M bytes. XCF will not allow the member to use the XCF Message-out Service (IXCMSGO) to send messages longer than 128M bytes. Messages sent to this member will be rejected by the XCF Message-out Service if they exceed 128M bytes. The member is capable of using the XCF Message-in Service (IXCMSGI) to receive messages of up to a maximum of 128M bytes, although the application might impose a lower message length.

The message exit parameter list, mapped by IXCYMEPL, the message notification exit parameter list, mapped by IXCYMNPL, or data, mapped by IXCYMQAA, returned by the Message Control Query Message service (IXCMMSGC REQUEST=QUERYMSG) provide the information needed to allow the member to participate in this protocol.

#### **,INTERVAL=*interval***

Use this input parameter to specify the status-checking interval, in hundredths of seconds, that determines the length of time that can elapse with no change to the status field before scheduling the user status routine. Specify the interval in full seconds; that is, the value must be greater than zero and must be a multiple of 100. If you specify INTERVAL, you must also specify STATEXIT and STATFLD.

**Note:** The user status routine might not be called immediately after the interval expires based on timing of the status check as well as system environmental conditions.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 4-byte field that contains the interval value in hundredths of seconds.

#### **,LASTING=NO**

#### **,LASTING=YES**

Use this input parameter to specify whether the member is to have permanent status recording. If you code LASTING=NO, XCF recognizes the member only while the member is active. If the sysplex is in XCF-local mode, you must use LASTING=NO.

If you code LASTING=YES, XCF recognizes the member even while the member is in the failed or quiesced state. If you are activating a member that already has permanent status recording in effect, use LASTING=YES.

MEMNAME is required with LASTING=YES.

#### **,LOCALCLEANUP=NEEDED**

#### **,LOCALCLEANUP=CONTINUE**

Use this input parameter to indicate whether the member needs time to perform cleanup processing when the system on which the member resides is being removed from the sysplex. XCF calls the member group user exit routine with the "system being removed from sysplex" event (GEPLTYPE=GESYSPT) to notify the member that its system is being removed from the sysplex.

#### **NEEDED**

The member needs to perform cleanup when the system on which it resides is being removed from the sysplex. XCF will wait no longer than the installation-defined CLEANUP interval for the member to accomplish the cleanup before putting the system in a wait state. The member must inform XCF when the cleanup is completed by invoking either the IXCLEAVE or IXCQUIES macro. System termination will continue when the CLEANUP interval expires, or when all members that need to do cleanup have indicated that their cleanup is finished.

**Note:** The system defaults to LOCALCLEANUP=CONTINUE if a group exit is not provided. XCF may also proceed as if LOCALCLEANUP=CONTINUE is specified if the group exit routine fails when presented with the "system being removed from sysplex" event (GEPLTYPE=GESYSPT).

#### **CONTINUE**

The member does not need time to perform additional cleanup after its group exit is presented with the "system being removed from the sysplex" event (GEPLTYPE=GESYSPT). XCF may wait for other members to finish their cleanup, but it need not wait for this member to perform cleanup. With respect to this member, XCF can immediately continue with system termination.

#### **,MEMASSOC=TASK**

#### **,MEMASSOC=JOBSTEP**

#### **,MEMASSOC=ADDRSPACE**

Use this input parameter to specify a member's association with a task, job step task, or address space. XCF uses the MEMASSOC value to determine when to terminate a member (put the member in the failed or not-defined state). For more information on member termination, see [z/OS MVS Programming: Sysplex Services Guide](#).

If you code MEMASSOC=TASK, the active member is associated with the task under which IXCJOIN was issued. XCF transitions the member to a "non-active" state when the associated task terminates. This means that if the member joins with LASTING=NO, the member becomes "not defined." If the member joins with LASTING=YES, XCF puts the member in a "failed" state.

If you code MEMASSOC=JOBSTEP, the active member is associated with the job step task under which IXCJOIN was issued. XCF transitions the member to a "non-active" state when the associated job terminates.

If you code MEMASSOC=ADDRSPACE, the active member is associated with the address space under which IXCJOIN was issued. XCF transitions the member to a "non-active" state when the associated address space terminates. Note that with this option, the system cannot provide SRB to task percolation.

**Remember:** An initiator address space does not end when a batch job running in it ends. So, if you are issuing IXCJOIN from a batch job and you want to terminate the member when the batch job ends, you must associate the member with the job, (MEMASSOC=JOBSTEP) not the initiator address space.

**,MEMDATA=NO\_MEMDATA**

**,MEMDATA=memdata**

Use this input parameter to specify a 64-bit member data field. XCF provides this field to the group, status, message, and message notify user routines for this member. If you do not specify MEMDATA, or you specify MEMDATA=NO\_MEMDATA, XCF sets the member data field to zeros.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-byte field that contains member data.

**,MEMNAME=NO\_MEMNAME**

**,MEMNAME=memname**

Use this input parameter to specify the name you choose for the member. If the member was previously in a created, failed, or quiesced state, you already have defined the member name. The name must be 16 characters long, padded on the right with blanks if necessary; the valid characters are A-Z, 0-9, and national characters (@, # and \$).

If you use LASTING=YES, MEMNAME is required (MEMNAME=NO\_MEMNAME is not acceptable). If you are defining one member per system, consider using the SYSNAME from CVTSNAME as the member name.

If you do not specify MEMNAME, or if you specify MEMNAME=NO\_MEMNAME, XCF generates the member name in the form Mnnnnnnnnnnnnnnnn, where nnnnnnnnnnnnnnn is a number.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-byte field that contains the XCF member name.

**,MF=S**

**,MF=(L,mfctrl)**

**,MF=(L,mfctrl,mfattr)**

**,MF=(L,mfctrl,0D)**

**,MF=(M,mfctrl)**

**,MF=(M,mfctrl,COMPLETE)**

**,MF=(M,mfctrl,NOCHECK)**

**,MF=(E,mfctrl)**

**,MF=(E,mfctrl,COMPLETE)**

**,MF=(E,mfctrl,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

#### **,mfctrl**

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

#### **,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

#### **,COMPLETE**

#### **,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

#### **COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if SMILE=*var* were an optional parameter and the default is SMILE=NO\_SMILE then it would not be documented. However, if the default was SMILE=-), then it would be documented because a value would be the default.

#### **NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

#### **,MSGEXIT=NO\_MSGEXIT**

#### **,MSGEXIT=msgexit**

Use this input parameter to specify the address of the message user routine. This routine executes in the primary address space of the issuer of the IXCJOIN macro, in 31-bit addressing mode and in SRB mode. The routine receives control when a message becomes available for this member from another member of the group.

See *z/OS MVS Programming: Sysplex Services Guide* for information about how to code a message user routine.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 4-byte field that contains the address of the message exit routine.

#### **,MSGISO=NONE**

#### **,MSGISO=MSGORSN**

Use this input parameter to indicate the degree to which the member is to be made aware of message isolation. A member is "message isolated" when XCF determines that the member is consuming so much signalling resource that it impedes the ability of XCF to deliver signals to other members. Messages sent with the XCF Message-Out Service (macros IXCMGO or IXCMGOX) to a member that is "message isolated" will either be delayed or rejected.

#### **NONE**

indicates that the member is not prepared to deal with unique reason codes related to message isolation. The delay or reject of an XCF Message-Out request targeted to a member that is message isolated will be attributed to a "no buffer" condition (ixcMsgoRsnNoBuffer).

#### **MSGORSN**

indicates that the member is capable of dealing with unique reason codes related to message isolation. The delay or rejection of an XCF Message-out request targeted to a member that is message isolated will be attributed to "message isolation".

The reason code (ixcMsgoRsnTargetIsolated) could be reported by any XCF interface that reports the result of a Message-out request. For example, it can be returned by either IXCMSGO or IXCMSGOX. It can be reported to the notify exit by the MNPL (macro IXCYMNPL) or to the invoker of IXCMSGC when querying the state of a message (by macro IXCYMQAA).

MSGISO=NONE is the default. If both NONE and MSGORSN are specified, MSGORSN takes precedence and NONE is ignored.

**,MSGOUTASID=MEMBER**

**,MSGOUTASID=ANY**

Use this input parameter to indicate which address spaces can use the IXCMSGO macro to send messages to the members in the specified XCF group.

**MEMBER**

Indicates that when IXCMSGO is issued, the primary address space must equal the requesting member's primary address space at the time the group was joined, or the primary address space must be the MASTER address space.

**ANY**

Indicates that IXCMSGO can be issued from any address space.

**,NOTIFYEXIT=NO\_NOTIFYEXIT**

**,NOTIFYEXIT=notifyexit**

Use this input parameter to specify the name of a user routine to receive control for message notifications. The routine must reside in the user's address space and run in 31-bit addressing mode. See *z/OS MVS Programming: Sysplex Services Guide* for the types of events that the system can present to the message notify user routine.

Specifying or defaulting to NO\_NOTIFYEXIT indicates that the member does not want to receive unsolicited notification of events that have occurred. Lack of a message notify user routine at join time does not prevent the member from specifying the name of a routine to other services (such as IXCMSGO) that support it.

See *z/OS MVS Programming: Sysplex Services Guide* for information about how to code a message notify user routine.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword field that contains the address of the message notify user routine.

**,PLISTVER=IMPLIED\_VERSION**

**,PLISTVER=MAX**

**,PLISTVER=plistver**

Use this input parameter to specify the version of the macro. See "Understanding IXCJOIN Version Support" on page 81 for a description of the options available with PLISTVER.

**,RECOVERYMGR=NO**

**,RECOVERYMGR=YES**

Use this input parameter to indicate whether the member is a recovery manager. A recovery manager is a group member that coordinates a sysplex-wide recovery process.

**NO**

The member does not designate itself as a recovery manager.

**YES**

The member designates itself as a recovery manager.

**,RETCODE=retcode**

Use this output parameter to specify the location in which the system is to copy the return code from GPR 15.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 4-byte field that will contain the return code when the join request is complete.

**,RSNCODE=rsncode**

Use this output parameter to specify the location in which the system is to copy the reason code from GPR 0.



**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 4-byte field that will contain the reason code (if any) when the join request completes.

**,STATEXIT=statexit**

Use this input parameter to specify the address of the status user routine. This routine executes in the primary address space of the issuer of the IXCJOIN macro, in 31-bit addressing mode and in SRB mode. The routine receives control if the status field of the member is unchanged during the time period determined by the INTERVAL parameter. If you specify STATEXIT, you must also specify STATFLD and INTERVAL.

See *z/OS MVS Programming: Sysplex Services Guide* for information about how to code a status user routine.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 4-byte field that contains the address of the status user-routine.

**,STATFLD=NO\_STATFLD**

**,STATFLD=statfld**

Use this input parameter to specify a 64-bit status field in fixed or disabled reference (DREF) common storage, with any storage key. If the status field remains unchanged over the specified interval, XCF schedules the status user-routine identified in STATEXIT. If you specify STATFLD, you must also specify STATEXIT and INTERVAL.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-byte field that contains the status of the member.

**,SYSCLEANUPMEM=NO**

**,SYSCLEANUPMEM=YES**

Use this input parameter to specify whether the member will perform system-wide cleanup after a system leaves the sysplex. If YES is specified, the system will wait for this member to issue the IXCSYSCL macro before any elements may be restarted by automatic restart management.

**NO**

Indicates the member joining the group will not perform system-wide cleanup of resources when a system leaves the sysplex.

**YES**

Indicates the member joining the group will perform system-wide cleanup when a system leaves the sysplex. When this member is notified that a system has left the sysplex, it must issue the IXCSYSCL macro to indicate to the system that cleanup is complete.

**,TERMLEVEL=MEMASSOC**

**,TERMLEVEL=ADDRSPACE**

**,TERMLEVEL=SYSTEM**

Use this input parameter to specify the first member-specific termination action the system is to take against this member when it needs to be terminated. For example, XCF may terminate a member that is determined to be causing signalling sympathy sickness, or a critical member that becomes impaired.

**MEMASSOC**

The system will use the MEMASSOC keyword specification to determine the task or address space with which the member is associated for termination purposes.

- For MEMASSOC=TASK, the task from which the member invoked the IXCJOIN macro will be terminated.
- For MEMASSOC=JOBSTEP, the job step task from which the member invoked the IXCJOIN macro will be terminated.
- For MEMASSOC=ADDRSPACE, the address space from which the member invoked the IXCJOIN macro will be terminated.

**ADDRSPACE**

The system will terminate the address space from which the member invoked the IXCJOIN macro.

**SYSTEM**

The system will terminate the system on which the member resides. The system will enter wait state and be removed from the sysplex.

**Note:**

1. The termination will cause any other members associated with the relevant task, space, or system to be terminated too.
2. The setting on TERMLEVEL is honored whenever XCF needs to terminate the member, for example, when IXCTERM is used to terminate a target member.

**,USLEN=*uslen***

Use this input parameter to specify the length in bytes of the user state data that you provide on the USTATE parameter. The length must be from 1 to 32 bytes. If you specify less than 32 bytes, XCF pads the remainder of the user state field, up to 32 bytes, on the right with zeros. IXCQUERY always returns the full 32 bytes, and group user-routines always receive the full 32 bytes. If you code USTATE, USLEN is required.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 4-byte field that contains the length of the user state data (USTATE).

**,USTATE=NO\_USTATE****,USTATE=*ustate***

Use this input parameter to specify the user state data that XCF is to place in the member's user state field. If you do not specify the USTATE parameter, XCF retains the existing value in the user state field unless the joining member was previously not-defined. In this case, XCF clears the user state field to zeros. Specify the length of the user state data on the USLEN parameter.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an area (with a length of USLEN) that contains the user state information.

## ABEND Codes

---

None.

## Return and Reason Codes

---

When the IXCJOIN macro returns control to your program:

- GPR 15 (and *retcode*, if you coded RETCODE) contains a return code.
- GPR 0 (and *rsncode*, if you coded RSNCODE) contains a reason code, if applicable.

Macro IXCYCON provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

**0**

IXCRETCODEOK

**4**

IXCRETCODEWARNING

**8**

IXCRETCODEPARMERROR

**C**

IXCRETCODEENVERROR

**10**

IXCRETCODECOMPERROR

The following table identifies the hexadecimal return and reason codes and the equate symbol associated with each reason code.

Table 7. Return and Reason Codes for the IXCJOIN Macro

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
00	None.	<p><b>Meaning:</b> IXCJOIN completed successfully. XCF places the member in the active state, and places member information in ANSAREA.</p> <p><b>Action:</b> None.</p>
04	04	<p><b>Equate Symbol:</b> IXCJOINRSNFIRSTACTIVEMEMBER</p> <p><b>Meaning:</b> IXCJOIN completed successfully; the member is the first active member of the group. XCF places member information in ANSAREA.</p> <p><b>Action:</b> None required.</p>
04	08	<p><b>Equate Symbol:</b> IXCJOINRSNWASFAILED</p> <p><b>Meaning:</b> IXCJOIN completed successfully; the member already exists in a failed state, and the caller correctly specified LASTING=YES. A new member token has been assigned to the new instance of the member.</p> <p><b>Action:</b> None required. However, because the previous instance of the member was in a failed state, your application might need to perform some type of cleanup or takeover processing. MVS placed the member in the failed state when the member's corresponding task, address space, or system terminated. If the user state of the previous instance is needed and you must specify a user state on the IXCJOIN request, you can change your program to issue an IXCQUERY prior to the IXCJOIN to capture the user state of the previous instance. If you do not specify a user state on IXCJOIN, the new instance of the member will retain the user state of the old instance.</p>
04	0C	<p><b>Equate Symbol:</b> IXCJOINRSNWASQUIESCED</p> <p><b>Meaning:</b> IXCJOIN completed successfully; the member already exists in a quiesced state, and the caller correctly specified LASTING=YES. A new member token has been assigned to the new instance of the member.</p> <p><b>Action:</b> None required. However, because the previous instance of the member was in a quiesced state, your application might need to perform some type of cleanup or takeover processing. The previous instance of the member voluntarily placed itself in a quiesced state. If the user state of the previous instance is needed and you must specify a user state on the IXCJOIN request, you can change your program to issue an IXCQUERY prior to the IXCJOIN to capture the user state of the previous instance. If you do not specify a user state on IXCJOIN, the new instance of the member will retain the user state of the old instance.</p>

Table 7. Return and Reason Codes for the IXCJOIN Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
04	10	<p><b>Equate Symbol:</b> IXCJOINRSNWASCREATED</p> <p><b>Meaning:</b> IXCJOIN completed successfully. The member already exists in a created state, and the caller correctly specified LASTING=YES.</p> <p><b>Action:</b> None required. However, you might take some action based on your application. You can use the member's current user state to determine what action needs to be taken.</p>
08	04	<p><b>Equate Symbol:</b> IXCJOINRSNISCREATED</p> <p><b>Meaning:</b> Program error. The member already exists in a created state, and the caller did not specify LASTING=YES. LASTING=YES must be specified since the member already has permanent status recording. The member's state and user state are not altered.</p> <p><b>Action:</b> Ensure that your program specifies LASTING=YES, and retry the request. If permanent status recording is not required for this instance of the member, issue IXCDELET to delete the member, and retry the IXCJOIN request.</p>
08	08	<p><b>Equate Symbol:</b> IXCJOINRSNISACTIVE</p> <p><b>Meaning:</b> Program error. The member already exists in an active state.</p> <p><b>Action:</b> None required. However, you might take some action depending on your application.</p>
08	0C	<p><b>Equate Symbol:</b> IXCJOINRSNISQUIESCED</p> <p><b>Meaning:</b> Program error. The member already exists in a quiesced state, and the caller did not specify LASTING=YES. LASTING=YES must be specified since the member already has permanent status recording. The member's state and user state are not altered.</p> <p><b>Action:</b> Ensure that your program specifies LASTING=YES, and retry the request. If permanent status recording is not required for this instance of the member, issue IXCDELET to delete the member, and retry the IXCJOIN request. Before you delete the member, you can use the IXCQUERY service to capture the member's current user state.</p>
08	10	<p><b>Equate Symbol:</b> IXCJOINRSNISFAILED</p> <p><b>Meaning:</b> Program error. The member already exists in a failed state, and the caller did not specify LASTING=YES.</p> <p><b>Action:</b> Ensure that your program specifies LASTING=YES, and retry the request. If permanent status recording is not required for this instance of the member, issue IXCDELET to delete the member, and retry the IXCJOIN request. Prior to deleting the member, you can use the IXCQUERY service to capture the member's current user state.</p>

Table 7. Return and Reason Codes for the IXCJOIN Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
08	14	<b>Equate Symbol:</b> IXCJOINRSNGRPNAMEBAD <b>Meaning:</b> Program error. The group name is not valid. <b>Action:</b> Correct the group name, and retry the request.
08	18	<b>Equate Symbol:</b> IXCJOINRSNMEMNAMEBAD <b>Meaning:</b> Program error. The member name is not valid. <b>Action:</b> Correct the member name, and retry the request.
08	1C	<b>Equate Symbol:</b> IXCJOINRSNINTERVALBAD <b>Meaning:</b> Program error. The status-checking interval value is zero or is not a multiple of 100. <b>Action:</b> Correct the status-checking interval, and retry the request.
08	20	<b>Equate Symbol:</b> IXCJOINRSNSTATFLDBADSTG <b>Meaning:</b> Program error. XCF could not access the STATFLD. <b>Action:</b> Ensure that the STATFLD parameter is correct. The STATFLD must be a doubleword in fixed or DREF common storage.
08	24	<b>Equate Symbol:</b> IXCJOINRSNLASTINGNEEDSMEMNAME <b>Meaning:</b> Program error. The caller specified LASTING=YES without specifying the member name. XCF will not generate member names for members that have permanent status recording. <b>Action:</b> If the member requires permanent status recording, provide a member name. If the member does not require permanent status recording, do not specify LASTING=YES.
08	28	<b>Equate Symbol:</b> IXCJOINRSNSTATUSMONINCOMPLETE <b>Meaning:</b> Program error. The STATFLD, INTERVAL, or STATEXIT parameter is missing. For XCF to monitor the member's status, all three of these values must be specified. <b>Action:</b> Ensure that the STATFLD, INTERVAL, and STATEXIT are correct, and retry the request.

Table 7. Return and Reason Codes for the IXCJOIN Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
08	xyyy003C	<p><b>Equate Symbol:</b> IXCJOINRSNANSAREAINCOMPLETE</p> <p><b>Meaning:</b> Program error. The caller specified ANSAREA or ANSLEN incorrectly. Even though the member might have been placed in an active state, the ANSAREA was nonaddressable or the ANSLEN was not large enough.</p> <p><b>Action:</b> Check the two high-order bytes of this reason code fullword xyyy003C for the return code xx (either 00 or 04) and reason code yy the caller would have received from IXCJOIN if the caller had coded those parameters correctly. Take action as described by the corresponding return and reason code. You might want to abnormally end your program or take some other action that will record the problem. You should correct your program to ensure that the ANSAREA is addressable and that the ANSLEN is large enough. See the description of these keywords for their requirements.</p> <p>If the return and reason code indicate that the member was placed in an active state, you can use the IXCQUERY service to get the information that would have been returned in the ANSAREA.</p> <p><b>Note:</b> You should specify a member name or a unique user state value on IXCJOIN invocation. If you request to have XCF define the member name, you might not be able to use IXCQUERY to determine which member was created on your behalf.</p>
08	40	<p><b>Equate Symbol:</b> IXCJOINRSNPLISTRSDNOTVALID</p> <p><b>Meaning:</b> Program error or environmental error. A reserved field in the control parameter list is not zero.</p> <p><b>Action:</b> Check to see if your program inadvertently overlaid the parameter list storage, and that it was assembled with the correct macro library for the release of MVS your program is running on.</p>
08	44	<p><b>Equate Symbol:</b> IXCJOINRSNMEMASSOCBAD</p> <p><b>Meaning:</b> Program error. The member association value (task, address space or job) is not correct in the parameter list.</p> <p><b>Action:</b> Check to see if your program inadvertently overlaid the parameter list storage, and that it was assembled with the correct macro library for the release of MVS your program is running on.</p>

Table 7. Return and Reason Codes for the IXCJOIN Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
08	100	<p><b>Equate Symbol:</b> IXCJOINRSNPLISTBADALET</p> <p><b>Meaning:</b> Program error. Your program is not running in primary ASC mode, and the ALET that qualifies the address of the control parameter list is neither zero nor associated with a valid public entry on the caller's DU-AL.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• Your program is not intended to run in primary ASC mode.</li> <li>• You specified SYSSTATE ASCENV=AR before issuing the IXCJOIN macro.</li> <li>• The ALET for the parameter list is a valid public entry on the DU-AL or is zero (primary address space ALET).</li> </ul>
08	104	<p><b>Equate Symbol:</b> IXCJOINRSNPLISTVERSIONNOTVALID</p> <p><b>Meaning:</b> Program error or environmental error. The version number in the control parameter list is not valid.</p> <p><b>Action:</b> Check to see if your program inadvertently overlaid the parameter list storage, and that it was assembled with the correct macro library for the release of MVS your program is running on.</p>
08	108	<p><b>Equate Symbol:</b> IXCJOINRSNPLISTBADFUNCTION</p> <p><b>Meaning:</b> Program error or environmental error. The function code in the control parameter list is not valid.</p> <p><b>Action:</b> Check to see if your program inadvertently overlaid the parameter list storage, and that it was assembled with the correct macro library for the release of MVS your program is running on.</p>
08	10C	<p><b>Equate Symbol:</b> IXCJOINRSNPLISTBADSTG</p> <p><b>Meaning:</b> Program error. XCF could not access the control parameter list.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• Ensure that the correct parameter list storage area was specified.</li> <li>• If your program is running in AR ASC mode: <ul style="list-style-type: none"> <li>– Ensure that you specified SYSSTATE ASCENV=AR before issuing the IXCJOIN macro.</li> <li>– Ensure that the parameter list ALET is correct.</li> </ul> </li> <li>• Ensure that the parameter list storage area was not inadvertently freed by your program.</li> </ul>

Table 7. Return and Reason Codes for the IXCJOIN Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
08	110	<p><b>Equate Symbol:</b> IXCJOINRSNUSTATEBADSTG</p> <p><b>Meaning:</b> Program error. XCF could not access the USTATE value.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• Ensure that the correct USTATE address was specified.</li> <li>• If your program is running in AR ASC mode: <ul style="list-style-type: none"> <li>– Ensure that you specified SYSSTATE ASCENV=AR before issuing the IXCJOIN macro.</li> <li>– Ensure that the USTATE ALET corresponds to the parameter list address.</li> </ul> </li> <li>• Ensure that the USTATE storage area was not inadvertently freed by your program.</li> </ul>
08	114	<p><b>Equate Symbol:</b> IXCJOINRSNUSLENBADVALUE</p> <p><b>Meaning:</b> Program error. The USLEN value is less than 1 or greater than 32.</p> <p><b>Action:</b> Correct the USLEN, and retry the request.</p>
08	118	<p><b>Equate Symbol:</b> IXCJOINRSNNOTTASKMODE</p> <p><b>Meaning:</b> Program error. The caller is not in task mode.</p> <p><b>Action:</b> Correct your program so that it issues IXCJOIN only while in task mode.</p>
08	11C	<p><b>Equate Symbol:</b> IXCJOINRSNNOTENABLED</p> <p><b>Meaning:</b> The caller is not enabled.</p> <p><b>Action:</b> Correct your program so that it does not issue IXCJOIN while it is disabled.</p>
08	120	<p><b>Equate Symbol:</b> IXCJOINRSNPRIMARYNOTHOME</p> <p><b>Meaning:</b> Program error. The primary address space is not equal to the home address space.</p> <p><b>Action:</b> Correct your program so that it does not use IXCJOIN while in cross memory mode. You might want to pass this restriction on to your caller when you are unsure of the environment in which your caller may call your program.</p>
08	128	<p><b>Equate Symbol:</b> IXCJOINRSNTASKTERM</p> <p><b>Meaning:</b> Program error. XCF does not allow the JOIN process during or after task termination.</p> <p><b>Action:</b> Correct your program so that it does not issue IXCJOIN from a task that is terminating. Ensure that your application does not invoke IXCJOIN from a task termination resource manager. You might want to pass this restriction on to your caller when you are unsure of the environment in which your caller may call you.</p>



Table 7. Return and Reason Codes for the IXCJOIN Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
08	12C	<p><b>Equate Symbol:</b> IXCJOINRSNFUNCDESCBAD</p> <p><b>Meaning:</b> Program error. The FUNCTION keyword does not specify a valid descriptor (bad character, leading blanks, all blank).</p> <p><b>Action:</b> Correct the FUNCTION specification.</p>
0C	04	<p><b>Equate Symbol:</b> IXCJOINRSNMAXGROUPS</p> <p><b>Meaning:</b> Environmental error. The maximum number of groups already exists.</p> <p><b>Action:</b> Retry the request at least once. If the problem persists, consult your system programmer to determine if the number of groups defined in the couple data set should be increased, or if the XCF limit on the number of groups has been reached. Your application should cover cases in which the couple dataset has no available room. Before running or installing your application, you should make the system programmer aware of your XCF group and member resource requirements.</p>
0C	08	<p><b>Equate Symbol:</b> IXCJOINRSNMAXMEMBERS</p> <p><b>Meaning:</b> Environmental error. The maximum number of members in the group already exists.</p> <p><b>Action:</b> Retry the request at least once. If the problem persists, consult your system programmer to determine if the maximum number of members defined in the couple data set should be increased, or if the XCF limit on the number of members has been reached. Your application should cover cases in which no room is available within the group. Before running or installing your application, you should make the system programmer aware of your XCF group and member resource requirements.</p>
0C	10	<p><b>Equate Symbol:</b> IXCJOINRSNPARTITIONING</p> <p><b>Meaning:</b> Environmental error. The system is being removed from the sysplex, and XCF permanently suspends all requests to join a group on this system.</p> <p><b>Action:</b> The action is dependent on your application. You might want to prepare your application for system termination. (See the chapter on XCF in <a href="#">z/OS MVS Programming: Sysplex Services Guide</a>)</p>

Table 7. Return and Reason Codes for the IXCJOIN Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0C	14	<p><b>Equate Symbol:</b> IXCJOINRSNXCLOCALMODE</p> <p><b>Meaning:</b> Environmental error. IXCJOIN with LASTING=YES is not allowed when the sysplex is in XCF-local mode. Permanent status recording requires a couple data set in which XCF can retain the member's status. XCF-local mode does not support a couple data set.</p> <p><b>Action:</b> This depends on your program. If XCF-local mode is not supported by your application, you must make the system programmer aware of this requirement. The system programmer will have to ensure that the system is not IPLed in XCF-local mode. Before running or installing your application, you should make the system programmer aware of your XCF group and member resource requirements. (See <a href="#">z/OS MVS Programming: Sysplex Services Guide</a>.)</p>
10	None.	<p><b>Meaning:</b> System error. XCF processing failed.</p> <p><b>Action:</b> Retry the request at least once. If the problem persists, record the return code and reason code, if applicable, and supply them to the appropriate IBM support personnel.</p>

## Example

*Operation:* Join a member MEMB1 to a group MYGROUP with LASTING=NO. Place X'11' in the user state field, and assign an interval of 1000 (or 10 seconds). Register 2 points to the area where the member information will be returned. The user-routine addresses are as follows:

- Group user-routine address is in register 4.
- Status user-routine address is in register 5.
- Message user-routine address is in register 6.

The status field must be in common storage. Register 7 has the address of this area, obtained through the STORAGE OBTAIN macro. XCF is to store the return code and reason code into the variables RETURN and REASON. The code is as follows:

```

LA    R2,MYAREA           OBTAIN ADDRESS OF OUTPUT AREA X
                                FOR IXCJOIN
L     R4,GXTADDR          OBTAIN ADDRESS OF GROUP      X
                                USER-ROUTINE FOR IXCJOIN
L     R5,SXTADDR          OBTAIN ADDRESS OF STATUS      X
                                USER-ROUTINE FOR IXCJOIN
L     R6,MXTADDR          OBTAIN ADDRESS OF MESSAGE     X
                                USER-ROUTINE FOR IXCJOIN
STORAGE OBTAIN,LENGTH=8,SP=228 OBTAIN STORAGE FOR STATUS X
                                FIELD
ST    R1,FIELD1           SAVE ADDRESS OF STATUS FIELD
LR    R7,R1              PLACE ADDRESS IN REGISTER FOR X
                                IXCJOIN INVOCATION

IXCJOIN GRPNAME=MYGROUP,ANSAREA=(R2),ANSLEN=AREALEN,      X
        LASTING=NO,MEMNAME=MEMB1,STATFLD=(R7),          X
        MEMASSOC=JOBSTEP,                                X
        GRPEXIT=(R4),STATEXIT=(R5),MSGEXIT=(R6),        X
        MEMDATA=DATA1,INTERVAL=INTER1,                  X
        USTATE=STATE1,USLEN=LEN,                         X
        RETCODE=RETURN,RSNCODE=REASON,MF=S

EXTRN GEXIT

```

	EXTRN	SEXIT		
	EXTRN	MEXIT		
MYGROUP	DC	CL8'MYGROUP '	GROUP NAME	
MYAREA	DS	CL124	OUTPUT AREA TO CONTAIN DATA	X
			RETURNED BY IXCJOIN	
DATA1	DS	CL8	MEMBER DATA FOR THIS MEMBER	
FIELD1	DS	1F	ADDRESS OF STATUS FIELD	
RETURN	DS	1F	RETURN CODE	
REASON	DS	1F	REASON CODE	
STATE1	DC	X'11'	USER STATE VALUE	
LEN	DC	F'1'	LENGTH OF USER STATE DATA	
INTER1	DC	F'1000'	INTERVAL VALUE	
GXTADDR	DC	A(GEXIT)	ADDRESS OF GROUP USER-ROUTINE	
SXTADDR	DC	A(SEXIT)	ADDRESS OF STATUS USER-ROUTINE	
MXTADDR	DC	A(MEXIT)	ADDRESS OF MESSAGE USER-ROUTINE	
AREALEN	DC	F'124'	LENGTH OF OUTPUT AREA	
MEMB1	DC	CL16'MEMB1	MEMBER NAME	



## Chapter 9. IXCLEAVE — Place an XCF Member in the Not-Defined State

### Description

The IXCLEAVE macro allows a member of a cross-system coupling facility (XCF) group to change its state from active to not-defined. XCF does not recognize the existence of a member that is in a not-defined state.

Additionally, IXCLEAVE notifies active members in the group that have defined a group user-routine that the target member is now in the not-defined state. XCF delivers outstanding messages sent by the member and discards undelivered messages that were sent to the member.

### Environment

The following are the environment requirements for the caller.

Environment	Environment requirement
Minimum authorization:	Supervisor state or PKM allowing key 0 - 7
Dispatchable unit mode:	Task
Cross memory mode:	PASN=HASN, any HASN, any SASN. The primary address space must be the same as the primary address space of the caller of the IXCJOIN that placed the member in the active state, or the caller must be executing in the master scheduler address space.
AMODE:	31-bit
ASC mode:	Primary or access register (AR)
Interrupt status:	Enabled for I/O and external interrupts
Locks:	No locks held
Control parameters:	Must be in the primary address space or be in an address/data space that is addressable through a public entry on the caller's dispatchable unit access list (DU-AL)

### Programming Requirements

If the program is in access register (AR) mode, issue the SYSSTATE ASCENV=AR macro before IXCLEAVE. SYSSTATE ASCENV=AR tells the system to generate code appropriate for AR mode.

### Restrictions

The caller can have no enabled, unlocked task (EUT) FRRs established.

### Input Register Information

Before issuing the IXCLEAVE macro, the caller does not have to place any information into any register unless using it in register notation for a particular parameter, or using it as a base register.

## Output Register Information

When control returns to the caller, the general purpose registers (GPRs) contain:

### Register Contents

**0**

If GPR 15 contains a zero or X'10', GPR 0 is used as a work register by the system; otherwise, GPR 0 contains a reason code.

**1**

Used as a work register by the system.

**2-13**

Unchanged.

**14**

Used as a work register by the system.

**15**

Return code.

When control returns to the caller, the access registers (ARs) contain:

### Register Contents

**0-1**

Used as work registers by the system

**2-13**

Unchanged

**14-15**

Used as work registers by the system

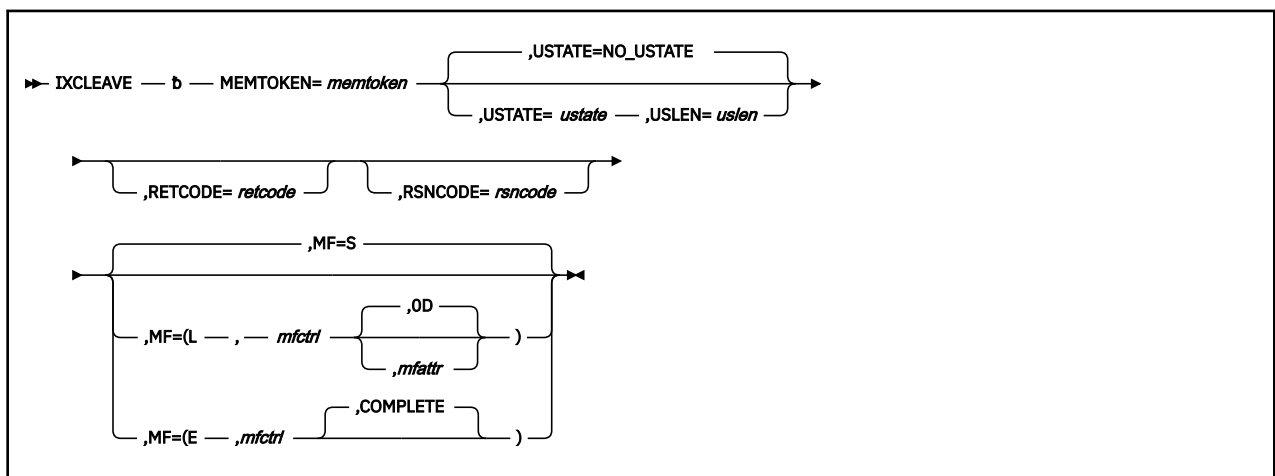
Some callers depend on register contents remaining the same before and after issuing a service. If the system changes the contents of registers on which the caller depends, the caller must save them before issuing the service, and restore them after the system returns control.

## Performance Implications

None.

## Syntax Diagram

The syntax of the IXCLEAVE macro is as follows:



## Parameter Descriptions

The parameter descriptions are listed in alphabetical order. Default values are underlined:

### **MEMTOKEN=*memtoken***

Use this input parameter to specify the 64-bit token of the member that IXCLEAVE is to place in the not-defined state. XCF provided this token when it activated the member.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-byte field that contains the member token that was returned in the ANSAREA when IXCJOIN completed.

### **,MF=S**

### **,MF=(L,*mfctrl*)**

### **,MF=(L,*mfctrl*,*mfattr*)**

### **,MF=(L,*mfctrl*,0D)**

### **,MF=(M,*mfctrl*)**

### **,MF=(M,*mfctrl*,COMPLETE)**

### **,MF=(M,*mfctrl*,NOCHECK)**

### **,MF=(E,*mfctrl*)**

### **,MF=(E,*mfctrl*,COMPLETE)**

### **,MF=(E,*mfctrl*,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

### **,*mfctrl***

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

### **,*mfattr***

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

### **,COMPLETE**

### **,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

### **COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if SMILE=*var* were an optional parameter and the default is SMILE=NO\_SMILE then it would not be documented. However, if the default was SMILE=:-), then it would be documented because a value would be the default.

**NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

**,RETCODE=*retcode***

Use this output parameter to specify a field to contain the return code. (The return code is also returned in register 15.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the return code when the request has completed.

**,RSNCODE=*rsncode***

Use this output parameter to specify a field to contain the reason code returned, if applicable. (The reason code is also returned in register 0.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the reason code (if any) when the request has completed.

**,USLEN=*uslen***

Use this input parameter to specify the length in bytes of the user state data that you provide on the USTATE parameter. The length must be from 1 to 32 bytes. XCF overlays any previous value in the user state field up to the length you specify on USLEN. XCF does not pad the remainder of the user state field (up to 32 bytes) with zeros. IXCQUERY always returns the full 32 bytes, and group user-routines always receive the full 32 bytes. If you code USTATE, USLEN is required.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 4-byte field that contains the length of the user state data (USTATE).

**,USTATE=NO USTATE****,USTATE=*ustate***

Use this input parameter to specify the area containing data that you want XCF to place in the user state field associated with the member. Use the USLEN parameter to specify the length of the user state data.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the area (with a length of USLEN) that contains the user state information.

## ABEND Codes

---

None.

## Return and Reason Codes

---

When the IXCLEAVE macro returns control to your program:

- GPR 15 (and *retcode*, if you coded RETCODE) contains a return code.
- GPR 0 (and *rsncode*, if you coded RSNCODE) contains a reason code, if applicable.

Macro IXCYCON provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

**0**

IXCRETCODEOK

**4**

IXCRETCODEWARNING

**8**

IXCRETCODEPARMERROR

**C**

IXCRETCODEENVERROR

**10**

IXCRETCODECOMPERROR



The following table identifies the hexadecimal return and reason codes and the equate symbol associated with each reason code.

Table 8. Return and Reason Codes for the IXCLEAVE Macro		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
00	None.	<p><b>Meaning:</b> IXCLEAVE completed successfully. XCF places the member in the not-defined state and informs group members of the change.</p> <p><b>Action:</b> None.</p>
04	04	<p><b>Equate Symbol:</b> IXCLEAVERSNEXTSNOTPURGED</p> <p><b>Meaning:</b> Environmental error. IXCLEAVE completed successfully. XCF did not purge the group, status, or message user-routine SRBs successfully. For a group, status, or message exit, it is possible that the member specified on IXCJOIN is still running. Once an exit returns to XCF, it will not be scheduled again.</p> <p><b>Action:</b> Your application should be aware that one of the exits might still be running.</p> <p>If this return code occurs more than once for IXCLEAVE, take the following actions:</p> <ul style="list-style-type: none"> <li>• Determine if any asynchronous abends are being issued against the current task. If so, you may need to reduce their frequency.</li> <li>• XCF should have taken an SDUMP to record the abend or abends. If the SDUMP indicates that the error was caused by SRB-to-task percolation or application code issuing a CALLRTM against your task, this is probably not an XCF error. If you feel this is an XCF error, record the return and reason code, and supply it to the appropriate IBM support personnel.</li> </ul>
08	04	<p><b>Equate Symbol:</b> IXCLEAVERSNNOTACTIVE</p> <p><b>Meaning:</b> Program error. The member token does not identify an active member.</p> <p><b>Action:</b> Ensure that the correct MEMTOKEN was specified. Further action depends on your application. If the member must be in a not-defined state, you can use the IXCQUERY service to determine what state the member is currently in. If the member is currently in a failed, quiesced, or created state, you can use the IXCDELET service to place the member in a not-defined state.</p>
08	08	<p><b>Equate Symbol:</b> IXCLEAVERSNNINAPPROPRIATEPRIMARY</p> <p><b>Meaning:</b> Program error. The primary address space is neither the master scheduler address space nor the primary address space of the caller that issued IXCJOIN to place the member in the active state.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The correct MEMTOKEN was specified.</li> <li>• Your program issues IXCLEAVE only from the master scheduler address space or the address space from which the member (MEMTOKEN) joined the group.</li> </ul>

Table 8. Return and Reason Codes for the IXCLEAVE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
08	10	<p><b>Equate Symbol:</b> IXCLEAVERSINAPPROPRIATESYSTEM</p> <p><b>Meaning:</b> Program error. The system is not the system on which the IXCJOIN for the member was issued.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The correct MEMTOKEN was specified.</li> <li>• Your program issues IXCLEAVE only for members that joined on the same system as your program.</li> </ul>
08	40	<p><b>Equate Symbol:</b> IXCLEAVERSINPLISTRSDNOTVALID</p> <p><b>Meaning:</b> Program error or environmental error. A reserved field in the control parameter list is not zero.</p> <p><b>Action:</b> Check to see if your program inadvertently overlaid the parameter list storage, and that it was assembled with the correct macro library for the release of MVS your program is running on.</p>
08	100	<p><b>Equate Symbol:</b> IXCLEAVERSINPLISTBADALET</p> <p><b>Meaning:</b> Program error. Your program is running in AR mode, and the ALET that qualifies the address of the control parameter list is neither zero nor associated with a valid public entry on the caller's DU-AL.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• You specified SYSSTATE ASCENV=AR prior to issuing the IXCLEAVE macro.</li> <li>• The ALET for the parameter list is a valid public entry on the DU-AL or is zero (primary address space ALET).</li> <li>• Your program is not intended to run in primary ASC mode.</li> </ul>
08	104	<p><b>Equate Symbol:</b> IXCLEAVERSINPLISTVERSIONNOTVALID</p> <p><b>Meaning:</b> Program error or environmental error. The version number in the control parameter list is not valid.</p> <p><b>Action:</b> Check to see if your program inadvertently overlaid the parameter list storage, and that it was assembled with the correct macro library for the release of MVS your program is running on.</p>
08	108	<p><b>Equate Symbol:</b> IXCLEAVERSINPLISTBADFUNCTION</p> <p><b>Meaning:</b> Program error. The function code in the control parameter list is not valid.</p> <p><b>Action:</b> Check to see if your program inadvertently overlaid the parameter list storage.</p>

Table 8. Return and Reason Codes for the IXCLEAVE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
08	10C	<p><b>Equate Symbol:</b> IXCLEAVERSAMPLISTBADSTG</p> <p><b>Meaning:</b> Program error. XCF could not access the control parameter list.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• Ensure that the correct parameter list storage area was specified.</li> <li>• If your program is running in AR mode: <ul style="list-style-type: none"> <li>– Ensure that you specified SYSSTATE ASCENV=AR before issuing the IXCLEAVE macro.</li> <li>– Ensure that the parameter list ALET is correct.</li> </ul> </li> <li>• Ensure that the parameter list storage area was not inadvertently freed by your program.</li> </ul>
08	110	<p><b>Equate Symbol:</b> IXCLEAVERSNUSTATEBADSTG</p> <p><b>Meaning:</b> Program error. XCF could not access the USTATE value.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• Ensure that the correct USTATE address was specified.</li> <li>• If your program is running in AR mode: <ul style="list-style-type: none"> <li>– Ensure that you specified SYSSTATE ASCENV=AR before issuing the IXCLEAVE macro.</li> <li>– Ensure that the USTATE ALET is correct.</li> </ul> </li> <li>• Ensure that the USTATE storage area was not inadvertently freed by your program.</li> </ul>
08	114	<p><b>Equate Symbol:</b> IXCLEAVERSUSLENBADVALUE</p> <p><b>Meaning:</b> Program error. The USLEN value is less than 1 or greater than 32.</p> <p><b>Action:</b> Correct the USLEN, and retry the request.</p>
08	118	<p><b>Equate Symbol:</b> IXCLEAVERSNOTTASKMODE</p> <p><b>Meaning:</b> Program error. The caller is not in task mode.</p> <p><b>Action:</b> Correct your program so that it issues IXCLEAVE only while in task mode.</p>
08	11C	<p><b>Equate Symbol:</b> IXCLEAVERSNOTENABLED</p> <p><b>Meaning:</b> Program error. The caller is not enabled.</p> <p><b>Action:</b> Correct your program so that it issues IXCLEAVE only while enabled.</p>

Table 8. Return and Reason Codes for the IXCLEAVE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
08	120	<b>Equate Symbol:</b> IXCLEAVERSNNPRIMARYNOTHOME <b>Meaning:</b> Program error. The primary address space is not equal to the home address space. <b>Action:</b> Correct your program so that it does not use IXCLEAVE while in cross memory mode. You might want to pass this restriction on to your caller when you are unsure of the environment your caller may have been in.
0C	18	<b>Equate Symbol:</b> IXCLEAVERSNTASKABENDED <b>Meaning:</b> Environmental error. While the issuing task was suspended for XCF processing, the task was abended (i.e. another unit of work attempted to abnormally terminate this task). The state of the IXCLEAVE request is unpredictable. <b>Action:</b> Determine why another unit of work has decided to abnormally terminate this task.
10	None.	<b>Meaning:</b> System error. XCF processing failed. <b>Action:</b> Retry the request at least once. If the problem persists, record the return code, and supply it to the appropriate IBM support personnel.

## Example

A member places itself in the not-defined state. XCF is to place the value X'11' in the user state field and store the return code and reason code into the variables RETURN and REASON. The code is as follows:

IXCLEAVE MEMTOKEN=TOKEN1, USTATE=STATE, USLEN=LEN, RETCODE=RETURN, RSNCODE=REASON, MF=S			X
TOKEN1	DS	CL8	TOKEN OF MEMBER TO BE PLACED IN NOT-DEFINED STATE X
RETURN	DS	1F	RETURN CODE
REASON	DS	1F	REASON CODE
STATE	DC	X'11'	USER STATE VALUE
LEN	DC	F'1'	LENGTH OF USER STATE DATA

You can obtain the member token from the QUAMTKN field in the area returned by IXCJOIN or IXCQUERY, and mapped by the IXCYQUAA mapping macro.

## Chapter 10. IXCMG – Obtain Tuning and Capacity Planning Data

### Description

The IXCMG macro provides a cross-system coupling facility (XCF) installation with information that can help system programmers plan tuning activities and capacity requirements for the sysplex.

The data that IXCMG generates consists of one header, followed by zero or more data records. The IXCYAMDA mapping macro maps the data that IXCMG returns, including the header record.

The header information returned in the DATAAREA is mapped differently depending on the IXCMG GATHERFROM specification. If GATHERFROM is not specified, AMDAREA is used. For GATHERFROM=LOCAL and GATHERFROM=TOKEN, AMDAGFD is used. For GATHERFROM=OTHER, AMDAGFO is used.

The GATHERFROM parameter specifies which system you want to gather the information from:

- GATHERFROM=LOCAL, the default, is used to request the gathering of data from the local system.
- GATHERFROM=OTHER is used to request the gathering of data from some other system in the sysplex.
- GATHERFROM=TOKEN is used to obtain the result of a previous IXCMG GATHERFROM=OTHER request.

The TYPE parameter specifies which of the five types of data records you can request:

- TYPE=PATH returns one record for each signalling path that XCF is using.
- TYPE=MSGPEND returns one record for each pending or delayed message.
- TYPE=SYSTEM returns records summarizing message traffic for the system.
- TYPE=SRCDST returns counts of messages sent and received by members in the system.
- TYPE=MEMBER returns one record for each active member on the target system.

TYPE=ALL, the default, returns all possible types of data records.

**Note:** For AMDALEVEL 0 requests, ALL is equivalent to having specified PATH, MSGPEND, SYSTEM, and SRCDST for TYPE. For AMDALEVEL > 0 requests, ALL is equivalent to having specified PATH, MSGPEND, SYSTEM, SRCDST, and MEMBER for TYPE.

Two required parameters, DATAAREA and DATALEN, specify the area where IXCMG is to return the data.

### Environment

The following are the environment requirements for the caller.

Environment	Environment requirement
Minimum authorization:	Supervisor state or PKM allowing key 0 - 7
Dispatchable unit mode:	Task or SRB
Cross memory mode:	Any PASN, any HASN, any SASN
AMODE:	31-bit
ASC mode:	Primary or access register (AR)
Interrupt status:	Enabled or disabled for I/O and external interrupts
Locks:	Holds either no locks or only the CPU lock

Environment	Environment requirement
Control parameters:	Must be in the primary address space or addressable through a public entry on the caller's dispatchable unit access list (DU-AL)

## Programming Requirements

---

If the program is in AR mode, issue the SYSSTATE ASCENV=AR macro before IXCMG. SYSSTATE ASCENV=AR tells the system to generate code appropriate for AR mode.

Include the mapping macro IXCYAMDA in your program to map the data returned in DATAAREA.

## Restrictions

---

This macro must be issued from a nonswappable primary address space.

## Input Register Information

---

Before issuing the IXCMG macro, the caller does not have to place any information into any register unless using it in register notation for a particular parameter, or using it as a base register.

## Output Register Information

---

When control returns to the caller, the general purpose registers (GPRs) contain:

### Register

#### Contents

#### 0

If GPR 15 contains a zero or X'10', GPR 0 is used as a work register by the macro; otherwise, GPR 0 contains a reason code.

#### 1

Used as a work register by the system.

#### 2-13

Unchanged.

#### 14

Used as a work register by the system.

#### 15

Return code.

When control returns to the caller, the access registers (ARs) contain:

### Register

#### Contents

#### 0-1

Used as work registers by the system

#### 2-13

Unchanged

#### 14-15

Used as work registers by the system

Some callers depend on register contents remaining the same before and after issuing a service. If the system changes the contents of registers on which the caller depends, the caller must save them before issuing the service, and restore them after the system returns control.

## Performance Implications

---

The use of the IXCMG macro might degrade the performance of the XCF signalling service.

## Understanding IXCMG version support

---

The IXCMG macro supports versions in the range of 0 - 3.

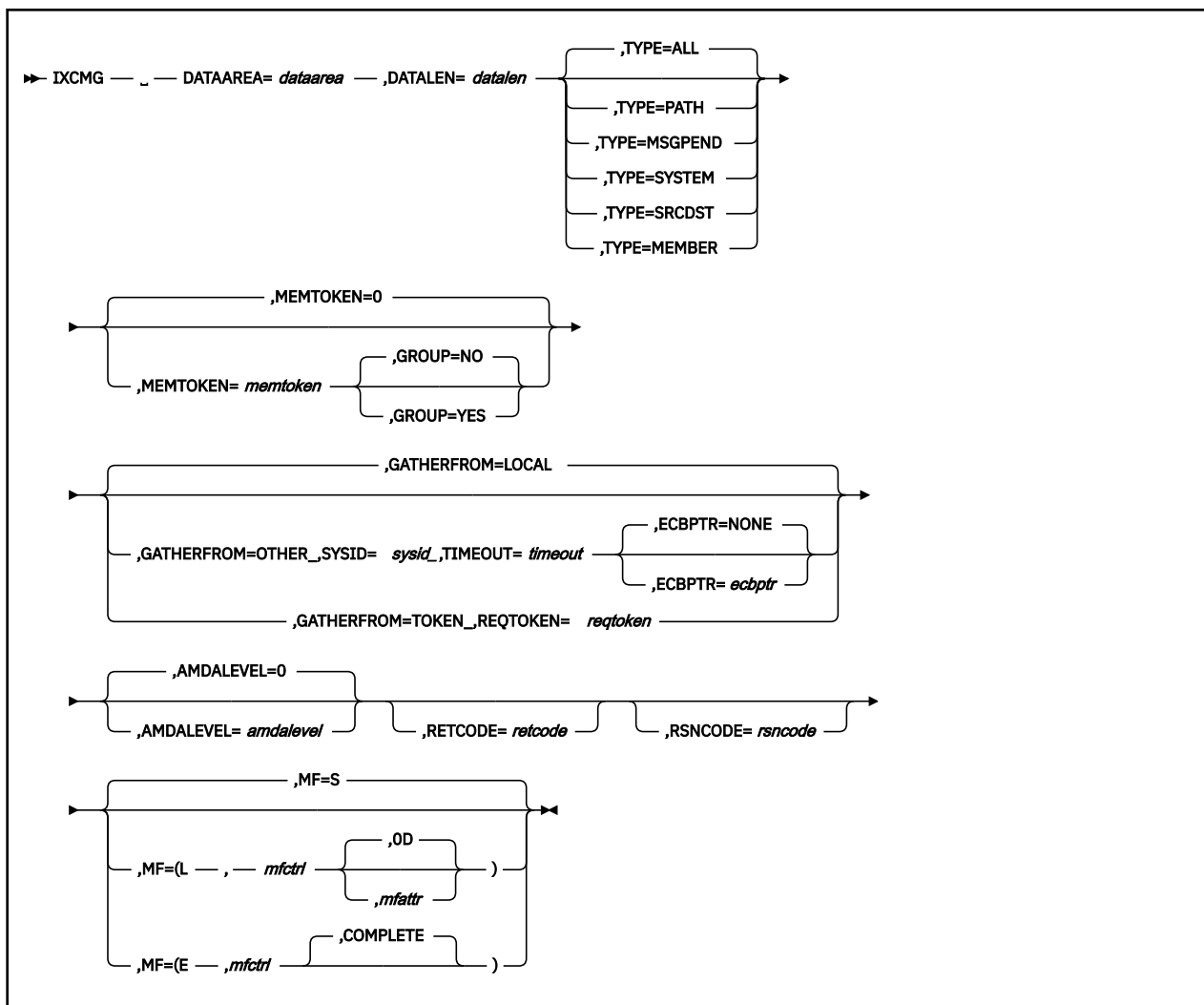
- Keywords not specifically noted here are supported by all versions starting with version 0 and higher of the IXCMG macro.
- The following keywords are supported by all versions starting with version 1 and higher of the IXCMG macro.
  - AMDALEVEL
  - MEMTOKEN
  - TYPE=MEMBER
- The following keywords are supported by all versions starting with version 2 and higher of the IXCMG macro.
  - ECBPTR
  - GATHERFROM
  - GROUP
  - REQTOKEN
  - SYSID
  - TIMEOUT
- The following keyword is supported by all versions starting with version 3 and higher of the IXCMG macro.
  - CLASSNAME

Specify the version of the parameter list that you want generated with the PLISTVER keyword. See [Chapter 2, “Specifying a Macro Version Number,” on page 5](#) for considerations when specifying the version of the parameter list with PLISTVER.

## Syntax Diagram

---

The syntax of the IXCMG macro is as follows:



**Note:** To specify more than one TYPE, enclose the choices in parenthesis and separate them by commas. For example, TYPE=(PATH,SYSTEM).

## Parameter Descriptions

The parameter descriptions are listed in alphabetical order. Default values are underlined:

### **,AMDALEVEL=0**

### **,AMDALEVEL=amdalevel**

Use this input parameter to specify the level of the IXCYAMDA record mappings that the system returns in the answer area. Valid values are 0, 1, 2, 3, and 4.

- A value of 0 indicates that base level IXCYAMDA records will be returned.
- A value of 1 indicates that level-1 IXCYAMDA records will be returned.
- A value of 2 indicates that level-2 IXCYAMDA records will be returned.
- A value of 3 indicates that level-3 IXCYAMDA records will be returned.
- A value of 4 indicates that level-4 IXCYAMDA records will be returned.

### **DATAAREA=dataarea**

Use this output area to identify the area in which IXCMG is to return the data you request.

The data area must be in fixed or disabled reference (DREF) storage. It must be in the primary address space or in an address/data space that is addressable through a public entry on the caller's dispatchable unit access list (DU-AL).



**To Code:** Specify the name of a fullword field (or register 2-12 containing the address of the field) that identifies the data area.

**,DATALEN=*datalen***

Use this input parameter to specify the length of the area you provided on the DATAAREA parameter. The length should be long enough to accommodate the IXCYAMDA mapping of the data area.

**To Code:** Specify the name of a fullword field (or register 2-12 containing the address of the field) that contains the length of the data area.

**,ECBPTR=NONE**

**,ECBPTR=*ecbptr***

Use this input parameter to specify the address of an ECB to be posted either when the results become available or when the request is discarded. If ECBPTR is not specified or ECBPTR=NONE is specified, no ECB is posted.

This is an optional parameter and ECBPTR=NONE is the default.

**To Code:** Specify the RS-type name or address (using a register from 2-12) of the 4 byte field containing the ECB address.

**,GATHERFROM=LOCAL**

**,GATHERFROM=OTHER**

**,GATHERFROM=TOKEN**

Use this input parameter to specify from which system you want to gather the information.

- GATHERFROM=LOCAL, the default, is used to request the gathering of data from the local system. Note that the contents of the DATAAREA is different depending on whether GATHERFROM=LOCAL is explicitly coded or is processed by default. If not explicitly coded, the DATAAREA will contain the header AMDAREA followed by the requested data records. If GATHERFROM=LOCAL is explicitly coded, the header information is mapped by AMDAGFD and there will be two new records (AMDGLI) provided.
- GATHERFROM=OTHER is used to request the gathering of data from some other system in the sysplex. The requester provides the XCF system ID of the system from which the data is to be gathered and a timeout value indicating how long they are willing to wait for the results. An optional ECB can be provided if the requester wants XCF to post the user when the results arrive (or when the request times out). If not posted, the requester is expected to poll for the results. If accepted, the output DATAAREA contains a request token (AMDAGFO\_REQTOKEN) that is used to obtain the results of the asynchronous data gathering. Use this token as input to a subsequent IXCMG GATHERFROM=TOKEN request to retrieve the results. If the user does not gather the results before the timeout, or if the target system is removed from the sysplex, XCF discards the request and its results.
- GATHERFROM=TOKEN is used to obtain the result of a previous IXCMG GATHERFROM=OTHER request. The DATAAREA must be large enough for the AMDAGFD header. The requester provides the appropriate REQTOKEN to identify which results are to be obtained. The expected answer is one of the following:
  - The results were copied into the indicated DATAAREA.
  - The indicated DATAAREA is too small to hold all of the results. Try again with a bigger DATAAREA.
  - Expected results have not yet arrived. Try again later.
  - No such request exists anymore

The TYPE, MEMTOKEN, GROUP, and AMDALEVEL parameters should not be coded when issuing a GATHERFROM=TOKEN request to retrieve the results.

This is an optional parameter and GATHERFROM=LOCAL is the default.

**,GROUP=NO****,GROUP=YES**

Use this input parameter to specify whether information is to be returned only for the indicated member (GROUP=NO) or for all active members of the member's group (GROUP=YES) on the system where the data is gathered. GROUP is only used when the MEMTOKEN value is nonzero.

This is an optional parameter and GROUP=NO is the default.

**,MEMTOKEN=0****,MEMTOKEN=memtoken**

Use this input parameter to specify the MEMTOKEN of the member whose MEMBER data is to be gather. MEMTOKEN is only used when gathering MEMBER data. It is ignored for all other data options. If the indicated member does not reside on the local system, no MEMBER data will be returned.

**To Code:** Specify the name of a 64-bit field (or register 2-12 containing the address of the field) that contains the MEMTOKEN of the member.

**,MF=S****,MF=(L,mfctrl)****,MF=(L,mfctrl,mfattr)****,MF=(L,mfctrl,0D)****,MF=(M,mfctrl)****,MF=(M,mfctrl,COMPLETE)****,MF=(M,mfctrl,NOCHECK)****,MF=(E,mfctrl)****,MF=(E,mfctrl,COMPLETE)****,MF=(E,mfctrl,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

**,mfctrl**

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

**,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

**,COMPLETE****,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

**COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if `SMILE=var` were an optional parameter and the default is `SMILE=NO_SMILE` then it would not be documented. However, if the default was `SMILE=-`, then it would be documented because a value would be the default.

#### **NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

#### **,REQTOKEN=reqtoken**

Use this input parameter to specify the request token that identifies the request whose results are to be collected.

**To Code:** Specify the RS-type name or address (using a register from 2-12) of the 16 character field that contains the request token.

#### **,RETCODE=retcode**

Use this output parameter to specify the location in which the system is to copy the return code from GPR 15.

**To Code:** Specify the name of a fullword field (or register 2-12 containing the address of the field) to contain the return code.

#### **,RSNCODE=rsncode**

Use this output parameter to specify the location in which the system is to copy the reason code from GPR 0.

**To Code:** Specify the name of a fullword field (or register 2-12 containing the address of the field) to contain the reason code.

#### **,SYSID=sysid**

Use this input parameter to specify the XCF system ID of the system from which the data is to be gathered.

**To Code:** Specify the RS-type name or address (using a register from 2-12) of the fullword field containing the target system ID.

#### **,TIMEOUT=timeout**

Use this input parameter to specify the maximum number of seconds that the system allows the request to persist. The specified value must be between 1 and 120, inclusive.

**To Code:** Specify the RS-type name or address (using a register from 2-12) of the halfword field containing the timeout value.

#### **,TYPE=ALL**

#### **,TYPE=PATH**

#### **,TYPE=MSGPEND**

#### **,TYPE=SYSTEM**

#### **,TYPE=SRCDST**

#### **,TYPE=MEMBER**

Use this input parameter to specify which types of data records XCF is to return. `TYPE=ALL` returns all types of data records. If you do not specify the `TYPE` parameter, you receive all types of data records. You may specify more than one type of data record with this parameter. For example, you could code: `TYPE=(PATH,MSGPEND)`.

The data that IXCMG generates consists of a header, followed by zero or more data records. The types of data records are as follows:

- `TYPE=PATH` returns one record for each signalling path that XCF is using. The `AMDPATH` structure in `IXCYAMDA` maps this record. It includes such information as, for each outbound signalling path, the number of times XCF tried to send a signal across the path.

- TYPE=MSGPEND returns one record for each pending message. The AMDMPEND structure maps this record. It includes such information as, for each outbound signal queued for data transfer, the member token, ASID, and length of the message.
- TYPE=SYSTEM returns records that describe the message traffic associated with the system on which you issue IXCMG. The AMDSYS structure maps this record. It includes such information as the total number of times the system refused message requests in each transport class because buffer space was not available.
- TYPE=SRCDST returns one record for each active member in the sysplex. The AMDSD structure maps this record. It includes such information as the number of messages the member has sent.
- TYPE=MEMBER returns one record for each member that resides on the local system. The AMDMEM structure maps this information. The MEMTOKEN keyword can be specified to limit the data to a specific member.

## ABEND Codes

None.

## Return and Reason Codes

When the IXCMG macro returns control to your program:

- GPR 15 (and *retcode*, if you coded RETCODE) contains a return code.
- GPR 0 (and *rsncode*, if you coded RSNCODE) contains a reason code, if applicable.

Macro IXCYCON provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

**0**

IXCRETCODEOK

**4**

IXCRETCODEWARNING

**8**

IXCRETCODEPARMERROR

**C**

IXCRETCODEENVERROR

**10**

IXCRETCODECOMPERROR

The following table identifies the hexadecimal return and reason codes and the equate symbol associated with each reason code.

Table 9. Return and Reason Codes for the IXCMG Macro		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0	None.	<b>Meaning:</b> IXCMG completed successfully, and XCF provided all the data that the caller requested. <b>Action:</b> None.

Table 9. Return and Reason Codes for the IXCMG Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	4	<p><b>Equate Symbol:</b> IXCMGRSNSTILLMOREDATA</p> <p><b>Meaning:</b> Program error. IXCMG completed successfully, and XCF provided some data; however, DATAAREA is too small to contain all the requested data.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• If the required information is contained in the DATAAREA, no further action needs to be taken.</li> <li>• If the required information is not contained in the DATAAREA, obtain a larger DATAAREA and reissue IXCMG. The AMDATLEN field, as defined by the XCF accounting and measurement data area mapping macro (IXCYAMDA), contains the DATAAREA size that is required to contain all of the information that would have been returned. However, because IXCMG returns only a snapshot of the current status, it is possible that the AMDATLEN might be too small on the next invocation.</li> <li>• Ensure that you specified the correct length for the data area.</li> <li>• Ensure that the parameter list was not inadvertently overlaid.</li> <li>• Ensure that you specified the correct data area address and ALET.</li> <li>• If your program is not running in primary ASC mode, ensure that you specified SYSSTATE ASCENV=AR.</li> </ul>
4	8	<p><b>Equate Symbol:</b> IXCMGRSNRESULTSPENDING</p> <p><b>Meaning:</b> For GATHERFROM=TOKEN, the expected results have not yet arrived. DATAAREA was not updated.</p> <p><b>Action:</b> Try again later.</p>
4	10	<p><b>Equate Symbol:</b> IXCMGRSNCHECKRESULTS</p> <p><b>Meaning:</b> GATHERFROM=OTHER might have failed.</p> <ul style="list-style-type: none"> <li>• For GATHERFROM=TOKEN, the request completed successfully, but the data gathering performed by the target system completed with a nonzero return code.</li> <li>• For GATHERFROM=OTHER, the local system determined that the target system could not process the request.</li> </ul> <p><b>Action:</b> Check the returned header for the return and reason codes that explain why the target system failed to process the request.</p>

Table 9. Return and Reason Codes for the IXCMG Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	14	<p><b>Equate Symbol:</b> IXCMGRSNDATAAREATOOSMALL</p> <p><b>Meaning:</b> Program error. DATAAREA field is too small to contain the header (AMDAREA, AMDAGFD or AMDAGFO).</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• Provide a data area that is large enough to contain at least the header information as mapped by the field (AMDAREA, AMDAGFD or AMDAGFO) of the XCF accounting and measurement data area mapping macro (IXCYAMDA).</li> <li>• Ensure that you specified the correct length for the data area.</li> <li>• Ensure that the parameter list was not inadvertently overlaid.</li> <li>• Ensure that you specified the correct data area address and ALET.</li> <li>• If your program is running in AR mode, ensure that you specified SYSSTATE ASCENV=AR.</li> </ul>
8	18	<p><b>Equate Symbol:</b> IXCMGRSNDATAAREABADSTG</p> <p><b>Meaning:</b> Program error. XCF could not access DATAAREA.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• Ensure that the correct DATAAREA address, ALET, and length were specified.</li> <li>• Ensure that the parameter list storage area was not inadvertently freed by your program.</li> <li>• If your program is running in AR mode, ensure that you specified SYSSTATE ASCENV=AR before issuing the IXCMG macro.</li> </ul>
8	1C	<p><b>Equate Symbol:</b> IXCMGRSNBADALET</p> <p><b>Meaning:</b> Program error. The ALET that qualifies the address specified on DATAAREA is neither zero nor associated with a valid public entry on the caller's DU-AL.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• If your program is running in AR mode, you specified SYSSTATE ASCENV=AR before issuing the IXCMG macro.</li> <li>• The ALET for the parameter list is a valid public entry on the DU-AL or is zero (primary address space ALET).</li> </ul>
8	40	<p><b>Equate Symbol:</b> IXCMGRSNPLISTRSDNOTVALID</p> <p><b>Meaning:</b> Program error or environmental error. A reserved field in the control parameter list is not zero. Your program might have inadvertently written over an area in the control parameter list.</p> <p><b>Action:</b> Check to see if your program inadvertently overlaid the parameter list storage, and that it was assembled with the correct macro library for the release of MVS your program is running on.</p>

Table 9. Return and Reason Codes for the IXCMG Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	100	<p><b>Equate Symbol:</b> IXCMGRSNPLISTBADALET</p> <p><b>Meaning:</b> Program error. Your program is not running in primary ASC mode, and the ALET that qualifies the address of the control parameter list is neither zero nor associated with a valid public entry on the caller's DU-AL.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• Your program is not intended to run in primary ASC mode.</li> <li>• You specified SYSSTATE ASCENV=AR before issuing the IXCMG macro.</li> <li>• The ALET for the parameter list is a valid public entry on the DU-AL or is zero (primary address space ALET).</li> </ul>
8	104	<p><b>Equate Symbol:</b> IXCMGRSNPLISTVERSIONNOTVALID</p> <p><b>Meaning:</b> Program error or environmental error. The version number in the control parameter list is not valid.</p> <p><b>Action:</b> Check to see if your program inadvertently overlaid the parameter list storage, and that it was assembled with the correct macro library for the release of MVS your program is running on.</p>
8	108	<p><b>Equate Symbol:</b> IXCMGRSNPLISTBADFUNCTION</p> <p><b>Meaning:</b> Program error or environmental error. The function code in the control parameter list is not valid.</p> <p><b>Action:</b> Check to see if your program inadvertently overlaid the parameter list storage, and that it was assembled with the correct macro library for the release of MVS your program is running on.</p>
8	10C	<p><b>Equate Symbol:</b> IXCMGRSNPLISTBADSTG</p> <p><b>Meaning:</b> Program error. XCF could not access the control parameter list.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• Ensure that the correct parameter list storage area was specified.</li> <li>• If your program is running in AR mode: <ul style="list-style-type: none"> <li>– Ensure that you specified SYSSTATE ASCENV=AR before issuing the IXCMG macro.</li> <li>– Ensure that the parameter list ALET is correct.</li> </ul> </li> <li>• Ensure that the parameter list storage area was not inadvertently freed by your program.</li> </ul>

Table 9. Return and Reason Codes for the IXCMG Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	110	<p><b>Equate Symbol:</b> IXCMGRSNPLISTBADAMDALEVEL</p> <p><b>Meaning:</b> Program error. AMDALEVEL value is not valid.</p> <p><b>Action:</b> Correct the AMDALEVEL value in the request and try again.</p> <p>The AMDAGFO_MaxAmdaLevel field in the DATAAREA indicates the highest AMDALEVEL supported by the target system.</p> <p>AMDALEVEL must be zero for GATHERFROM=OTHER requests.</p>
8	114	<p><b>Equate Symbol:</b> IXCMGRSNPLISTBADMEMTOKEN</p> <p><b>Meaning:</b> Program error. MEMTOKEN value is not a valid member token.</p> <p><b>Action:</b> Correct the MEMTOKEN value in the request and try again.</p> <p>MEMTOKEN must be zero for GATHERFROM=OTHER requests.</p>
8	118	<p><b>Equate Symbol:</b> IXCMGRSNPLISTBADSYSID</p> <p><b>Meaning:</b> Program error. SYSID value is not valid.</p> <p><b>Action:</b> Correct the SYSID value in the request and try again.</p> <p>SYSID must be zero for GATHERFROM=LOCAL and GATHERFROM=OTHER requests.</p>
8	11C	<p><b>Equate Symbol:</b> IXCMGRSNNOTENABLED</p> <p><b>Meaning:</b> Program error. The program must be running enabled when GATHERFROM=TOKEN or when GATHERFROM=OTHER and ECBPTR=ecbptr is requested.</p> <p><b>Action:</b> Make sure the program is running enabled when making the request.</p>
8	120	<p><b>Equate Symbol:</b> IXCMGRSNPLISTBADREQTOKEN</p> <p><b>Meaning:</b> Program error. REQTOKEN value is not a valid token.</p> <p><b>Action:</b> Correct the REQTOKEN value in the request and try again.</p> <p>REQTOKEN must be zero for GATHERFROM=LOCAL requests.</p>
8	124	<p><b>Equate Symbol:</b> IXCMGRSNPLISTBADTIMEOUT</p> <p><b>Meaning:</b> Program error. TIMEOUT value is not valid.</p> <p><b>Action:</b> Correct the TIMEOUT value in the request and try again.</p> <p>The TIMEOUT value must be in the range of 1 to 120, inclusive. Nonzero TIMEOUT value should only be used for GATHERFROM=OTHER requests.</p>



Table 9. Return and Reason Codes for the IXCMG Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	128	<b>Equate Symbol:</b> IXCMGRSNECBBADSTG <b>Meaning:</b> Program error. Unable to access ECB storage. <b>Action:</b> Make sure the ECB storage pointed by ECBPTR=ecbptr is valid.
8	12C	<b>Equate Symbol:</b> IXCMGRSNLOCKHELD <b>Meaning:</b> Environmental error. Caller holds locks when making the request. The program must be running with no lock held when requesting GATHERFROM=TOKEN or GATHERFROM=OTHER and ECBPTR=ecbptr. <b>Action:</b> Make sure the program does not hold any locks when making the request.
8	130	<b>Equate Symbol:</b> IXCMGRSNPLISTBADGROUP <b>Meaning:</b> Program error. GROUP value is not valid. <b>Action:</b> Correct the GROUP value in the request and try again. GROUP value must be NO for GATHERFROM=TOKEN requests.
8	134	<b>Equate Symbol:</b> IXCMGRSNPLISTBADECBPTR <b>Meaning:</b> Program error. ECBPTR value is not valid. <b>Action:</b> Correct the ECBPTR value in the request and try again. A nonzero ECBPTR value should only be used for GATHERFROM=OTHER requests.
8	138	<b>Equate Symbol:</b> IXCMGRSNPLISTBADTYPE <b>Meaning:</b> Program error. TYPE value is not valid. <b>Action:</b> Correct the TYPE value in the request and try again. The default value (TYPE=ALL) must be used for GATHERFROM=TOKEN requests.
C	4	<b>Equate Symbol:</b> IXCMGRSNNEEDSOFTWARE <b>Meaning:</b> Environmental error. The system is unable to process the request because it does not have the necessary software installed. For example, the target system does not support GATHERFROM=OTHER requests. <b>Action:</b> Make sure the system from which the data is to be gathered has the necessary level of software installed.

Table 9. Return and Reason Codes for the IXCMG Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	8	<p><b>Equate Symbol:</b> IXCMGRSNNEEDRESOURCES</p> <p><b>Meaning:</b> Environmental error. The request could not be processed because the system was unable to obtain the resources needed to process the request.</p> <p><b>Action:</b> Make sure the system from which the data is to be gathered has enough resources to process the request. Be aware that the amount of data collected when TYPE=ALL is specified could be quite large. You might want to use separate requests to collect different TYPEs of data instead.</p>
C	C	<p><b>Equate Symbol:</b> IXCMGRSNSYSTEMNOTACTIVE</p> <p><b>Meaning:</b> Environmental error. The system that is to provide the requested data is unable to do so because it is not active in the sysplex.</p> <ul style="list-style-type: none"> <li>For GATHERFROM=OTHER request, the local system was able to make this determination before the request was accepted for processing.</li> <li>For GATHERFROM=TOKEN request, the local system made this determination after the GATHERFROM=OTHER request was accepted but the system became inactive before the data could be made available to the local system.</li> </ul> <p><b>Action:</b> As appropriate, try again after the target system becomes active to obtain the desired data.</p>
C	10	<p><b>Equate Symbol:</b> IXCMGRSNSYSTEMNOTREADY</p> <p><b>Meaning:</b> Environmental error. For GATHERFROM=OTHER, the local system and the target system have not yet completed the setup needed to process the request.</p> <p><b>Action:</b> Try again later.</p>
C	14	<p><b>Equate Symbol:</b> IXCMGRSNNEEDNEWREQUEST</p> <p><b>Meaning:</b> Environmental error. For GATHERFROM=TOKEN, the request indicated by REQTOKEN no longer exists.</p> <p><b>Action:</b> As appropriate, issue a new GATHERFROM=OTHER request to obtain the desired data.</p> <p>Because this failure likely implies that the request timed out, consider increasing the TIMEOUT specification for the new request.</p> <p>If you have coded the maximum possible timeout, there might be system problems that prevent your GATHERFROM=TOKEN request from being processed in a timely manner, or that prevent XCF from being able to collect the data from the target system in a timely manner.</p>

Table 9. Return and Reason Codes for the IXCMG Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
10	None.	<b>Meaning:</b> System error. XCF processing failed. <b>Action:</b> Retry the request one or more times. If the problem persists, record the return code, and supply it to the appropriate IBM support personnel.

## Example

*Operation:* Obtain tuning and capacity planning information on the use of XCF signalling paths. Register 2 points to the area where XCF is to return member information. XCF is to store the return code and reason code into the variables RETURN and REASON. The code is as follows:

```

      LA      R2,MYAREA      LOAD ADDRESS OF DATA AREA
      IXCMG   DATAAREA=(R2),DATALEN=AREALEN,TYPE=PATH,          X
             RETCODE=RETURN,RSNCODE=REASON,MF=S
MYAREA DS    CL4096        AREA TO PLACE MEASUREMENT DATA
RETURN DS    1F            RETURN CODE
REASON DS    1F            REASON CODE
AREALEN DC    F'4096'      LENGTH OF THE DATA AREA WHERE      X
                           MEASUREMENT INFORMATION IS PLACED

```



## Chapter 11. IXCMOD – Modify Status-Checking Interval

### Description

The IXCMOD macro allows an active cross-system coupling facility (XCF) member to change its status-checking interval. The member must have status monitoring established.

A typical use of IXCMOD is for the caller to change its interval in response to a change in the system's environment, such as a change in the failure detection interval. See *z/OS MVS Programming: Sysplex Services Guide* for further information.

When a member changes its status-checking interval through IXCMOD, XCF notifies other active members of the group through their group user-routines.

### Environment

The following are the environment requirements for the caller.

Environment	Environment requirement
Minimum authorization:	Supervisor state or PKM allowing key 0 - 7
Dispatchable unit mode:	Task
Cross memory mode:	Any PASN, any HASN, any SASN. The primary address space must be the same as the primary address space of the caller of the IXCJOIN that placed the member in the active state.
AMODE:	31-bit
ASC mode:	Primary or access register (AR)
Interrupt status:	Enabled for external and I/O interrupts
Locks:	No locks held
Control parameters:	Must be in the primary address space or be in an address/data space addressable through a public entry in the dispatchable unit access list (DU-AL)

### Programming Requirements

If the program is in AR mode, issue the SYSSTATE ASCENV=AR macro before IXCMOD. SYSSTATE ASCENV=AR tells the system to generate code appropriate for AR mode.

### Restrictions

- The member must have status monitoring established.
- The caller can have no enabled unlocked task (EUT) FRRs established.

### Input Register Information

Before issuing the IXCMOD macro, the caller does not have to place any information into any register unless using it in register notation for a particular parameter, or using it as a base register.

## Output Register Information

When control returns to the caller, the general purpose registers (GPRs) contain:

### Register Contents

**0**

If GPR 15 contains a zero or X'10', GPR 0 is used as a work register by the system; otherwise, GPR 0 contains a reason code.

**1**

Used as a work register by the system.

**2-13**

Unchanged.

**14**

Used as a work register by the system.

**15**

Return code.

When control returns to the caller, the access registers (ARs) contain:

### Register Contents

**0-1**

Used as work registers by the system

**2-13**

Unchanged

**14-15**

Used as work registers by the system

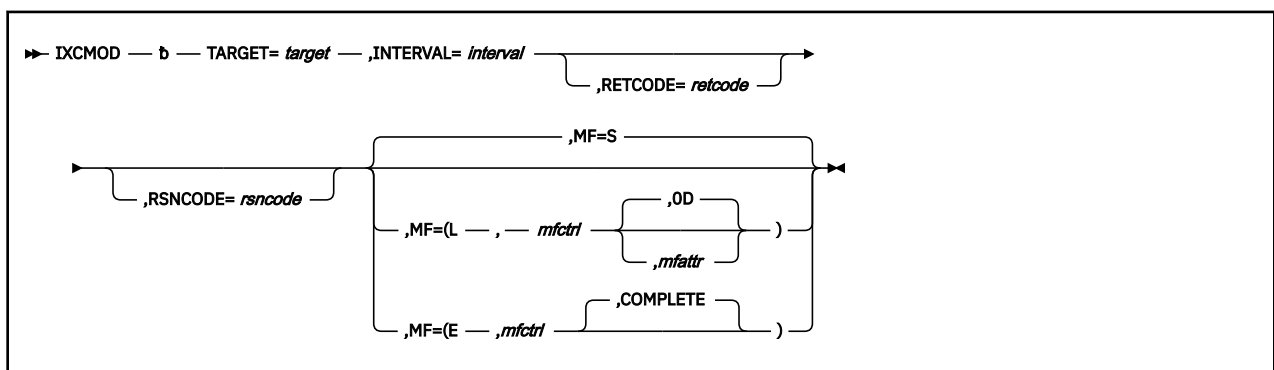
Some callers depend on register contents remaining the same before and after issuing a service. If the system changes the contents of registers on which the caller depends, the caller must save them before issuing the service, and restore them after the system returns control.

## Performance Implications

When a member fails to update its status field within the member's status-checking interval (the INTERVAL parameter on the IXCJOIN or the IXCMOD macro), XCF schedules the member's status user-routine. By increasing the interval value, you decrease the number of times the status user-routine receives control, thereby reducing system overhead.

## Syntax Diagram

The syntax of the IXCMOD macro is as follows:



## Parameter Descriptions

The parameter descriptions are listed in alphabetical order. Default values are underlined:

### **,INTERVAL=*interval***

Use this input parameter to specify the status-checking interval that is to replace the existing status-checking interval. IXCJOIN set this interval on the INTERVAL parameter, or IXCMOD reset the interval in a previous invocation. Specify the interval in full seconds; that is, the value must be greater than zero and must be a multiple of 100.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 4-byte field that contains the new interval.

### **,MF=S**

### **,MF=(L,*mfctrl*)**

### **,MF=(L,*mfctrl*,*mfattr*)**

### **,MF=(L,*mfctrl*,0D)**

### **,MF=(M,*mfctrl*)**

### **,MF=(M,*mfctrl*,COMPLETE)**

### **,MF=(M,*mfctrl*,NOCHECK)**

### **,MF=(E,*mfctrl*)**

### **,MF=(E,*mfctrl*,COMPLETE)**

### **,MF=(E,*mfctrl*,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

### **,*mfctrl***

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

### **,*mfattr***

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

### **,COMPLETE**

### **,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

### **COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if SMILE=*var* were an optional parameter and the default is SMILE=NO\_SMILE then it would not be documented. However, if the default was SMILE=:-), then it would be documented because a value would be the default.

**NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

**,RETCODE=retcode**

Use this output parameter to specify the location in which the system is to copy the return code from GPR 15.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 4-byte field that will contain the return code when this request completes.

**,RSNCODE=rsncode**

Use this output parameter to specify the location in which the system is to copy the reason code from GPR 0.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 4-byte field that will contain the reason code (if any) when the request completes.

**TARGET=target**

Use this input parameter to specify the token of the target member. IXCJOIN provided this token when it activated the member.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-byte field that contains the member token.

## ABEND Codes

---

None.

## Return and Reason Codes

---

When the IXCMOD macro returns control to your program:

- GPR 15 (and *retcode*, if you coded RETCODE) contains a return code.
- GPR 0 (and *rsncode*, if you coded RSNCODE) contains a reason code, if applicable.

Macro IXCYCON provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

**0**

IXCRETCODEOK

**4**

IXCRETCODEWARNING

**8**

IXCRETCODEPARMERROR

**C**

IXCRETCODEENVERROR

**10**

IXCRETCODECOMPERROR

The following table identifies the hexadecimal return and reason codes and the equate symbol associated with each reason code.



Table 10. Return and Reason Codes for the IXCMOD Macro

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
00	None.	<b>Meaning:</b> IXCMOD completed successfully; XCF changed the value of the status-checking interval. <b>Action:</b> None.
08	04	<b>Equate Symbol:</b> IXCMODRSNNOTACTIVE <b>Meaning:</b> Program error. The target member is not active. <b>Action:</b> Take one or more of the following actions: <ul style="list-style-type: none"> <li>• Ensure that the correct MEMTOKEN was specified.</li> <li>• You might want your program to take some action based on the fact the member is no longer active.</li> </ul>
08	08	<b>Equate Symbol:</b> IXCMODRSNNOSTATUSMON <b>Meaning:</b> Program error. The IXCJOIN that activated the target member did not request status monitoring, or XCF has stopped monitoring the member. <b>Action:</b> Take one or more of the following actions: <ul style="list-style-type: none"> <li>• Ensure that the correct MEMTOKEN was specified. Verify the address and ALET (if appropriate).</li> <li>• Correct your application so that it joins the member with status monitoring.</li> </ul>
08	0C	<b>Equate Symbol:</b> IXCMODRSNINTERVALBAD <b>Meaning:</b> Program error. The status-checking interval value is either zero or not a multiple of 100. <b>Action:</b> Take one or more of the following actions: <ul style="list-style-type: none"> <li>• Ensure that the correct status-checking interval was specified. Verify the address and ALET (if appropriate).</li> <li>• Correct your program and retry the request.</li> </ul>
08	10	<b>Equate Symbol:</b> IXCMODRSNINAPPROPRIATECALLER <b>Meaning:</b> Program error. The primary address space is not the same as the primary address space of the issuer of the IXCJOIN that activated the target member. <b>Action:</b> Take one or more of the following actions: <ul style="list-style-type: none"> <li>• Ensure that the correct MEMTOKEN was specified. Verify the address and ALET (if appropriate).</li> <li>• Correct your program so that it issues IXCMOD only from the address space in which the IXCJOIN was issued for the member.</li> </ul>

Table 10. Return and Reason Codes for the IXCMOD Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
08	40	<p><b>Equate Symbol:</b> IXCMODRSNPLISTRSDNOTVALID</p> <p><b>Meaning:</b> Program error or environmental error. A reserved field in the control parameter list is not zero.</p> <p><b>Action:</b> Check to see if your program inadvertently overlaid the parameter list storage, and that it was assembled with the correct macro library for the release of MVS your program is running on.</p>
08	100	<p><b>Equate Symbol:</b> IXCMODRSNPLISTBADALET</p> <p><b>Meaning:</b> Program error. The ALET that qualifies the address of the control parameter list is neither zero nor associated with a valid public entry on the caller's DU-AL.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• If your program is running in AR mode, you specified SYSSTATE ASCENV=AR before issuing the IXCMOD macro.</li> <li>• The ALET for the parameter list is a valid entry on the DU-AL or is zero (primary address space ALET).</li> </ul>
08	104	<p><b>Equate Symbol:</b> IXCMODRSNPLISTVERSIONNOTVALID</p> <p><b>Meaning:</b> Program error or environmental error. The version number in the control parameter list is not valid.</p> <p><b>Action:</b> Check to see if your program inadvertently overlaid the parameter list storage, and that it was assembled with the correct macro library for the release of MVS your program is running on.</p>
08	108	<p><b>Equate Symbol:</b> IXCMODRSNPLISTBADFUNCTION</p> <p><b>Meaning:</b> Program error. The function code in the control parameter list is not valid.</p> <p><b>Action:</b> Check to see if your program inadvertently overlaid the parameter list storage.</p>
08	10C	<p><b>Equate Symbol:</b> IXCMODRSNPLISTBADSTG</p> <p><b>Meaning:</b> Program error. XCF could not access the control parameter list.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• Ensure that the correct parameter list storage area was specified.</li> <li>• If your program is running in AR mode: <ul style="list-style-type: none"> <li>– Ensure that you specified SYSSTATE ASCENV=AR before issuing the IXCCREAT macro.</li> <li>– Ensure that the parameter list ALET is correct.</li> </ul> </li> <li>• Ensure that the parameter list storage area was not inadvertently freed by your program.</li> </ul>

Table 10. Return and Reason Codes for the IXCMOD Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
08	118	<b>Equate Symbol:</b> IXCMODRSNNOTTASKMODE <b>Meaning:</b> Program error. The caller is not in task mode. <b>Action:</b> Correct your program so that it issues IXCMOD only while in task mode.
08	11C	<b>Equate Symbol:</b> IXCMODRSNNOTENABLED <b>Meaning:</b> Program error. The caller is not enabled. <b>Action:</b> Correct your program so it only calls IXCMOD while enabled.
0C	18	<b>Equate Symbol:</b> IXCMODRSNTASKABENDED <b>Meaning:</b> Environmental error. While the issuing task was suspended for XCF processing, the task was abended (ie. another unit of work attempted to abnormally terminate this task). The state of the IXCMOD request is unpredictable. <b>Action:</b> Determine why the task was being abended.
10	None.	<b>Meaning:</b> System error. XCF processing failed. <b>Action:</b> Retry the request one or more times. If the problem persists, record the return code, and supply it to the appropriate IBM support personnel.

## Example

*Operation:* A member changes its own status-checking interval to five seconds. XCF is to store the return code and reason code into the variables RETURN and REASON. The code is as follows:

```

IXCMOD  TARGET=TOKEN1,INTERVAL=INTER,          X
        RETCODE=RETURN,RSNCODE=REASON,MF=S
RETURN  DS    1F                                RETURN CODE
REASON  DS    1F                                REASON CODE
TOKEN1  DS    CL8                                TOKEN OF THE MEMBER TO BE      X
                                                MODIFIED
INTER   DC    F'500'                            NEW INTERVAL VALUE FOR THIS  X
                                                MEMBER

```

You can obtain the member token from the QUAMTKN field in the area returned by IXCJOIN or IXCQUERY, and mapped by the IXCYQUAA mapping macro.



## Chapter 12. IXCMMSGC – XCF Message Control

### Description

The IXCMMSGC macro allows a cross-system coupling facility (XCF) application to interact with the XCF signalling services to provide additional functions. The IXCMMSGC macro provides services that:

- Save a message for later processing
- Retrieve information about messages that have been saved or are pending completion
- Redeliver a message to the member that previously saved the message
- Cancel (discard) a pending message.
- Force a message to be considered complete
- Discard a message
- Release a client/server blocking receive request (IXCRECV), a client/server send message request (IXCSEND), or a message-out request (IXCMMSGOX) issued with the MSGACCESS=SYNCSUSPEND keyword.

### Environment

The following are the environment requirements for the caller.

Environment	Environment requirement
Dispatchable unit mode:	Task or SRB
Cross memory mode:	When the SENDTOKEN keyword is specified, any PASN, any HASN, any SASN. When the MEMTOKEN keyword is specified, the primary address space must equal the primary address space of the caller of the IXCJOIN macro when it was issued to join the XCF group.
AMODE:	31-bit
ASC mode:	Primary or access register (AR)
Interrupt status:	Enabled for I/O and external interrupts
Locks:	No locks held
Control parameters:	Must be in the primary address space or addressable through a public entry on the caller's dispatchable unit access list (DU-AL)

### Programming Requirements

If the program is in AR mode, issue the SYSSTATE ASCENV=AR macro before IXCMMSGC. SYSSTATE ASCENV=AR tells the system to generate code appropriate for AR mode.

Include the mapping macro IXCYMQAA in your program to map the data returned in ANSAREA when using the IXCMMSGC REQUEST=QUERYMSG service. Use the length and offsets provided in the IXCYMQAA records to ensure compatibility with additional data provided in the future.

Note that some request options are valid only when running in task mode, or when running as a message user routine or as a message notify user routine.

## Restrictions

---

The virtual storage areas specified by ANSAREA and RETMSGTOKEN must be addressable in the caller's primary address space, in an address space or data space that is addressable through a public entry on the caller's dispatchable unit access list (DU-AL), or in a common area data space.

A IXCMSGC REQUEST=CALLEXIT invocation cannot be made with FRRs established while in task mode.

## Input Register Information

---

Before issuing the IXCMSGC macro, the caller does not have to place any information into any register unless using it in register notation for a particular parameter, or using it as a base register.

## Output Register Information

---

When control returns to the caller of the IXCMSGC macro, the general purpose registers (GPRs) contain:

### Register

#### Contents

**0**

Reason code, if GPR 15 contains a non-zero return code that has applicable reason codes.

**1**

Used as a work register by the system.

**2-13**

Unchanged.

**14**

Used as a work register by the system.

**15**

Return code.

When control returns to the caller, the access registers (ARs) contain:

### Register

#### Contents

**0-1**

Used as work registers by the system

**2-13**

Unchanged

**14-15**

Used as work registers by the system

For registers that the system changes, a caller who depends on these registers containing the same value before and after issuing the macro must save these registers and restore them after the system returns control.

## Understanding IXCMSGC Version Support

---

The IXCMSGC macro supports version 0 - 2.

Specify the version of the parameter list that you want generated with the PLISTVER keyword. See Chapter 2, “Specifying a Macro Version Number,” on page 5 for considerations when specifying the version of the parameter list with PLISTVER.

Keywords supported by version 0 include the following:

- ANSAREA
- ANSLEN
- DATATYPE

- EXITPARMS
- MEMTOKEN
- MSGEXIT
- NOTIFYEXIT
- REQUEST
- RETMSGTOKEN
- SOURCE
- STATUS
- TYPE
- USERDATA

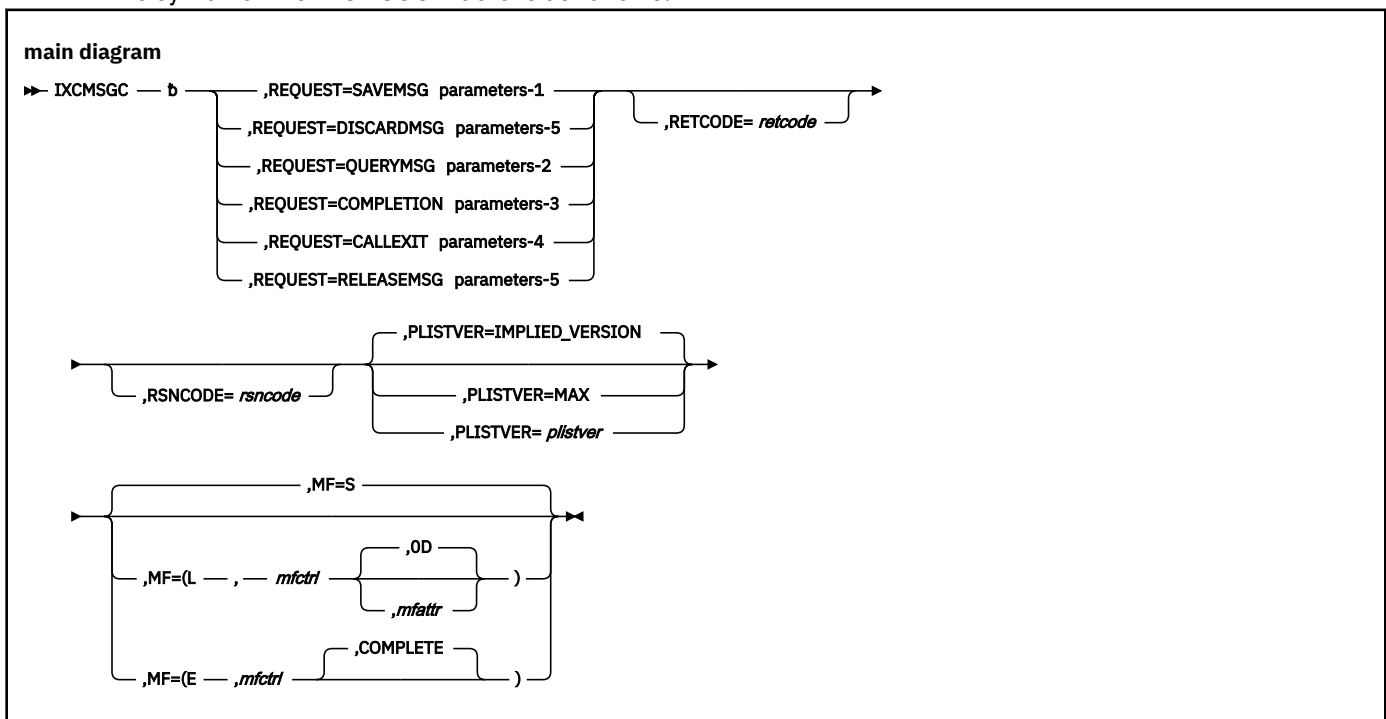
No keywords are supported by version 1.

Keywords supported by version 2 include the following:

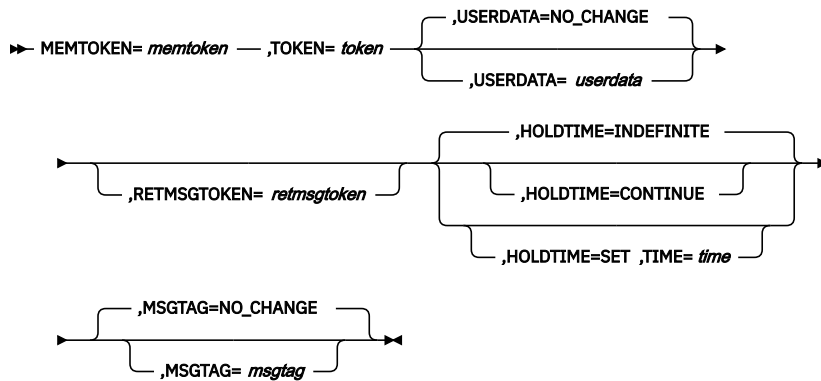
- HOLDTIME
- MQAALEVEL
- MSGTAG
- MSGTAGFILTER
- MSGTAGMASK
- SENDTOKEN
- TIME
- TOKEN

## Syntax Diagram

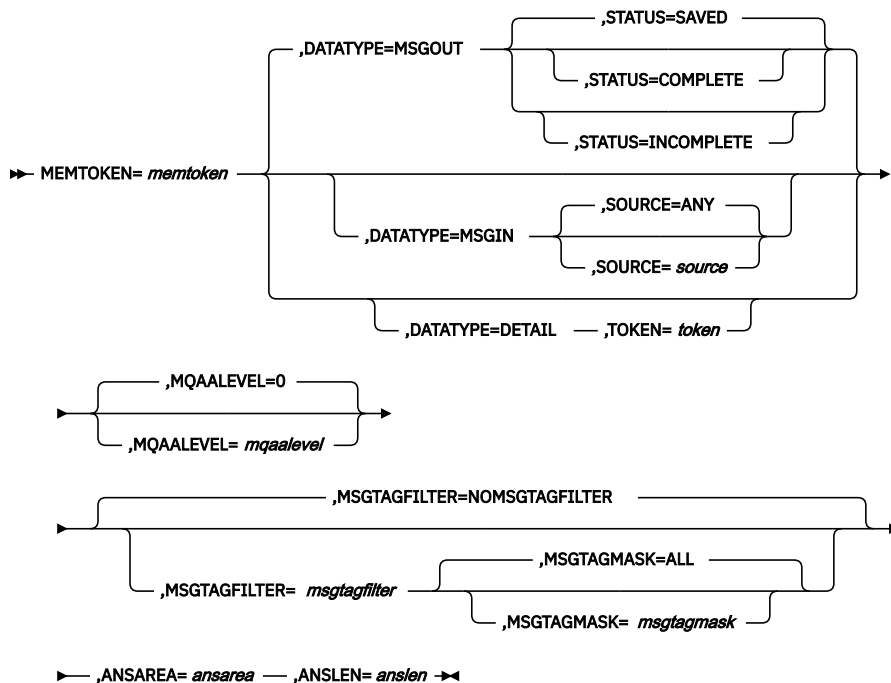
The syntax of the IXCMMSGC macro is as follows:



## parameters-1

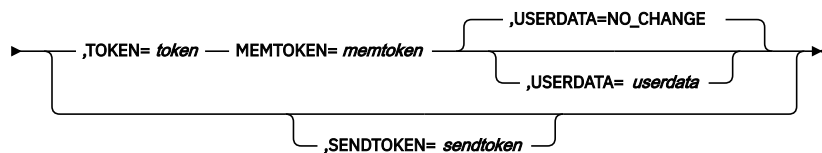


## parameters-2

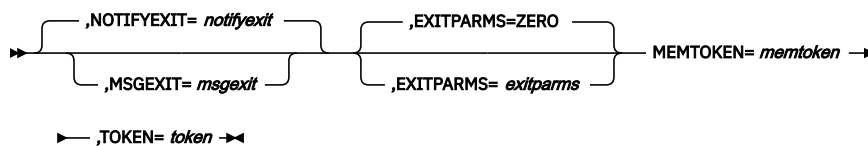


## parameters-3

► ,TYPE=FORCE ►



## parameters-4





**parameters-5**

## Parameter Descriptions

The parameter descriptions are listed in alphabetical order. Default values are underlined:

### **,ANSAREA=ansarea**

Use this output parameter to identify the storage area to contain the data returned by the REQUEST=QUERYMSG service. The data returned consists of a header record followed by zero or more records appropriate to the type of query. The IXCYMQAA macro defines the mappings for the header record and data records.

The storage area specified by ANSAREA must either be in the caller's primary address space or in an address or data space that is addressable through a public entry on the caller's dispatchable unit access list (DU-AL) or in a common area data space.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the output area where IXCMMSGC is to return the requested information.

### **,ANSLEN=anslen**

Use this input parameter to specify the length in bytes of the area you provided on the ANSAREA parameter.

The length of the answer area must be large enough to contain a complete header record (mapped by MQAHEADER). If the answer area is not large enough to contain all the available data records, data collection stops. The header record indicates how much storage would have been needed to collect all the data for the request (field MQAHDRTLEN). Note that the amount of storage needed to collect all the data can change by the time a new query is attempted.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword field, or a decimal value, that contains the length in bytes of the answer area.

### **,DATATYPE=MSGOUT**

### **,DATATYPE=MSGIN**

### **,DATATYPE=DETAIL**

Use this input parameter to indicate what type of information is to be collected by the REQUEST=QUERYMSG request.

### **DATATYPE=MSGOUT**

Collect summary information about messages sent by the member through the message-out service (IXCMMSGO). The data returned for each message includes:

- A token that identifies the message
- User data associated with the message
- Status of the message.

This data is mapped by the MQAMSGOUTSUMMARY record defined in IXCYMQAA.

Note that a message reported by this query might not exist by the time the member making the query request attempts to use the message token returned in the answer area. For example, a message that was incomplete at the time of the query could complete and be processed by a message notify user routine before the member could use the message token returned from the query.

### **DATATYPE=MSGIN**

Collect summary information about incoming messages that the member saved. Information is collected for messages that were saved by the message user routine or for responses saved by the message notify user routine. The data returned for each message includes:

- A token that identifies the message
- User data associated with the message
- The member token of the member that sent the message.

This data is mapped by the MQAMSGINSUMMARY record defined in IXCYMQAA.

#### **DATATYPE=DETAIL**

Collect detail information about the message identified by the TOKEN parameter. The data returned for the message depends on the type of message.

- For a message-out request, the data returned includes:
  - A token that identifies the message
  - User data associated with the message
  - Number of targets for the message
  - Message control data from the message-out request.
  - Flags to describe the characteristics of the message.
  - A table of response data with an entry for each possible target. The entry describes the result of the send and the associated response collection (if any). The entry also provides status information about the send, including whether it might require the asynchronous access of user storage.

This data is mapped by the MQAMSGOUTDETAIL record defined in IXCYMQAA.

- For a message saved by the message user routine or for a response saved by the message notify user routine, the data returned includes:
  - A token that identifies the message
  - User data associated with the message
  - The member token of the member that sent the message
  - Message length
  - Message control data from the sender.

This data is mapped by the MQAMSGINDETAIL record defined in IXCYMQAA.

#### **,EXITPARMS=ZERO**

#### **,EXITPARMS=exitparms**

Use this input parameter to specify the storage area that contains user parameters to be passed to the user routine identified either by the NOTIFYEXIT or MSGEXIT parameter of IXCMSGC.

- For a message user routine, MEPLEXEXITPARMS within the message exit parameter list (mapped by IXCYMEPL) contains a copy of these parameters.
- For a message notify user routine, MNPLEXITPARMS within the message notification parameter list (mapped by IXCYMNPL) contains a copy of these parameters.

The user can define the content and meaning of the 64-bit area. For example, you could use the area to pass the address and ALET of a storage area containing information that determines how the exit routine should perform its processing.

If the user does not specify EXITPARMS, the default is to set the parameters to X'0'.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the 64-bit storage area containing the user parameters.

#### **,HOLDTIME=INDEFINITE**

#### **,HOLDTIME=CONTINUE**

#### **,HOLDTIME=SET**

Use this optional keyword input to indicate how long XCF is to keep the saved message. The default is HOLDTIME=INDEFINITE.

**,HOLDTIME=INDEFINITE**

The message is to be saved until the message is processed by a CALLEXIT, discarded, or the member ends.

**,HOLDTIME=CONTINUE**

The currently established HOLDTIME for the message is to continue to be in effect. CONTINUE can only be specified if a message has previously been saved.

**,HOLDTIME=SET**

A finite HOLDTIME specified by the TIME keyword is to be assigned and set by XCF for the message. SET can be used to establish an initial HOLDTIME for a saved message or be used to modify the HOLDTIME for a message.

**,TIME=xtime**

The name (RS-type), or address in register (2)-(12), of a required halfword input that indicates the number of seconds, starting from the time the SAVEMSG request is processed, XCF is to keep a saved message before discarding the message. The *xtime* value must be in the range 1 - 65535.

**MEMTOKEN=memtoken**

Use this input parameter to specify the member token that was returned by IXCJOIN and that identifies the member making the IXCMSGC request.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 64-bit field that contains the member token.

**,MF=S****,MF=(L,mfctrl)****,MF=(L,mfctrl,mfattr)****,MF=(L,mfctrl,0D)****,MF=(M,mfctrl)****,MF=(M,mfctrl,COMPLETE)****,MF=(M,mfctrl,NOCHECK)****,MF=(E,mfctrl)****,MF=(E,mfctrl,COMPLETE)****,MF=(E,mfctrl,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

**,mfctrl**

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

**,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

**,COMPLETE****,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

**COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if SMILE=*var* were an optional parameter and the default is SMILE=NO\_SMILE then it would not be documented. However, if the default was SMILE=:-), then it would be documented because a value would be the default.

**NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

**,MQAALEVEL=0****,MQAALEVEL=*mqaalevel***

Use this optional keyword to specify the name (RS-type), or address in register (2)-(12), of a byte input variable specifying the level of the various IXCYMQAA record mappings which are to be returned. Valid values are 0 and 1. The IXCYMQAA macro documents the maximum level supported by each DATATYPE. A value of 0 indicates that the base level IXCYMQAA records are to be returned. A value of 1 indicates that the level-1 IXCYMQAA records are to be returned. The default is 0.

**,MSGEXIT=*msgexit***

Use this input parameter to identify the message user routine in the joiner's address space that is to receive control as specified by a REQUEST=CALLEXIT invocation. The user routine need not be the same as the message user routine defined when the member invoked IXCJOIN to join the XCF group.

Note that MSGEXIT is mutually exclusive with NOTIFYEXIT.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the 31-bit message user routine that is to receive control.

**,MSGTAG=NO\_CHANGE****,MSGTAG=*msgtag***

Use this optional keyword to specify the name (RS-type), or address in register (2)-(12), of an optional 64 character variable that contains a user-defined message tag to be associated with the saved message. *msgtag* persists for a message until it is changed. You can use MSGTAG later as a filter criteria in a Message Control QUERYMSG request. The storage area indicated by MSGTAG must either be in the caller's primary address space or in an address/data space that is addressable through a public entry on the caller's dispatchable unit access list (DU-AL), or in a common area data space. The default is MSGTAG=NO\_CHANGE, which means that the existing message tag is to be left as is.

**,MSGTAGFILTER=NO\_MSGTAGFILTER****,MSGTAGFILTER=*msgtagfilter***

Use this optional keyword to specify the name (RS-type), or address in register (2)-(12), of a 64-character variable that contains a user-defined message tag to be used as a search filter criteria when collecting data for the QUERYMSG request. The storage area indicated by MSGTAGFILTER must either be in the caller's primary address space or in an address/data space that is addressable through a public entry on the caller's dispatchable unit access list (DU-AL), or in a common area data space. The default is MSGTAGFILTER=NO\_MSGTAGFILTER, which means that a message tag filter is not to be used.

**,MSGTAGMASK=ALL****,MSGTAGMASK=*msgtagmask***

Use this optional keyword to specify the name (RS-type), or address in register (2)-(12), of a 64-character input that identifies a mask pattern to be applied to MSGTAGFILTER when selecting messages whose data is to be collected.

The position of each bit in MSGTAGMASK corresponds to the same bit position in MSGTAGFILTER and in the message tag associated with a message. For example, if bit *n* in MSGTAGMASK is 1, bit *n* of MSGTAGFILTER is to be compared to bit *n* of the message tag associated with a message. If bit *n* in MSGTAGMASK is 0, then the corresponding bit in MSGTAGFILTER is not to be used for comparison. If all bits selected by MSGTAGMASK are equal and the message passes all other filter criteria, data associated with the message is to be collected.

Specifying a MSGTAGMASK where all the bits are zero indicates that the MSGTAGFILTER is always a match, and data for messages that pass all other filter criteria is to be collected. The storage area indicated by MSGTAGMASK must either be in the caller's primary address space or in an address/data space that is addressable through a public entry on the dispatchable unit access list (DU-AL) of the caller, or in a common area data space. The default is MSGTAGFILTER= ALL, which means that all bits in the MSGTAGFILTER need to be compared.

**,NOTIFYEXIT=notifyexit**

Use this input parameter to identify the message notify user routine in the joiner's address space that is to receive control as specified by a REQUEST=CALLEXIT invocation. The user routine need not be the same as the message notify user routine defined when the member invoked IXCJOIN to join the XCF group nor need it be the one specified on the IXCMMSGC invocation to send the message.

Note that NOTIFYEXIT is mutually exclusive with MSGEXIT.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the 31-bit message notify user routine that is to receive control.

**,PLISTVER=IMPLIED\_VERSION**

**,PLISTVER=MAX**

**,PLISTVER=plistver**

Use this input parameter to specify the version of the macro. See [“Understanding IXCMMSGC Version Support”](#) on page 134 for a description of the options available with PLISTVER.

**,REQUEST=SAVEMSG**

**,REQUEST=DISCARDMSG**

**,REQUEST=QUERYMSG**

**,REQUEST=COMPLETION**

**,REQUEST=CALLEXIT**

**,REQUEST=RELEASEMSG**

Use this input parameter to indicate the type of request to be processed.

**REQUEST=SAVEMSG**

Save the message for later processing. This request is valid only when invoked from a message user routine or a message notify user routine. Use the TOKEN parameter to identify the message that is to be saved. If the user routine neither saves the message with the SAVEMSG service nor receives the message with the IXCMMSGC service, the system automatically discards the message when the user routine gives up control.

A message notify user routine can save the entire message-out request including any available responses and can save individual responses as independent messages. A message user routine can save its message.

Once the user routine saves the message, the system invalidates the message token passed through the TOKEN keyword. Thus, the message is no longer available to the user routine because both the IXCMMSGC service and the IXCMMSGC service will reject requests that attempt to use the invalidated token. You can process a saved message by invoking the IXCMMSGC REQUEST=CALLEXIT service.

The system automatically discards saved messages when the member becomes not active. An active member can discard its saved message by invoking the IXCMMSGC REQUEST=DISCARDMSG service.

When a message is saved from within a message user routine, only the undelivered portion of the message is saved along with the data required to create a new message exit parameter list (MEPL). Once a member has received message data into the member's buffer area, the message

data is considered delivered. The delivered part of the message cannot be redelivered (with REQUEST=CALLEXIT) nor can it be saved. When a partially delivered message is saved, only the undelivered portion of the message is saved for later retrieval by the REQUEST=CALLEXIT service.

When a message with or without its associated responses is saved from within a message notify user routine, the message and its associated responses can be saved as a single entity or each individual response can be saved independently of the message or both.

- When saved as a single message/response entity, the undelivered portion of the responses is saved along with the data required to create a new message notification parameter list (MNPL). The token, passed by the TOKEN parameter, identifying the message is invalidated, as well as all the message tokens identifying the individual responses, if any, associated with the message (MNPLTRMSGITOKEN). The saved message/response entity can be processed at a later time by a message notify user routine specified on the IXCMSGC REQUEST=CALLEXIT service.
- When an individual response is saved as an independent message, the undelivered portion of the responses is saved along with the data required to create a new message exit parameter list (MEPL). The message token passed with the TOKEN parameter is invalidated. Note that a saved response is considered an independent message and must be processed as such with a message user routine specified on the IXCMSGC REQUEST=CALLEXIT service.

### **REQUEST=DISCARDMSG**

Discard the message identified by TOKEN or SENDTOKEN. A discarded message is no longer available for processing:

- Usage with the IXCMSGI(X) and IXCMSGO(X) macros

A discarded message is no longer available for processing. For example, XCF does not deliver a discarded message to any exit routine and the message is not visible to the message control QUERYMSG service. Any responses that are still associated with a message/response entity are discarded; however, saved responses are not discarded because saving a response causes the response to become disassociated from the message/response entity.

If an exit routine does not explicitly save a message or if the message is not received by the IXCMSGI(X) macro, XCF automatically discards the message when the exit routine gives up control. Therefore, most members do not need to use REQUEST=DISCARDMSG on the macro from within an exit routine if they do not want to process the message. (Letting XCF discard the message automatically is preferable to discarding from within an exit routine because the automatic discard by XCF has less system processing overhead.)

If a message is discarded while an exit routine is processing the message, attempts that the exit routine makes to process the message through XCF macros are rejected with a return/reason code. Depending on the timing, the current operation that the exit is performing might be permitted to complete before the message is discarded. In such a case, IXCMSGC REQUEST=DISCARDMSG returns to indicate that the discard of the message is pending.

You can discard incomplete message-out requests before the message completes. For example, a message queued by XCF because of a lack of signalling resource or a message whose XCF-managed response collection is not complete can be discarded. Such a discard has the effect of cancelling the message-out request. For a broadcast to multiple targets, XCF does not send any remaining messages that have not yet been sent and discards any collected responses. XCF also discards any responses that arrive for the cancelled message, and the system does not call the notification exit for the cancelled message. If XCF has initiated a send to a particular target member before the message cancellation, it delivers any of those messages as usual.

- Usage with the client/server IXCSEND and IXCRECV macros.

For XCF IXCSEND messages identified by SENDTOKEN, responses to a discarded message cannot be retrieved using the IXCRECV service. Status information about the message is no longer available. Undelivered responses that arrived before the message was discarded are no longer available. Subsequently arriving responses are discarded. Discarding a message releases a blocking receive initiated for that message. That is, if a unit of work has issued IXCRECV REQTYPE=BLOCKING to receive a message, and another unit of work issues IXCMSGC

REQUEST=DISCARDMSG to discard the message, the blocked receiver is resumed and receives a return and reason code indicating that the message was not found. No data will be returned in the answer area for the resumed IXCRECV request.

Use the XCF message control completion service to force a message-out or client/server request to be stopped in a way that preserves the request for further processing by the member or IXCRECV.

To force a message-out or a client/server request to be stopped in a way that preserves the request for further processing by the member or by IXCRECV, use the IXCMMSGC REQUEST=COMPLETION.

### **REQUEST=QUERYMSG**

Return information appropriate for the requested type of query in the storage area specified by ANSAREA. The DATATYPE parameter specifies the type of information requested. The contents of the answer area are mapped by IXCYMQAA. Use the length and offsets provided in the IXCYMQAA records to ensure compatibility with additional data provided in the future.

### **REQUEST=COMPLETION**

Complete the message identified by the TOKEN parameter. You can request COMPLETION for a message-out request that XCF has accepted for delivery, but does not yet consider complete.

A message is considered complete in the following situations&colon;

- A message-out request is considered complete if it times-out or if the REQUEST=COMPLETION service is used to force its completion.
- A message with response is considered complete when one or more of the expected responses arrive.
- A message without response is considered complete as soon as XCF has initiated the send of the message.
- A message broadcast to multiple targets is considered complete when XCF has initiated the send of the message to every valid target member.

Once completed, processing for the message continues just as if it completed without direct intervention. XCF no longer attempts to send the message to any intended target. XCF discards any responses to the message that subsequently arrive. If notification was to be provided upon completion of the message, the message notify user routine receives control.

Consider the following usage of IXCMMSGC requests:

- For a message-out request: The user notify exit receives control. If not, you can use REQUEST=CALLEXIT to call a user notify exit routine.
- For an XCF client/server request: Undelivered responses that arrive before the message is forced to complete remain available. Any responses arriving after the message is considered to be complete are discarded. Completing a message releases a blocking initiated for that message; that is, if a unit of work has issued IXCRECV REQTYPE=BLOCKING to receive a message and another unit of work issues IXCMMSGC REQUEST=COMPLETION to force that message to completion, the blocked receiver is resumed and receives any responses that have arrived up to that point.

If a unit of work has issued IXCMMSGOX with MSGACCESS=SYNCSUSPEND or a client/server request (IXCSEND) and the unit of work is paused by XCF, the service invoker can receive a return and reason code indicating that the request may not have completed sending the message to all intended targets.

Use the REQUEST=DISCARDMSG service to force a message-out or client/server request to be stopped in a way that makes the message unavailable for further processing.

To force a message-out request to be stopped so that the message is not available for further processing, use REQUEST=DISCARDMSG (for example, to prevent the notify exit from getting control even though it was requested on the IXCMMSGO macro.)

**REQUEST=CALLEXIT**

Call a user routine to process the message identified by the TOKEN parameter. The message can be a saved message or a completed message-out request. The type of user routine must be appropriate for the message.

- Use a message user routine to process messages saved by a message user routine and responses saved by a message notify user routine.
- Use a message notify user routine to process completed message-out requests and saved message and response entities.

A CALLEXIT request first passes control to a system service routine that sets up the appropriate environment, collects information pertinent to the message, and calls the specified user routine. The user routine receives control synchronously — under the same unit of work as the invoker of the CALLEXIT service. After performing whatever processing is required, the user routine returns to the system service routine, which then releases resources as necessary, restores the environment, and returns control to the invoker of the CALLEXIT service.

Only one user routine is allowed to process a particular message at any one time. The system rejects the CALLEXIT request if a user routine is already processing the message.

**REQUEST=RELEASEMSG**

Release a client/server blocking receive request (IXCRECV), a client/server send message request (IXCSEND), or a message-out request (IXCMMSGOX) issued with the MSGACCESS=SYNCSUSPEND keyword.

If a unit of work has issued IXCRECV REQTYPE=BLOCKING to receive the results of an XCF client/server message, and another unit of work issues IXCMMSGC REQUEST=RELEASEMSG, the blocked receiver is resumed, and the receiver receives a return and reason code indicating that the receive request is released. Data is not returned in the answer area or the data area for the released IXCRECV request. The message persists for the HOLDTIME value specified on the IXCSEND macro service, and a subsequent IXCRECV can be issued to receive the message information.

If a unit of work has issued IXCMMSGOX with MSGACCESS=SYNCSUSPEND or a client/server request (IXCSEND) and the unit of work is paused by XCF, the service invoker can receive a return and reason code indicating that the request might not complete sending the message to all intended targets. For an IXCMMSGOX broadcast or IXCSEND to multiple targets, any remaining unsent messages are not sent. For message-out requests, if notification is to be provided upon completion of the message, the user notify exit receives control; otherwise, the message control CALLEXIT service can be used to call a user notify exit routine.

**Note:** If XCF initiates a send to a particular target member, the message is delivered. For an IXCSEND request with a HOLDTIME specified, the IXCRECV service macro can be issued to receive the results of the released client/server request.

**,RETCODE=retcode**

Use this output parameter to specify the location in which the system is to copy the return code from GPR 15.

**To Code:** Specify the name of a fullword field (or register 2-12 containing the address of the field) to contain the return code.

**,RETMSGTOKEN=retmsgtoken**

Use this output parameter to contain a token that the REQUEST=SAVEMSG service returns to identify a saved message.

Pass this token to the DISCARDMSG or CALLEXIT services of IXCMMSGC. You can also obtain a token for the message to be passed to these services by invoking the QUERYMSG service of IXCMMSGC.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-character storage area to contain the message token.

The storage area must be in the caller's primary address space, in an address space or data space that is addressable through a public entry on the caller's dispatchable unit access list (DU-AL), or in a common area data space.



**,RSNCODE=*rsncode***

Use this output parameter to specify the location in which the system is to copy the reason code from GPR 0.

**To Code:** Specify the name of a fullword field (or register 2-12 containing the address of the field) to contain the reason code.

**SENDTOKEN=*sendtoken***

Use this input parameter to specify a token that is associated with an XCF client/server request/response entity. You can obtain the token from the IXCSEND macro by using the RETMSGTOKEN parameter. For a DISCARDMSG request, the token identifies the XCF client/server request or server response to be discarded. For a COMPLETION request, the token identifies the XCF client/server request or server response to be completed. For a RELEASEMSG request, the token identifies the XCF client/server IXCRECV or IXCSEND request that is to be released.

SENDTOKEN=*sendtoken* is mutually exclusive with TOKEN=*token*.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 32-byte area containing the message token.

**,SOURCE=ANY****,SOURCE=*source***

Use this input parameter to specify a member token. The REQUEST=QUERY service will collect only the incoming messages sent by the specified member.

**To Code:** Specify the name of a 64-bit field (or register 2-12 containing the address of the field) to contain the member token.

**,STATUS=SAVED****,STATUS=COMPLETE****,STATUS=INCOMPLETE**

Use this input parameter to specify the status of the message-out requests for which data is to be collected with REQUEST=QUERYMSG.

**STATUS=SAVED**

Collect information about messages that were saved through use of the REQUEST=SAVMSG service. These messages are the message and response entities saved by the message notify user routine.

**STATUS=COMPLETE**

Collect information about messages that are completed. A message is considered complete in the following situations:

- A message-out request is considered complete if it times-out or if the REQUEST=COMPLETION service is used to force its completion.
- A message with response is considered complete when the expected response(s) arrive.
- A message without response is considered complete as soon as XCF has initiated the send of the message.
- A message broadcast to multiple targets is considered complete when XCF has initiated the send of the message to every valid target member.

**STATUS=INCOMPLETE**

Collect information about messages that have not yet completed. For example, the message might have been queued due to an XCF signalling buffer shortage, or in the case of a response-required signal, a response to the message might not have arrived and the message has either not yet timed-out or been forced to completion with IXCMMSGC REQUEST=COMPLETION.

**,TOKEN=*token***

Use this input parameter to identify the message to which the IXCMMSGC request refers. The token used to identify a message can vary. Two different tokens might identify the same message. Do not expect to compare tokens to determine if they identify the same message. For example, a token presented to a user routine and a token obtained by other means (such as with the RETMSGOTOKEN parameter of IXCMMSGC or the token obtained from the IXCMMSGC SAVMSG or the IXCMMSGC

QUERYMSG service) could be different and yet represent the same message. The tokens obtained by the other means described above will be the same for the same message, at least for a particular member.

A description of how to obtain a token suitable for each of the IXCMSGC services follows:

- For a SAVMSG request, the token identifies the message to be saved. The system invalidates this token once the message is saved.
  - Obtain the token when running in a message user routine from the message token (MEPLMSGITOKEN) passed in the message exit parameter list, mapped by IXCMEPL.
  - Obtain the token when running in a message notify user routine from the message token (MNPLMSGOTOKEN) passed in the message notification parameter list, mapped by IXCYNPL. This token identifies the message and its associated responses (if any) as a single entity. For an individual response, obtain the token from the message token (MNPLTRMSGITOKEN) passed in the MNPL.
- For a DISCARDMSG request, the token identifies the message to be discarded.
  - Obtain the token for an incoming message from one of these sources:
    - The SAVMSG service with the RETMSGTOKEN parameter
    - The QUERYMSG service with the DATATYPE=MSGIN
    - The MEPL (MEPLMSGITOKEN) or the MNPL (MNPLTRMSGITOKEN) if the user routine has not finished processing the message. Note that the tokens from these parameter lists are invalidated if the user routine saves the message, discards the message, receives all the message data, or gives up control. The MNPLTRMSGITOKEN also is invalidated if the message notify user routine finishes processing the message/response entity. Note also that the token is valid only under the user routine unit of work.
  - Obtain the token for a message/response entity from one of these sources:
    - The IXCMSGO(X) service with the RETMSGOTOKEN parameter
    - The SAVMSG service with the RETMSGTOKEN parameter
    - The QUERYMSG service with DATATYPE=MSGOUT
    - The MNPL (MNPLMSGOTOKEN) if the message notify user routine has not finished processing the message/response entity. Note that this token is invalidated if the user routine saves the message, discards the message, or gives up control.

Each of the tokens described is invalidated if any other unit of work discards the message.

TOKEN=*token* is mutually exclusive with SENDTOKEN=*sendtoken* .

- For a QUERYMSG request, the token identifies the message for which the system is to collect DETAIL information.
  - Obtain the token to identify the message from one of these sources:
    - The SAVMSG service with the RETMSGTOKEN parameter
    - The IXCMSGO(X) service with the RETMSGOTOKEN parameter
    - The IXCMSGC QUERYMSG service with DATATYPE=MSGIN or MSGOUT.
- For a COMPLETION request, the token identifies the message to be completed.
  - Obtain the token to identify the message from one of these sources:
    - The IXCMSGO(X) service with the RETMSGOTOKEN parameter
    - The IXCMSGC QUERYMSG service with DATATYPE=MSGOUT and STATUS=INCOMPLETE.
- For a CALLEXIT request, the token identifies the message to be processed by the user routine.
  - Obtain the token for a message user routine from one of these sources:
    - The IXCMSGC QUERYMSG service with DATATYPE=MSGIN

- The IXCMSGC SAVMSG service with RETMSGTOKEN parameter when the message was saved by a message user routine or a response was saved by a message notify user routine.
- Obtain the token for a message notify user routine from one of these sources:
  - The IXCMSGO(X) service with the RETMSGOTOKEN parameter
  - The IXCMSGC QUERYMSG service with DATATYPE=MSGOUT and STATUS=SAVED or COMPLETE
  - The IXCMSGC SAVMSG service with the RETMSGTOKEN parameter when the message/response entity was saved by a message notify user routine.

**,TYPE=FORCE**

Use this input parameter to specify that the message is to be immediately considered as complete when invoking the IXCMSGC REQUEST=COMPLETION service. This type of completion is comparable to forcing a message time-out value to expire immediately.

**,USERDATA=NO\_CHANGE****,USERDATA=userdata**

Use this input parameter to specify a storage area that contains user data to be associated with a saved or completed message. The system presents the user data to a user routine.

- For a saved message, the following fields are used to pass the user data:
  - The field MEPLEXUSERDATA contains the user data passed to a message user routine.
  - The field MNPLMSGOUSERDATA contains the user data passed to a message notify user routine.
- For a completed message, the field MNPLMSGOUSERDATA contains the user data passed to a message notify user routine. Note that the current user data associated with the message is replaced by this user data only if the invocation of the IXCMSGC COMPLETION service causes the message to be considered complete.

**To Code:** Specify the RS-name or address (using a register from 2 to 12) of the 8-character field that contains the user data to be associated with the message.

## ABEND Codes

---

An abend code that an issuer of IXCMSGC might receive is listed as follows. For detailed abend code information, see [z/OS MVS System Codes](#).

- X'00C'
- X'073'

## Return and Reason Codes

---

When control returns from IXCMSGC:

- GPR 15 (and *retcode*, if you coded RETCODE) contains a return code.
- GPR 0 (and *rsncode*, if you coded RSNCODE) contains a reason code, if applicable. Note that bits 0-15 of the reason code might contain component- diagnostic data for use by IBM service personnel. XCF provides the IXCMSGRSNCODEMASK constant to mask off the component-diagnostic data.

The IXCMSGC macro provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

**00**

IXMSGCRCSUCCESSFUL

**04**

IXMSGCRCWARNING

**08**

IXMSGCRCINVALIDPARMS

**0C**

IXMSGCRCENVIRONMENTALERROR

## 10

## IXCMMSGCRCSYSTEMERROR

Table 11 on page 148 identifies the hexadecimal return and reason codes and the equate symbol associated with each reason code.

Table 11. Return and Reason Codes for the IXCMMSGC Macro		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0	None.	<b>Meaning:</b> IXCMMSGC completed successfully. <b>Action:</b> None.
4	xxxx0004	<b>Equate Symbol:</b> IXCMMSGCRSNANSAREATOOSMALL <b>Meaning:</b> The IXCMMSGC QUERYMSG request completed successfully. The area specified by ANSAREA was large enough to contain the header information (MQAHEADER), but was not large enough to contain all the data that was requested. The MQAHDRTLEN field indicates the total length of the output answer area that would have been needed to contain all the requested information. It is possible that only the MQAHEADER was provided, in which case the MQAHDR#ENTRIES field would contain zero. <b>Action:</b> Retry the request with an answer area whose length is greater than or equal to the number of bytes indicated by MQAHDRTLEN. Note that the amount of data to be returned can change dynamically so that the length indicated by MQAHDRTLEN might be too small for all the data when you try the request again.
4	xxxx000C	<b>Equate Symbol:</b> IXCMMSGCRSNSAVEDMSGTIMEOUT <b>Meaning:</b> HOLDTIME(CONTINUE) was specified for a SAVEMSG request but the established HOLDTIME for the saved message has expired and the message will be discarded as soon as CALLEXIT processing completes. <b>Action:</b> If the message is to be saved, re-issue the SAVEMSG request specifying HOLDTIME(INDEFINITE) or HOLDTIME(SET) to establish a new HOLDTIME.
4	xxxx0008	<b>Equate Symbol:</b> IXCMMSGCRSNMSGALREADYCOMPLETE <b>Meaning:</b> The message has already completed. <b>Action:</b> None. A message COMPLETION was requested for a message that was already completed.
4	xxxx0018	<b>Equate Symbol:</b> IXCMMSGCRSNMSGDISCARDPENDING <b>Meaning:</b> A message-discard is pending. Some work unit is currently processing the message. (For example, the message notify user routine might be processing the message.) The system will delete the message as soon as the work unit is finished with the message. <b>Action:</b> None, the message is not available. Storage areas for a message-out request that were to be preserved until the message completed can be freed (applies to IXCMMSGO/IXCMMSGOX MSGACCESS=ASYN or MSGACCESS=SYNCSUSPEND requests).
4	xxxx0020	<b>Equate Symbol:</b> IXCMMSGCRSNNOMSGRELEASEDMSGOX <b>Meaning:</b> A RELEASMSG request for a paused message-out request identified by TOKEN did not find a unit of work to release. <b>Action:</b> None. The system returns to the caller without releasing a unit of work.

Table 11. Return and Reason Codes for the IXCMMSGC Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	xxxx0024	<p><b>Equate Symbol:</b> IXCMMSGCRSNNOMSGRELEASEDCLIENT</p> <p><b>Meaning:</b> A RELEASEMSG request for a blocking IXCRECV or a paused IXCSEND service call identified by SENDTOKEN did not find a unit of work to release.</p> <p><b>Action:</b> The system returns to the caller without releasing a unit of work.</p>
8	xxxx0004	<p><b>Equate Symbol:</b> IXCMMSGCRSNMEMBERNOTACTIVE</p> <p><b>Meaning:</b> The member token does not identify an active member associated with the primary address space that was current when IXCMMSGC was invoked.</p> <p><b>Action:</b> Reissue the request with a correct member token or from the correct address space.</p>
8	xxxx0016	<p><b>Equate Symbol:</b> IXCMMSGCRSNINAPPROPEXITROUTINENAME</p> <p><b>Meaning:</b> The user routine type specified was inappropriate for this request.</p> <ul style="list-style-type: none"> <li>• Messages saved by a message user routine and responses saved by a message notify user routine must be processed by a message user routine. Use the MSGEXIT parameter to specify the user routine.</li> <li>• A completed message-out request, or a saved message-out response entity must be processed by a message notify user routine. Use the NOTIFYEXIT parameter to specify the user routine.</li> </ul> <p><b>Action:</b> The type of user routine specified for a CALLEXIT request must be appropriate for the type of message to be processed. Retry the request with the correct user routine.</p>
8	xxxx0040	<p><b>Equate Symbol:</b> IXCMMSGCRSNRESERVEDFIELDNOTNULL</p> <p><b>Meaning:</b> Program error or environmental error. A reserved field in the control parameter list is not zero.</p> <p><b>Action:</b> Check to see if your program inadvertently overlaid the parameter list storage, and that it was assembled with the correct macro library for the release of z/OS on which your program is running.</p>
8	xxxx0100	<p><b>Equate Symbol:</b> IXCMMSGCRSNBADPLISTALET</p> <p><b>Meaning:</b> Program error. Your program is not running in primary ASC mode, and the ALET that qualifies the address of the control parameter list is not zero, associated with a valid public entry on the DU-AL, or for a common area data space.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• Your program is not intended to run in primary ASC mode.</li> <li>• You specified SYSSTATE ASCENV=AR before issuing the IXCMMSGC macro.</li> <li>• The ALET for the parameter list is a valid public entry on the DU-AL, is zero (primary address space ALET), or for a common area data space.</li> </ul>
8	xxxx0104	<p><b>Equate Symbol:</b> IXCMMSGCRSNBADPLISTVERSION</p> <p><b>Meaning:</b> The parameter list is not valid. Either the version number in the parameter list is not valid, or the release level of z/OS on which the caller is running does not support this version of the message control service.</p> <p><b>Action:</b> Retry the request with the correct version of the parameter list.</p>

Table 11. Return and Reason Codes for the IXCMSGC Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0108	<b>Equate Symbol:</b> IXMSGCRSNBADPLISTFUNCCODE <b>Meaning:</b> The parameter list is not valid. The function code in the parameter list is not valid. <b>Action:</b> Check to see if your program inadvertently overlaid the parameter list in storage, and that it was assembled with the correct macro library for the release of z/OS on which it is running.
8	xxxx010C	<b>Equate Symbol:</b> IXMSGCRSNBADPLISTADDRESS <b>Meaning:</b> The parameter list is not accessible. Its storage is not addressable. <b>Action:</b> Make sure the parameter list is accessible to XCF and retry the request.
8	xxxx011C	<b>Equate Symbol:</b> IXMSGCRSNNOTENABLED <b>Meaning:</b> The caller is not enabled. <b>Action:</b> Correct your program so that it does not issue IXCMSGC while it is disabled.
8	xxxx012C	<b>Equate Symbol:</b> IXMSGCRSNLOCKSHELD <b>Meaning:</b> The caller is holding a lock. <b>Action:</b> Correct your program so that it does not issue IXCMSGC while holding any locks.
8	xxxx013C	<b>Equate Symbol:</b> IXMSGCRSNANSAREASMALLERTHANHEADER <b>Meaning:</b> The area specified by ANSAREA is too small. The answer area must be at least as long as the header record (MQAHEADER). <b>Action:</b> Retry the request with a larger answer area.
8	xxxx0140	<b>Equate Symbol:</b> IXMSGCRSNANSAREABADALET <b>Meaning:</b> The area specified by ANSAREA is not accessible. The ALET of the answer area must be zero, a valid entry on the dispatchable unit access list (DU-AL), or a valid entry for a common area data space. <b>Action:</b> Ensure that: <ul style="list-style-type: none"> <li>• Your program is not intended to run in primary ASC mode.</li> <li>• You specified SYSSTATE ASCENV=AR before issuing the IXCMSGC macro.</li> <li>• The ALET for the parameter list is a valid public entry on the DU-AL, is zero (primary address space ALET), or for a common area data space.</li> </ul>
8	xxxx0148	<b>Equate Symbol:</b> IXMSGCRSNANSAREABADADDRESS <b>Meaning:</b> An error occurred attempting to access the answer area. <b>Action:</b> Make sure the answer area is accessible to XCF and reissue the request.
8	xxxx0150	<b>Equate Symbol:</b> IXMSGCRSNMsgTagBadAlet <b>Meaning:</b> MSGTAG not accessible. <b>Action:</b> The ALET of the MSGTAG is neither zero nor a valid entry on the Dispatchable Unit Access List (DU-AL), nor a valid entry for a common area data space. Retry the request with the correct ALET.

Table 11. Return and Reason Codes for the IXCMMSGC Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0152	<b>Equate Symbol:</b> IXCMMSGCRSNMSGTAGBADADDRESS <b>Meaning:</b> Error accessing MSGTAG. <b>Action:</b> Make sure the MSGTAG is accessible to XCF, and reissue the request.
8	xxxx0154	<b>Equate Symbol:</b> IXCMMSGCRSNMSGTAGFILTERBADALET <b>Meaning:</b> MSGTAGFILTER not accessible. <b>Action:</b> The ALET of the MSGTAGFILTER is neither zero nor a valid entry on the Dispatchable Unit Access List (DU-AL), nor a valid entry for a common area data space. Retry the request with the correct ALET.
8	xxxx0158	<b>Equate Symbol:</b> IXCMMSGCRSNMSGTAGFILTERBADADDRESS <b>Meaning:</b> Error accessing MSGTAGFILTER. <b>Action:</b> Make sure the MSGTAGFILTER is accessible to XCF, and reissue the request.
8	xxxx0160	<b>Equate Symbol:</b> IXCMMSGCRSNMSGTAGMASKBADALET <b>Meaning:</b> MSGTAGMASK not accessible. <b>Action:</b> The ALET of the MSGTAGMASK is neither zero nor a valid entry on the Dispatchable Unit Access List (DU-AL), nor a valid entry for a common area data space. Retry the request with the correct ALET.
8	xxxx0168	<b>Equate Symbol:</b> IXCMMSGCRSNMSGTAGMASKBADADDRESS <b>Meaning:</b> Error accessing MSGTAGMASK. <b>Action:</b> Make sure the MSGTAGMASK is accessible to XCF, and reissue the request.
8	xxxx0170	<b>Equate Symbol:</b> IXCMMSGCRSNTOKENBADALET <b>Meaning:</b> SENDTOKEN not accessible. <b>Action:</b> The ALET of the SENDTOKEN is neither zero nor a valid entry on the Dispatchable Unit Access List (DU-AL), nor a valid entry for a common area data space. Retry the request with the correct ALET.
8	xxxx0170	<b>Equate Symbol:</b> IXCMMSGCRSNTOKENBADALET <b>Meaning:</b> SENDTOKEN not accessible. <b>Action:</b> The ALET of the SENDTOKEN is neither zero nor a valid entry on the Dispatchable Unit Access List (DU-AL), nor a valid entry for a common area data space. Retry the request with the correct ALET.
8	xxxx0172	<b>Equate Symbol:</b> IXCMMSGCRSNTOKENBADADDRESS <b>Meaning:</b> Error accessing SENDTOKEN <b>Action:</b> Make sure the SENDTOKEN is accessible to XCF, and reissue the request.  Ensure that you are issuing the SAVMSG request under the user routine unit of work.

Table 11. Return and Reason Codes for the IXCMMSGC Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0204	<p><b>Equate Symbol:</b> IXCMMSGCRSNTOKENNOTFORDISCARDMSG</p> <p><b>Meaning:</b> The token specified by TOKEN is not valid for the DISCARDMSG service.</p> <p><b>Action:</b> Check to see if your program inadvertently overlaid the parameter list in storage.</p> <p>Ensure that you are issuing the DISCARDMSG request under the user routine unit of work.</p>
8	xxxx0208	<p><b>Equate Symbol:</b> IXCMMSGCRSNTOKENFORCALLEXITINVALID</p> <p><b>Meaning:</b> The token specified by TOKEN is not valid for the CALLEXIT service. Valid tokens for the CALLEXIT service are those returned by the IXCMMSGC SAVEMSG service, the IXCMMSGC service (RETMSGOTOKEN), and the IXCMMSGC QUERYMSG service.</p> <p><b>Action:</b> Ensure that the token is one that was returned by a successful invocation of the IXCMMSGC SAVEMSG or QUERYMSG service or the RETMSGOTOKEN from the IXCMMSGC service.</p> <p>Also, check to see if your program inadvertently overlaid the parameter list in storage.</p> <p>Note that message tokens presented to the message user routine and the message notify user routine are not valid for the CALLEXIT service.</p>
8	xxxx020C	<p><b>Equate Symbol:</b> IXCMMSGCRSNMESSAGEUNAVAILABLE</p> <p><b>Meaning:</b> Message not available. The message identified by the input message token (TOKEN or SENDTOKEN) either does not exist, or is no longer accessible using the indicated token. The message was either completely delivered, processed, or discarded. In the case of an input TOKEN, this situation can also occur if the specified MEMTOKEN does not identify the XCF group member to whom the message token was given.</p> <p><b>Action:</b> Verify that the input token (either TOKEN or SENDTOKEN) was validly specified in an appropriate context. If the message was for an XCF group member, verify that the message token (TOKEN) is relevant to the specified member (MEMTOKEN). Reissue the request with the correct token(s) in the correct context.</p> <p>When a user routine saves the message, the message is no longer available to the user routine. If the message still exists, use the token returned via the RETMSGOTOKEN keyword on the IXCMMSGC invocation that saved the message to process the message.</p> <p>Another option is to use the token returned by the IXCMMSGC QUERYMSG service.</p>
8	xxxx0212	<p><b>Equate Symbol:</b> IXCMMSGCRSNSENDTOKENINVALID</p> <p><b>Meaning:</b> SENDTOKEN not valid. The SENDTOKEN must be a token that was returned by the IXCMMSGC service through the RETMSGOTOKEN keyword.</p> <p><b>Action:</b> Verify the SENDTOKEN and retry the request with the correct SENDTOKEN.</p>



Table 11. Return and Reason Codes for the IXCMMSGC Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0220	<p><b>Equate Symbol:</b> IXCMMSGCRSNTOKENNOTFORFORCECOMPLETION</p> <p><b>Meaning:</b> The message TOKEN is not valid for Force Completion. The message token must be a token that was returned either by the IXCMMSGO service with the RETMSGOTOKEN keyword or by the IXCMMSGC QUERYMSG DATATYPE=MSGOUT STATUS=INCOMPLETE service.</p> <p><b>Action:</b> Verify the token and retry the request with the correct token. Also, check to see if your program inadvertently overlaid the parameter list in storage.</p>
8	xxxx0224	<p><b>Equate Symbol:</b> IXCMMSGCRSNTOKENNOTFORRELEASEMSG</p> <p><b>Meaning:</b> The input message token identified by TOKEN or SENDTOKEN is not valid for Release Message (RELEASEMSG). The message token must be a token that was returned by the IXCMMSGOX service via the RETMSGOTOKEN keyword or a send token that was returned by the IXCSEND service via the RETMSGTOKEN keyword.</p> <p><b>Action:</b> Verify the token and retry the request with the correct message token.</p>
8	xxxx0308	<p><b>Equate Symbol:</b> IXCMMSGCRSNBADRETMSGTOKENALET</p> <p><b>Meaning:</b> The ALET that qualifies the address of the RETMSGTOKEN is not zero, nor a valid entry on the dispatchable unit access list (DU-AL), nor a valid entry for a common area data space.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• Your program is not intended to run in primary ASC mode.</li> <li>• You specified SYSSTATE ASCENV=AR before issuing the IXCMMSGC macro.</li> <li>• The ALET for the parameter list is a valid public entry on the DU-AL, is zero (primary address space ALET), or for a common area data space.</li> </ul>
8	xxxx0309	<p><b>Equate Symbol:</b> IXCMMSGCRSNBADRETMSGTOKENADDRESS</p> <p><b>Meaning:</b> The RETMSGTOKEN was not accessible. The message control service was not able to store a message token in the storage area specified by RETMSGTOKEN.</p> <p><b>Action:</b> Verify the storage area is accessible to XCF and retry the request.</p>
8	xxxx030A	<p><b>Equate Symbol:</b> IXCMMSGCRSNBADEXITFORCALLEXIT</p> <p><b>Meaning:</b> For a CALLEXIT request, XCF attempted to call the user routine but the message or message notify user routine abended. The user routine address might not be valid or the user routine might have done some processing. The specified message might have been processed by the user routine before it abended. As such, the token might or might not specify a currently valid message.</p> <p><b>Action:</b> If necessary, perform any recovery action and resource cleanup for the user routine.</p> <p>Verify the user routine address and retry the CALLEXIT request.</p>
8	xxxx030E	<p><b>Equate Symbol:</b> IXCMMSGCRSNTASKMODECALLEXITWITHFRR</p> <p><b>Meaning:</b> For a CALLEXIT request that was made in task mode, the caller had an FRR established.</p> <p><b>Action:</b> Correct your program so that it does not issue an IXCMMSGC CALLEXIT request with FRRs established while in task mode.</p>

Table 11. Return and Reason Codes for the IXCMMSGC Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0C14	<b>Equate Symbol:</b> IXCMMSGCRSNTOKENNOTFORQUERYMSG <b>Meaning:</b> Environmental error. TOKEN not valid for IXCMMSGC QUERYMSG service. <b>Action:</b> Retry the request with the correct token.
8	xxxx0C1C	<b>Equate Symbol:</b> IXCMMSGCRSNBADMQAALevel <b>Meaning:</b> MQAALevel specified for the QUERYMSG request was not valid. <b>Action:</b> Retry the request with a MQAALevel that is supported by the local system and by the DATATYPE. The IXCYMQAA macro documents the supported MQAALevel.
8	xxxx0C1D	<b>Equate Symbol:</b> IXCMMSGCRSNHOLDTIMENOTSET <b>Meaning:</b> HOLDTIME(CONTINUE) was specified for a message that had not previously been saved. <b>Action:</b> Retry the request specifying HOLDTIME(SET) or omit the HOLDTIME(CONTINUE) keyword to allow the message to be saved indefinitely.
C	xxxx0C1E	<b>Equate Symbol:</b> IXCMMSGCRSNHOLDTIMEINVALID <b>Meaning:</b> HOLDTIME(SET) was specified with an invalid TIME value. TIME must be a non-zero value. <b>Action:</b> Use the message control DISCARDMSG service to discard one or more messages in order to make more storage available.
C	xxxx0C0C	<b>Equate Symbol:</b> IXCMMSGCRSNDUALCANNOTBEEXPANDED <b>Meaning:</b> Specify a TIME value that is non-zero. <b>Action:</b> Retry the request at a later time or remove an entry from the DU-AL and retry the request.
C	xxxx0C10	<b>Equate Symbol:</b> IXCMMSGCRSNNOWORKINGSTORAGE <b>Meaning:</b> Environmental error. An IXCMMSGC request could not be performed because XCF could not obtain enough working storage in the caller's address space to process the request. <b>Action:</b> Retry your request at a later time.
C	xxxx0C18	<b>Equate Symbol:</b> IXCMMSGCRSNMSGPENDING <b>Meaning:</b> Environmental error. The CALLEXIT service is not valid for a message-out request that is not yet completed. <b>Action:</b> Reissue the CALLEXIT request after the message is completed. Use the IXCMMSGC COMPLETION service to force the message to be considered complete, if necessary.
10	None	<b>Meaning:</b> System failure. XCF processing failure. <b>Action:</b> Save the return and reason code information and contact the IBM Support Center.

---

## Chapter 13. IXCMSTGI — Receive a Message from Another Member in the XCF Group

**Note:** IBM suggests using the IXCMSTGIX macro interface service for XCF message-in service requests. IXCMSTGIX contains functions that the IXCMSTGI macro interface does not have. For details about the IXCMSTGIX macro, see [Chapter 14, “IXCMSTGIX — Receive a Message from Another Member in the XCF Group,” on page 157.](#)



## Chapter 14. IXCMSGIX — Receive a Message from Another Member in the XCF Group

### Description

The IXCMSGIX macro is the interface to the XCF (cross-system coupling facility) message-in service. This interface is the successor to the IXCMGSI macro interface, contains all the functionality of the IXCMGSI message-in macro interface and is the suggested XCF message-in service interface to use.

The IXCMSGIX macro allows a member of an XCF group to receive a message or response from another member in its XCF group. IXCMSGIX, and its companion services, IXCMGGOX (message-out service) and IXCMGSC (message control service), comprise the XCF signalling services. To receive messages and responses through XCF signalling services, you must write either a message user routine or a message notify user routine that invokes the IXCMSGIX macro. When an XCF member sends a message or response by issuing the IXCMGGOX macro, XCF schedules either the message user routine or the message notify user routine belonging to the member receiving the message.

Messages can vary in length up to 128M bytes. A “large message” is defined to be one that is greater than 62464 (61K) bytes. XCF handles large messages somewhat differently from messages of less than 61K bytes, and also imposes the following restrictions when sending large messages:

- Both the sending and the target **systems** must be running OS/390 Release 8 or higher. (This does not necessarily imply that all **members** residing on those systems support large message delivery.)
- Both the sending and the target **members** must have specified when they joined the XCF group that they supported large message delivery. (Each must have specified GT61KMSG=YES on the IXCJOIN invocation.)

Members of an XCF group can receive message data using either a single buffer or multiple buffers.

Be sure to read the IXCMSGIX guidance information in *z/OS MVS Programming: Sysplex Services Guide*. The information presented here assumes you have done so. IXCMSGIX guidance information includes:

- Illustrations showing the message data element formats specified by the various IXCMSGIX parameters.
- Information on designing a protocol for sending and receiving multipart messages.
- Information on writing a message user routine and a message notify user routine.

### Environment

The following are the environment requirements for the caller.

Environment	Environment requirement
Minimum authorization:	Supervisor state or PKM allowing key 0-7
Dispatchable unit mode:	Task or SRB
Cross memory mode:	HASN=PASN, any SASN. The primary address space must equal the primary address space of the caller of the IXCJOIN macro when it was issued to join the XCF group. IXCMSGIX must be invoked under the work unit of the message or message notify user routine.

Environment	Environment requirement
AMODE:	<p>31-bit or 64-bit. To pass keyword values that name and reference 64-bit virtual storage, specify SYSSTATE AMODE64=YES before invoking the IXCMSGIX macro service.</p> <p>The following IXCMSGIX macro parameters may reside in either 31-bit or 64-bit addressable virtual storage:</p> <ul style="list-style-type: none"> <li>• MSGBUF</li> <li>• ELEMENT</li> <li>• PARTLENTBL</li> <li>• PARTALETTBL</li> <li>• The TABLE or QUEUE of message elements that describe the message parts of a MULTIPART message. In addition, the buffer addresses and next message data element address for a message data element queue, which the message data elements contain, may reference 31-bit or 64-bit storage</li> <li>• An area to be used for the parameter list of the IXCMSGIX macro</li> </ul>
ASC mode:	Primary or access register (AR)
Interrupt status:	Enabled for I/O and external interrupts
Locks:	No locks held
Control parameters:	Must be in the primary address space of the caller and might be in the 31-bit or 64-bit addressable virtual storage.

## Programming Requirements

- If the program is in AR mode, issue the SYSSTATE ASCENV=AR macro before issuing IXCMSGIX to generate code for AR mode.
- If the parameter list specified for the macro service call resides in 64-bit virtual storage above the 2-gigabyte bar, the caller must be executing in AMODE 64 when invoking the IXCMSGIX macro service.
- The IXCYMEPL mapping macro provides the format of the message exit parameter list (MEPL) that XCF passes to the message user routine. Include that mapping macro in the code for the message user routine. For more information about the MEPL, see *z/OS MVS Data Areas* in the *z/OS Internet library* ([www.ibm.com/servers/resourcelink/svc00100.nsf/pages/zosInternetLibrary](http://www.ibm.com/servers/resourcelink/svc00100.nsf/pages/zosInternetLibrary)).
- If you specify the MSGBUF parameter, the buffer it designates must be large enough to receive the entire message. If the buffer is too small you will either receive a program check or overwrite storage.
- For MULTIPART=YES, the invoker of IXCMSGIX is responsible for maintaining the integrity of the element structure until the message-in service returns. If the elements or their structure change while being processed by the message-in service, the message data actually received may be corrupted.
- All message data elements (but not buffers whose addresses are located in the message data elements) must reside in the same address space or data space so they can be accessed using the same ALET.

## Restrictions

- The IXCMSGIX macro can be issued within a message user routine or message notify user routine that XCF schedules on behalf of the receiving member or to which XCF gives control on behalf of the IXCMSGC CALLEXIT service.
- The sending and receiving members must be active members of the same XCF group.

## Input Register Information

---

Before issuing the IXCMMSGIX macro, the caller does not have to place any information into any register unless using it in register notation for a particular parameter, or using it as a base register.

## Output Register Information

---

When control returns to the caller, the GPRs contain:

### Register Contents

- 0**  
Reason code, if GPR 15 contains a non-zero return code that has applicable reason codes.
- 1**  
Used as a work register by the system.
- 2-13**  
Unchanged.
- 14**  
Used as a work register by the system.
- 15**  
Return code.

When control returns to the caller, the access registers (ARs) contain:

### Register Contents

- 0-1**  
Used as work registers by the system
- 2-13**  
Unchanged
- 14-15**  
Used as work registers by the system

Some callers depend on register contents remaining the same before and after issuing a service. If the system changes the contents of registers on which the caller depends, the caller must save them before issuing the service, and restore them after the system returns control.

## Performance Implications

---

None.

## Understanding IXCMMSGIX Version Support

---

The IXCMMSGIX macro supports versions in the range of 4 - 6.

- Keywords not specifically noted here are supported by version 4 and subsequent versions of the IXCMMSGIX macro.
- The following keywords and functions are supported by version 5 and subsequent versions of the IXCMMSGIX macro.
  - ELEMADDRMODE
  - ELEMENT
  - ELEMFORM
  - ENDOFQUEUE
  - MULTIPART

- NEXTOFF
- NEXTPTROFF
- NEXTPTROFF
- PARTALET
- PARTALETTOFF
- PARTALETTLBL
- PARTLEN
- PARTLENOFF
- PARTLENTBL
- PARTOFF
- PARTPTROFF
- #MSGPARTS
- The following keywords and functions are supported by version 6 and subsequent versions of the IXCMSGIX macro.
  - MOVECURSOR
  - STARTOFFSET

Specify the version of the parameter list that you want generated with the PLISTVER keyword. See [Chapter 2, “Specifying a Macro Version Number,” on page 5](#) for considerations when specifying the version of the parameter list with PLISTVER.

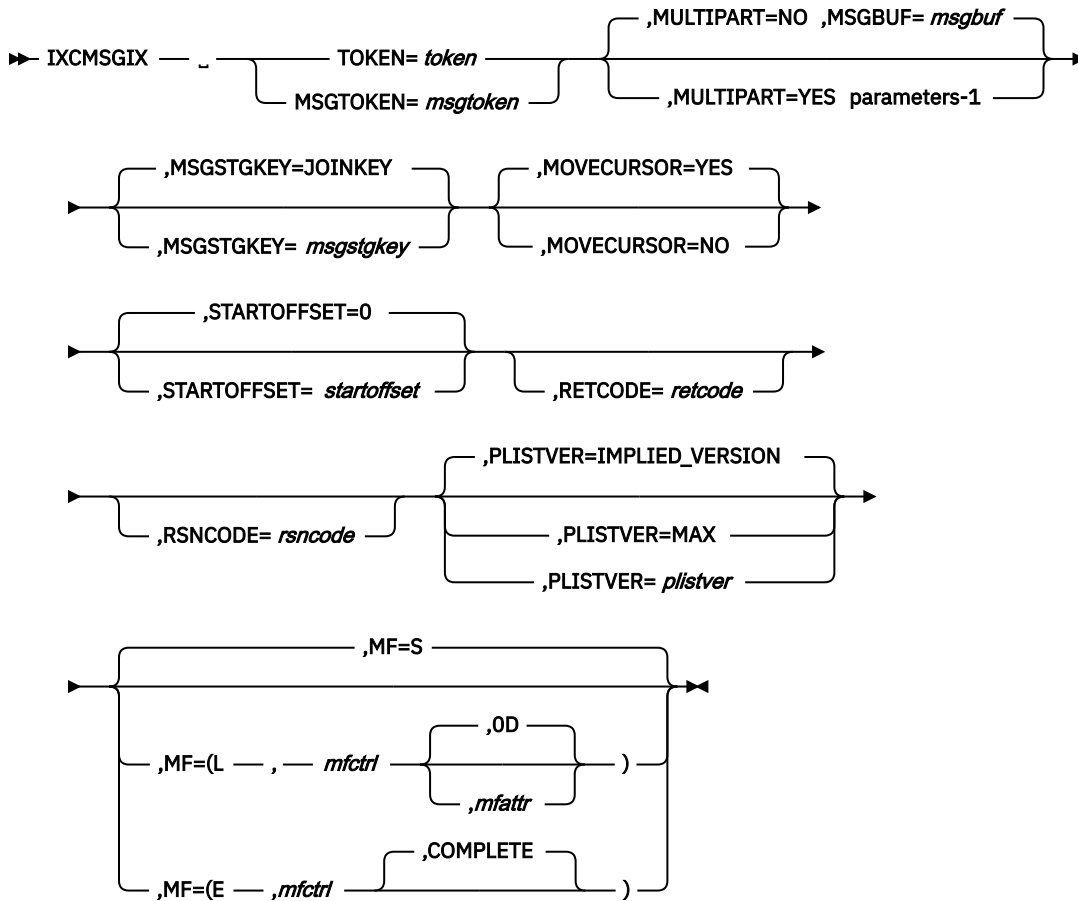
## Syntax Diagram

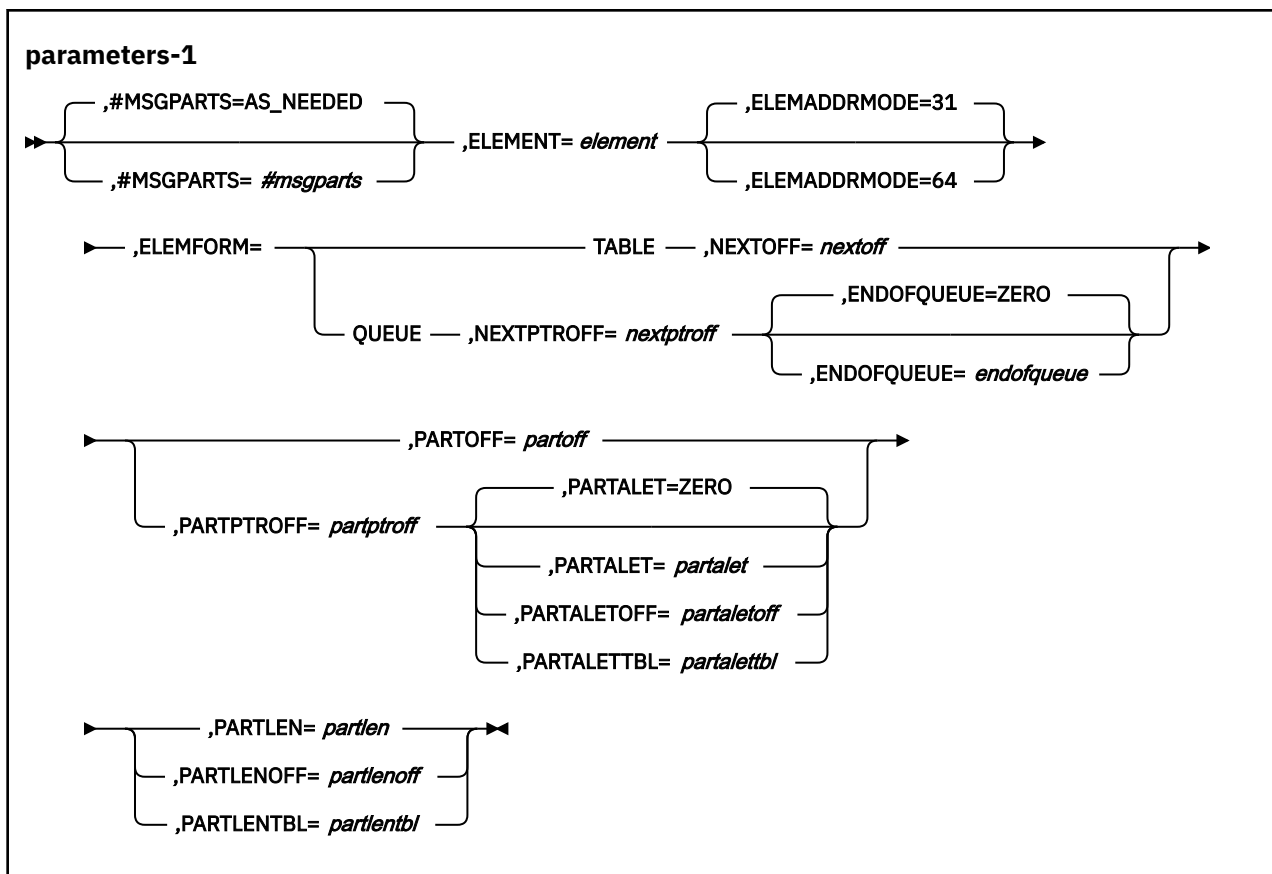
---

The syntax of the IXCMSGIX macro is as follows:



## main diagram





## Parameter Descriptions

The parameter descriptions are listed in alphabetical order. Default values are underlined:

### **,ELEMADDRMODE=31**

### **,ELEMADDRMODE=64**

Use this parameter to specify the precision format of the addresses located within the message data element at the offsets specified by the PARTPTROFF and NEXTPTROFF keywords. The default is ELEMADDRMODE=31.

Use ELEMADDRMODE=31 to indicate that all of the addresses in message data elements are of 31-bit addressing precision (single word - 4 bytes) and reference only virtual storage below the 2-gigabyte virtual storage bar.

Use ELEMADDRMODE=64 to indicate that all of the addresses in message data elements are of 64-bit addressing precision (double word - 8 bytes) and can reference virtual storage below or above the 2-gigabyte virtual storage bar.

### **,ELEMENT=element**

Use this input parameter to specify the first element of the table or queue of message data elements. Message data elements contain either buffers or pointers to buffers that are to receive parts of the message.

Specifying the PARTOFF parameter indicates that each element contains a buffer. Specifying the PARTPTROFF parameter indicates that each element contains a pointer to a buffer.

Elements must all reside in the same space, either an address space or a data space. Elements can be in the caller's primary address space, addressable through a public entry on the DU-AL, or in a common area data space.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the first element.

**,ELEMFORM=TABLE**

**,ELEMFORM=QUEUE**

Use ELEMFORM=TABLE to indicate that the message data elements are organized as a table.

Use ELEMFORM=QUEUE to indicate that the message data elements are organized as a queue.

**,ENDOFQUEUE=ZERO**

**,ENDOFQUEUE=endofqueue**

Use this input parameter to specify the address that marks the end of the queue. When the pointer to the next element contains this address, queue processing ends. You can use either ENDOFQUEUE or #MSGPARTS to limit the amount of message data elements that are processed. A partial delivery occurs if the number of message data elements is insufficient to contain all the data. To receive more of the data, continue to reissue IXCMMSGIX until all data has been delivered.

**Note:** The queue must have at least one element.

If you omit ENDOFQUEUE, the default value for the end-of-queue address is 0 (you can code this explicitly by specifying ENDOFQUEUE=ZERO).

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword containing the end-of-queue address.

**,MF=S**

**,MF=(L,mfctrl)**

**,MF=(L,mfctrl,mfattr)**

**,MF=(L,mfctrl,0D)**

**,MF=(M,mfctrl)**

**,MF=(M,mfctrl,COMPLETE)**

**,MF=(M,mfctrl,NOCHECK)**

**,MF=(E,mfctrl)**

**,MF=(E,mfctrl,COMPLETE)**

**,MF=(E,mfctrl,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

**,mfctrl**

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

**,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

**,COMPLETE**

**,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

**COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if `SMILE=var` were an optional parameter and the default is `SMILE=NO_SMILE` then it would not be documented. However, if the default was `SMILE=-:-`, then it would be documented because a value would be the default.

**NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

**,MOVECURSOR=YES****,MOVECURSOR=NO**

Use this optional parameter to indicate whether the read position within the message data to be delivered should be advanced or not.

**,MOVECURSOR=YES**

`MOVECURSOR=YES` indicates that you want the read position within the message data being delivered advanced as data is returned to the caller. When the read position reaches the end of the data to be returned, XCF considers the message delivered, and the message token is invalidated.

**,MOVECURSOR=NO**

`MOVECURSOR=NO` indicates that you do not want the position within the data to be delivered advanced as data is returned to the caller. All data returned on a call specifying `MOVECURSOR(NO)` is eligible for re-delivery on a subsequent IXCMMSGIX Message-In service call within the context of the message exit or notify exit or for a message saved using the IXCMMSGC message control service. XCF will return data starting from the last established read position. The IXCMMSGIX message-in service will not invalidate a message token on a call specifying `MOVECURSOR(NO)` even if all message data is returned.

**,MSGBUF=msgbuf**

Use this output parameter to specify the buffer to receive the incoming message. The size of the buffer must be greater than or equal to the value of `MEPLMLEN` in `IXCYMEPL` (if processing in a message user routine) or `MNPLTRRESPMLEN` in `IXCYMNPL` (if processing in a message notify user routine). The storage key of the buffer specified by `MSGBUF` must match the storage key specified with the `MSGSTGKEY` parameter.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a storage area to receive the message.

**MSGTOKEN=msgtoken**

Use the `TOKEN` keyword instead of the `MSGTOKEN` keyword. `MSGTOKEN` is not supported for IXCMMSGIX requests invoked from a message notify user routine. `MSGTOKEN` is provided for compatibility with:

- Message user routines that are running on systems with releases prior to OS/390 Release 3.
- Older applications that have been assembled with pre-OS/390 Release 2 code but that are running on an OS/390 Release 3 or later system.

Otherwise, use this input parameter to specify the message token that your message user routine received from XCF in the `MEPLMTOK` field of the `MEPL` (mapped by `IXCYMEPL`).

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 32-bit field that contains the message token.

**,MSGSTGKEY=JOINKEY****,MSGSTGKEY=msgstgkey**

Use this input parameter to specify the storage key to be used by XCF when storing the message into each receiving buffer. The storage key of each receiving buffer must match the storage key specified by `MSGSTGKEY`.

If you omit the MSGSTGKEY parameter, or if you specify MSGSTGKEY=JOINKEY, XCF uses as the storage key the value of the PSW key at the time you joined the XCF group (when IXCJOIN was issued).

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-bit field formatted as **kkkkxxxx**, where the high-order four bits contain the storage key. The low-order four bits are ignored.

**,MULTIPART=NO**

**,MULTIPART=YES**

Use MULTIPART=NO to indicate that the message is to be received into a single buffer.

Use MULTIPART=YES to indicate that the message is to be received into one or more buffers.

**,NEXTOFF=nextoff**

Use this input parameter to specify the number of bytes to be added to the address of the current element to obtain the address of the next element. This value equals the size in bytes of an individual message data element. It is used when the message data elements are in table form.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword containing the number of bytes to be added.

**,NEXTPTROFF=nextptroff**

Use this input parameter to specify the number of bytes to be added to the address of the current element, to locate within the current element either a 31- or 64-bit precision pointer to the next element. This value is used when the message data elements are in queue form.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword containing the number of bytes.

**,PARTALET=ZERO**

**,PARTALET=partalet**

Use this input parameter to specify a single ALET to be used to qualify every buffer address. The ALET must be zero, a public entry on your dispatchable unit access list (DU-AL), or an entry for a common area data space.

If you omit PARTALET, PARTALETTOFF, and PARTALETTBL, the default is PARTALET with an ALET of 0 (if you would like to code this explicitly, specify PARTALET=ZERO).

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword containing the ALET.

**,PARTALETTOFF=partalettoff**

Use this input parameter to specify the number of bytes to be added to the address of the current element, to locate within the element the fullword field that contains the ALET to be used to qualify the buffer address associated with that element. The ALET must be zero, a public entry on your dispatchable unit access list (DU-AL), or an entry for a common area data space.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword containing the number of bytes.

**,PARTALETTBL=partalettbl**

Use this input parameter to specify a table in which each entry is a fullword containing the ALET to qualify a buffer address in the table or queue of message data elements. For instance, the 3rd entry in PARTALETTBL must contain the ALET to qualify the buffer address in the 3rd message data element. *partalettbl* must begin on a fullword boundary. Each ALET must be zero, a public entry on your dispatchable unit access list (DU-AL), or an entry for a common area data space.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of *partalettbl*.

**,PARTLEN=partlen**

Use this input parameter to specify the length in bytes of each buffer when all the buffers are the same length. All buffers must be at least as long as PARTLEN.

To receive the entire message in a message user routine, the total amount of storage for the message (the value of PARTLEN multiplied by the number of elements) must be greater than or equal to

MEPLMLLEN. For a message notify user routine to receive the entire response, the total amount of storage must be greater than or equal to MNPLTRRESPMLLEN.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword containing the length in bytes of each buffer.

**,PARTLENOFF=partlenoff**

Use this input parameter to specify the number of bytes to be added to the address of the current element, to locate within the element the fullword field that contains the length in bytes of the buffer associated with that element.

The value of the field indicated by PARTLENOFF should contain a value at least as long (but may be less than or equal to the length of the buffer) as the part of the message you want to put in this buffer. If you specify a length of zero for the message part, XCF does not use that buffer.

To receive the entire message, the total amount of storage for the message (the sum of the buffer lengths) must be greater than or equal to MEPLMLLEN. For a message notify user routine to receive an entire response, the total amount of storage must be greater than or equal to MNPLTRRESPMLLEN.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword containing the number of bytes.

**,PARTLENTBL=partlentbl**

Use this input parameter to specify a table in which each entry is a fullword containing the length of a corresponding buffer. For instance, the 3rd entry in PARTLENTBL contains the length of the buffer whose address is associated with the 3rd message data element in the table or queue. If you specify the length of a buffer as zero, XCF does not use that buffer.

The table specified by PARTLENTBL must begin on a fullword boundary.

To receive the entire message, the total amount of storage for the message (the sum of the buffer lengths) must be greater than or equal to MEPLMLLEN. For a message notify user routine to receive the entire response, the total amount of storage must be greater than or equal to MNPLTRRESPMLLEN.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of *partlentbl*.

**,PARTOFF=partoff**

Use this input parameter to specify the number of bytes to be added to the address of the current element, to obtain the address within the element of the start of the buffer associated with it.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword containing the number of bytes.

**,PARTPTROFF=partptroff**

Use this input parameter to specify the number of bytes to be added to the address of the current element, to locate within the element either a 31- or 64-bit precision pointer to the buffer associated with it.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword containing the number of bytes.

**,PLISTVER=IMPLIED\_VERSION**

**,PLISTVER=MAX**

**,PLISTVER=plistver**

Use this input parameter to specify the version of the macro. See [“Understanding IXCMMSGIX Version Support” on page 159](#) for a description of the options available with PLISTVER.

**,RETCODE=retcode**

Use this output parameter to specify the location in which the system is to copy the return code from GPR 15.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword to contain the return code.

**,RSNCODE=*rsncode***

Use this output parameter to specify the location in which the system is to copy the reason code from GPR 0.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword to contain the reason code.

**,STARTOFFSET=0****,STARTOFFSET=*startoffset***

Use this optional input parameter to indicate the offset into the message data from the current read position from which to begin returning data. The value must not exceed the amount of available user message data to be returned. The default is 0.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword containing the ALET.

**TOKEN=*token***

Use this input parameter to specify the location of the 16-character field that contains a token that identifies the message whose message data is to be delivered.

The token identifies one of the following:

- A message presented to a message user routine. On entry to the routine, R1 contains the address of the message exit parameter list (MEPL) that contains the token (MEPLMSGITOKEN).
- A response message presented to a message notify user routine. On entry to the routine, R1 contains the address of the message notification parameter list (MNPL) that contains the token (MNPLTRMSGITOKEN).

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the 16-character field containing the token.

**,#MSGPARTS=AS\_NEEDED****,#MSGPARTS=*msgparts***

Use this input parameter to specify the number of buffers (1 or more) to be used to receive the message. The number of buffers specified determines whether an entire message is received or only part of the data. You can use #MSGPARTS to limit the amount of message data you receive. To receive some but not all of the message or response, specify #MSGPARTS to be less than the number of buffers that would be needed to receive all the data. To receive an entire message or response, you can omit the #MSGPARTS parameter, specify #MSGPARTS=AS\_NEEDED, or specify a number of buffers that is sufficient to contain all the data.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword containing the number of buffers.

## ABEND Codes

---

None.

## Return and Reason Codes

---

When the IXCMMSGIX macro returns control to your program:

- GPR 15 (and *retcode*, if you coded RETCODE) contains a return code.
- GPR 0 (and *rsncode*, if you coded RSNCODE) contains a reason code, if applicable.

Macro IXCYCON provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

**0**

IXCRETCODEOK

**4**

IXCRETCODEWARNING

8

IXCRETCODEPARMERROR

C

IXCRETCODEENVERROR

10

IXCRETCODECOMPERROR

The following table identifies the hexadecimal return and reason codes and the equate symbol associated with each reason code.

Table 12. Return and Reason Codes for the IXCMMSGIX Macro		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
00	None.	<b>Meaning:</b> IXCMMSGIX completed successfully; the message data has been received as specified. <b>Action:</b> None.
04	224	<b>Equate Symbol:</b> IXCMMSGIRSNSTILLMOREDATA <b>Meaning:</b> The total amount of buffer storage was insufficient. The buffers contain as much of the message as could fit. There is more message data to receive. This return code and reason code are only returned if you specified MULTIPART=YES. <b>Action:</b> Continue to reissue the IXCMMSGIX macro to receive the remainder of the message. Alternately, you could save or discard the remainder of the message.
08	04	<b>Equate Symbol:</b> IXCMMSGIRSNMSGBUFBADSTG <b>Meaning:</b> Program error. XCF could not access the buffer specified by MSGBUF. <b>Action:</b> Check for errors such as the following; <ul style="list-style-type: none"> <li>• The buffer address is incorrect.</li> <li>• The buffer was previously freed.</li> <li>• If your program is running in AR mode, check for the following types of errors:               <ul style="list-style-type: none"> <li>– The message buffer ALET is incorrect.</li> <li>– You did not specify SYSSTATE ASCENV=AR before issuing the IXCMMSGIX macro.</li> </ul> </li> </ul>
08	08	<b>Equate Symbol:</b> IXCMMSGIRSNMSGALREADYDELIVERED <b>Meaning:</b> Program error. XCF already delivered the message, and it is no longer available. <b>Action:</b> If this result was acceptable, no action might be necessary. Otherwise, check for errors such as the following: <ul style="list-style-type: none"> <li>• You specified an incorrect message token.</li> <li>• You tried to receive the same message more than once.</li> <li>• Your user routine is not reentrant.</li> </ul>



Table 12. Return and Reason Codes for the IXCMMSGIX Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
08	09	<p><b>Equate Symbol:</b> IXCMMSGIRSNMEMBERNOTACTIVE</p> <p><b>Meaning:</b> Program error. The message is no longer available because the member is no longer active.</p> <p><b>Action:</b> If this result was acceptable, no action might be necessary. Otherwise:</p> <ul style="list-style-type: none"> <li>• Check to see if you specified an incorrect message token.</li> <li>• Have your user routine give up control.</li> </ul>
08	0C	<p><b>Equate Symbol:</b> IXCMMSGIRSNMSGBUFBADALET</p> <p><b>Meaning:</b> Program error. The ALET that qualifies the address of the buffer specified by MSGBUF must be zero, a public entry on your DU-AL, or an entry for a common area data space.</p> <p><b>Action:</b> Check for errors such as the following:</p> <ul style="list-style-type: none"> <li>• You did not specify SYSSTATE ASCENV=AR before issuing the IXCMMSGIX macro.</li> <li>• You specified an incorrect buffer area ALET.</li> </ul>
08	40	<p><b>Equate Symbol:</b> IXCMMSGIRSNPLISTRSDNOTVALID</p> <p><b>Meaning:</b> Program error. A reserved field in the control parameter list was not zero.</p> <p><b>Action:</b> Check for errors such as the following:</p> <ul style="list-style-type: none"> <li>• Your program overlaid the control parameter list storage.</li> <li>• Your program was assembled with the wrong macro library for the release of z/OS on which your program is running.</li> </ul>
08	44	<p><b>Equate Symbol:</b> IXCMMSGIRSNMSGTOKENNOTVALID</p> <p><b>Meaning:</b> Program error. The message token was not valid.</p> <p><b>Action:</b> Check for errors such as the following:</p> <ul style="list-style-type: none"> <li>• The message token you specified was not the one contained in the MEPLMTOK field of the MEPL, the MEPLMSGITOKEN field of the MEPL, or the MNPLTRMSGITOKEN field of the MNPL.</li> <li>• Your program overlaid the control parameter list storage.</li> <li>• Your program is not reentrant.</li> </ul>
08	45	<p><b>Equate Symbol:</b> IXCMMSGIRSNUSETOKENKEYWORD</p> <p><b>Meaning:</b> Program error. MSGTOKEN is not valid. Use the TOKEN keyword.</p> <p><b>Action:</b> Correct your program to use the TOKEN keyword, which is required when invoking a message user routine through a message notify user routine.</p>

Table 12. Return and Reason Codes for the IXCMMSGIX Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
08	100	<p><b>Equate Symbol:</b> IXCMMSGIRSNPLISTBADALET</p> <p><b>Meaning:</b> Program error. The ALET that qualified the address of the control parameter list was not zero.</p> <p><b>Action:</b> Check for errors such as the following:</p> <ul style="list-style-type: none"> <li>You did not specify SYSSTATE ASCENV=AR before issuing the IXCMMSGIX macro.</li> <li>The ALET for the parameter list storage was not zero (primary address space ALET).</li> </ul> <p><b>Note:</b> The ALET for the control parameter list must be zero.</p>
08	104	<p><b>Equate Symbol:</b> IXCMMSGIRSNPLISTVERSIONNOTVALID</p> <p><b>Meaning:</b> Program error. The version number in the control parameter list was not valid.</p> <p><b>Action:</b> Check for errors such as the following:</p> <ul style="list-style-type: none"> <li>Your program overlaid the control parameter list storage.</li> <li>Your program was assembled with the wrong macro library for the release of z/OS on which your program is running.</li> </ul>
08	10C	<p><b>Equate Symbol:</b> IXCMMSGIRSNPLISTBADSTG</p> <p><b>Meaning:</b> Program error. The control parameter list could not be accessed.</p> <p><b>Action:</b> Check for errors such as the following:</p> <ul style="list-style-type: none"> <li>You specified an incorrect address for the control parameter list.</li> <li>The control parameter list storage area had been previously freed.</li> </ul>
08	20C	<p><b>Equate Symbol:</b> IXCMMSGIRSNMSGBUFSTGKEYMISMATCH</p> <p><b>Meaning:</b> Program error. The service could not store the message data into the buffer area specified by MSGBUF using the storage key specified by MSGSTGKEY or the default storage key (if MSGSTGKEY was not specified, or if MSGSTGKEY=JOINKEY was specified).</p> <p><b>Action:</b> Check for errors such as the following:</p> <ul style="list-style-type: none"> <li>The buffer area was allocated in a storage key that did not match the key specified using the MSGSTGKEY parameter.</li> <li>MSGSTGKEY was not specified, or MSGSTGKEY=JOINKEY was specified, and the buffer area was allocated in a storage key that did not match the key of the caller of IXCJOIN.</li> </ul>

Table 12. Return and Reason Codes for the IXCMMSGIX Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
08	20D	<p><b>Equate Symbol:</b> IXCMMSGIRSNMSGBUFPAGEPROTECT</p> <p><b>Meaning:</b> Program error. The buffer specified by MSGBUF was page-protected (read-only) so the message data could not be placed in the buffer area.</p> <p><b>Action:</b> Provide IXCMMSGIX with buffer storage that is not page-protected.</p>
08	210	<p><b>Equate Symbol:</b> IXCMMSGIRSNPARTPTROFFBADSTG</p> <p><b>Meaning:</b> Program error. The element is not accessible. The address within an element at the offset specified by PARTPTROFF is not accessible. In the parameter list generated by IXCMMSGIX:</p> <ul style="list-style-type: none"> <li>• The field, PART#, indicates the index of the element containing the invalid address. PART# values start with 1.</li> <li>• The field, ELEMENTPTR, contains the address of the element containing the invalid address.</li> </ul> <p><b>Action:</b> Check for errors such as the following:</p> <ul style="list-style-type: none"> <li>• You specified an incorrect element address.</li> <li>• Your program overlaid the element address.</li> <li>• The element had been previously freed.</li> <li>• You specified an incorrect number of buffers to receive the message.</li> <li>• The message data element is damaged or has not maintained the integrity of its data.</li> <li>• You specified an incorrect end-of-queue value.</li> <li>• If your program is running in AR mode, check for the following: <ul style="list-style-type: none"> <li>– You specified the wrong ALET for the buffer.</li> <li>– You did not specify SYSSTATE ASCENV=AR before issuing the IXCMMSGIX macro.</li> </ul> </li> </ul>
08	212	<p><b>Equate Symbol:</b> IXCMMSGIRSNELEMENTBADALET</p> <p><b>Meaning:</b> Program error. The ALET that qualifies the address of an ELEMENT must be zero, a public entry on your DU-AL or an entry for a common area data space.</p> <p><b>Action:</b> Check for errors such as the following:</p> <ul style="list-style-type: none"> <li>• You did not specify SYSSTATE ASCENV=AR before issuing the IXCMMSGIX macro.</li> <li>• You specified the wrong ALET for the element.</li> </ul>

Table 12. Return and Reason Codes for the IXCMMSGIX Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
08	213	<p><b>Equate Symbol:</b> IXCMMSGIRSNNEXTPTROFFBADSTG</p> <p><b>Meaning:</b> Program error. The next element pointer at offset NEXTPTROFF within an ELEMENT is not accessible. In the parameter list generated by IXCMMSGIX:</p> <ul style="list-style-type: none"> <li>• The field, PART#, indicates the index of the ELEMENT containing the incorrect NEXTPTROFF field. PART# values start with 1.</li> <li>• The field, ELEMENTPTR, contains the address of the element containing the incorrect address.</li> </ul> <p><b>Action:</b> Check for errors such as the following:</p> <ul style="list-style-type: none"> <li>• You specified an incorrect element address.</li> <li>• Your program overlaid the element address.</li> <li>• You specified an incorrect number of buffers to receive the message.</li> <li>• The message data element is damaged or has not maintained the integrity of its data.</li> <li>• You specified an incorrect end-of-queue value.</li> </ul>
08	214	<p><b>Equate Symbol:</b> IXCMMSGIRSN#MSGPARTSZERO</p> <p><b>Meaning:</b> Program error. The value specified by the #MSGPARTS parameter was zero but must be greater than zero.</p> <p><b>Action:</b> Specify a value greater than zero for #MSGPARTS or omit the #MSGPARTS parameter.</p>
08	215	<p><b>Equate Symbol:</b> IXCMMSGIRSNTOOMANYZEROLENPARTS</p> <p><b>Meaning:</b> Program error. The message data element table or queue is assumed to be damaged. You omitted the #MSGPARTS parameter, and XCF processed more than 65,536 consecutive elements with buffers of length zero.</p> <p><b>Action:</b> Check for errors such as the following:</p> <ul style="list-style-type: none"> <li>• You specified PARTLEN with a value of zero.</li> <li>• You specified PARTLENTBL but the address of the table was not correct.</li> <li>• You specified PARTLENOFF with an incorrect value.</li> <li>• You specified an incorrect end-of-queue value.</li> </ul> <p>Ensure that the elements have not been corrupted.</p> <p>Specify the #MSGPARTS parameter when invoking IXCMMSGIX.</p>

Table 12. Return and Reason Codes for the IXCMMSGIX Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
08	218	<p><b>Equate Symbol:</b> IXCMMSGIRSNPARTPTROFF@BADSTG</p> <p><b>Meaning:</b> Program error. A buffer whose address was at the offset specified by PARTPTROFF could not be accessed. In the parameter list generated by IXCMMSGIX:</p> <ul style="list-style-type: none"> <li>• The field, PART#, indicates the index of the element containing the incorrect address. PART# values start with 1.</li> <li>• The field, ELEMENTPTR, contains the address of the element containing the incorrect address.</li> </ul> <p><b>Action:</b> Check for errors such as the following:</p> <ul style="list-style-type: none"> <li>• The buffer address is incorrect.</li> <li>• The buffer was previously freed.</li> <li>• You specified PARTPTROFF with an incorrect value.</li> <li>• If your program is running in AR mode, check for the following types or errors: <ul style="list-style-type: none"> <li>– You specified the wrong ALET for the buffer.</li> <li>– You did not specify SYSSTATE ASCENV=AR before issuing the IXCMMSGIX macro.</li> </ul> </li> </ul>
08	219	<p><b>Equate Symbol:</b> IXCMMSGIRSNPARTOFF@BADSTG</p> <p><b>Meaning:</b> Program error. A buffer located at the offset specified by PARTOFF could not be accessed. In the parameter list generated by IXCMMSGIX:</p> <ul style="list-style-type: none"> <li>• The field, PART#, indicates the index of the element containing the buffer that could not be accessed. PART# values start with 1.</li> <li>• The field, ELEMENTPTR, contains the address of the element containing the buffer that could not be accessed.</li> </ul> <p><b>Action:</b> Check for errors such as the following:</p> <ul style="list-style-type: none"> <li>• The storage containing the element has been freed.</li> <li>• The PARTOFF value is incorrect.</li> <li>• The storage containing the element has been overlaid.</li> </ul>

Table 12. Return and Reason Codes for the IXCMMSGIX Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
08	21A	<p><b>Equate Symbol:</b> IXCMMSGIRSNPARTPTROFF@PAGEPROTECT</p> <p><b>Meaning:</b> Program error. A buffer whose address was at the offset specified by PARTPTROFF could not be used because it was page-protected (read-only). In the parameter list generated by IXCMMSGIX:</p> <ul style="list-style-type: none"> <li>• The field, PART#, indicates the index of the element containing the address of the page-protected buffer. PART# values start with 1.</li> <li>• The field, ELEMENTPTR, contains the address of the element containing the address of the page-protected buffer.</li> </ul> <p><b>Action:</b> Provide IXCMMSGIX with buffer storage that is not page-protected.</p>
08	21B	<p><b>Equate Symbol:</b> IXCMMSGIRSNPARTOFFPAGEPROTECT</p> <p><b>Meaning:</b> Program error. A buffer located at the offset specified by PARTOFF could not be used because it was page-protected (read-only). In the parameter list generated by IXCMMSGIX:</p> <ul style="list-style-type: none"> <li>• The field, PART#, indicates the index of the element containing the page-protected buffer. PART# values start with 1.</li> <li>• The field, ELEMENTPTR, contains the address of the element containing the page-protected buffer.</li> </ul> <p><b>Action:</b> Provide IXCMMSGIX with buffer storage that is not page-protected.</p>
08	21C	<p><b>Equate Symbol:</b> IXCMMSGIRSNPARTPTROFF@KEYMISMATCH</p> <p><b>Meaning:</b> Program error. XCF could not store the message data into a buffer whose address was at the offset specified by PARTPTROFF using the storage key specified by MSGSTGKEY or the default storage key (if MSGSTGKEY was not specified, or if MSGSTGKEY=JOINKEY was specified). In the parameter list generated by IXCMMSGIX:</p> <ul style="list-style-type: none"> <li>• The field, PART#, indicates the index of the element containing the address of the buffer that could not be accessed. PART# values start with 1.</li> <li>• The field, ELEMENTPTR, contains the address of the element containing the address of the buffer that could not be accessed.</li> </ul> <p><b>Action:</b> Check for errors such as the following:</p> <ul style="list-style-type: none"> <li>• The buffer was allocated in a storage key that did not match the key specified using the MSGSTGKEY parameter.</li> <li>• MSGSTGKEY was not specified or if MSGSTGKEY=JOINKEY was specified, and the buffer area was allocated in a storage key that did not match the key of the caller of IXCJOIN.</li> </ul>

Table 12. Return and Reason Codes for the IXCMMSGIX Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
08	21D	<p><b>Equate Symbol:</b> IXCMMSGIRSNPARTOFFKEYMISMATCH</p> <p><b>Meaning:</b> Program error. XCF could not store the message data into a buffer area located at the offset specified by PARTOFF using the storage key specified by MSGSTGKEY or the default storage key (if MSGSTGKEY was not specified or if MSGSTGKEY=JOINKEY was specified). In the parameter list generated by IXCMMSGIX:</p> <ul style="list-style-type: none"> <li>• The field, PART#, indicates the index of the element containing the buffer that could not be accessed. PART# values start with 1.</li> <li>• The field, ELEMENTPTR, contains the address of the element containing the buffer that could not be accessed.</li> </ul> <p><b>Action:</b> Check for errors such as the following:</p> <ul style="list-style-type: none"> <li>• The buffer area was allocated in a storage key that did not match the key specified using the MSGSTGKEY parameter.</li> <li>• MSGSTGKEY was not specified, or if MSGSTGKEY=JOINKEY was specified, and the buffer area was allocated in a storage key that did not match the key of the caller of IXCJOIN.</li> </ul>
08	220	<p><b>Equate Symbol:</b> IXCMMSGIRSNPARTLENTBLBADSTG</p> <p><b>Meaning:</b> Program error. The table specified by PARTLENTBL could not be accessed.</p> <p><b>Action:</b> Check for errors such as the following:</p> <ul style="list-style-type: none"> <li>• PARTLENTBL was inadvertently freed.</li> <li>• The element was overlaid.</li> <li>• You specified an incorrect address for the table specified by PARTLENTBL.</li> </ul>
08	221	<p><b>Equate Symbol:</b> IXCMMSGIRSNPARTLENTBLNOTWORDBDY</p> <p><b>Meaning:</b> Program error. The table specified by PARTLENTBL does not begin on a fullword boundary.</p> <p><b>Action:</b> Ensure that the table specified by PARTLENTBL begins on a fullword boundary.</p>
08	222	<p><b>Equate Symbol:</b> IXCMMSGIRSNPARTLENTBLBADALET</p> <p><b>Meaning:</b> Program error. The ALET that qualifies the address of PARTLENTBL must be zero, a public entry on your DU-AL, or an entry for a common area data space.</p> <p><b>Action:</b> Check for errors such as the following:</p> <ul style="list-style-type: none"> <li>• You did not specify SYSSTATE ASCENV=AR before issuing the IXCMMSGIX macro.</li> <li>• You specified the wrong ALET for PARTLENTBL.</li> </ul>

Table 12. Return and Reason Codes for the IXCMMSGIX Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
08	223	<p><b>Equate Symbol:</b> IXCMMSGIRSNPARTLENOFFBADSTG</p> <p><b>Meaning:</b> Program error. The buffer length located at the offset specified by PARTLENOFF could not be accessed within an ELEMENT. In the parameter list generated by IXCMMSGIX:</p> <ul style="list-style-type: none"> <li>• The field, PART#, indicates the index of the element containing the buffer length that could not be accessed. PART# values start with 1.</li> <li>• The field, ELEMENTPTR, contains the address of the element containing the buffer length that could not be accessed.</li> </ul> <p><b>Action:</b> Check for following types of errors:</p> <ul style="list-style-type: none"> <li>• The storage containing the element has been freed.</li> <li>• The PARTLENOFF value is incorrect.</li> <li>• The storage containing the element has been overlaid.</li> <li>• You specified an incorrect number of buffers to receive the message.</li> <li>• The message data element is damaged or has not maintained the integrity of its data.</li> <li>• You specified an incorrect end-of-queue value.</li> </ul>
08	230	<p><b>Equate Symbol:</b> IXCMMSGIRSNPARTALETTLBADSTG</p> <p><b>Meaning:</b> Program error. The table specified by PARTALETTL could not be accessed.</p> <p><b>Action:</b> Check for errors such as the following:</p> <ul style="list-style-type: none"> <li>• PARTALETTL was inadvertently freed.</li> <li>• PARTALETTL was overlaid or incorrect.</li> <li>• You specified an incorrect address for the table specified by PARTALETTL.</li> </ul>
08	231	<p><b>Equate Symbol:</b> IXCMMSGIRSNPARTALETTLNOTWORDBDY</p> <p><b>Meaning:</b> Program error. The table specified by PARTALETTL does not begin on a fullword boundary.</p> <p><b>Action:</b> Ensure that the table specified by PARTALETTL begins on a fullword boundary.</p>
08	232	<p><b>Equate Symbol:</b> IXCMMSGIRSNPARTALETTLBADALET</p> <p><b>Meaning:</b> Program error. The ALET that qualifies the address of PARTALETTL must be zero, a public entry on your DU-AL, or an entry for a common area data space.</p> <p><b>Action:</b> Check for errors such as the following:</p> <ul style="list-style-type: none"> <li>• You did not specify SYSSTATE ASCENV=AR before issuing the IXCMMSGIX macro.</li> <li>• You specified the wrong ALET for PARTALETTL.</li> </ul>



Table 12. Return and Reason Codes for the IXCMMSGIX Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
08	233	<p><b>Equate Symbol:</b> IXCMMSGIRSNPARTALETOFFBADSTG</p> <p><b>Meaning:</b> Program error. The ALET located at the offset specified by PARTALETOFF could not be accessed. In the parameter list generated by IXCMMSGIX:</p> <ul style="list-style-type: none"> <li>• The field, PART#, indicates the index of the element containing the ALET that could not be accessed. PART# values start with 1.</li> <li>• The field, ELEMENTPTR, contains the address of the element containing the ALET that could not be accessed.</li> </ul> <p><b>Action:</b> Check for errors such as the following:</p> <ul style="list-style-type: none"> <li>• The storage containing the element has been freed.</li> <li>• The PARTALETOFF value is incorrect.</li> <li>• The storage containing the element has been overlaid.</li> <li>• You specified an incorrect number of buffers to receive the message.</li> <li>• The message data element is damaged or has not maintained the integrity of its data.</li> <li>• You specified an incorrect end-of-queue value.</li> </ul>
08	234	<p><b>Equate Symbol:</b> IXCMMSGIRSNPARTALET@BADALET</p> <p><b>Meaning:</b> Program error. The ALET specified by PARTALET was not valid. The ALET must be zero, a valid public entry on your DU-AL, or an entry for a common area data space.</p> <p><b>Action:</b> Specify an ALET that is zero, a public entry on your DU-AL, or an entry for a common area data space.</p>
08	235	<p><b>Equate Symbol:</b> IXCMMSGIRSNPARTALET@BADALET</p> <p><b>Meaning:</b> Program error. An ALET specified in a PARTALET@BADALET entry was not valid. Each ALET must be zero, a public entry on your DU-AL, or an entry for a common area data space. In the parameter list generated by IXCMMSGIX:</p> <ul style="list-style-type: none"> <li>• The field, PART#, indicates the index of the element containing the ALET that was not valid. PART# values start with 1.</li> <li>• The field, ELEMENTPTR, contains the address of the element containing the ALET that was not valid.</li> </ul> <p><b>Action:</b> Correct the ALET.</p>

Table 12. Return and Reason Codes for the IXCMMSGIX Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
08	236	<p><b>Equate Symbol:</b> IXCMMSGIRSNPARTALETOFF@BADALET</p> <p><b>Meaning:</b> Program error. An ALET specified at the offset designated by PARTALETOFF was not valid. Each ALET must be zero, a public entry on your DU-AL, or an entry for a common area data space. In the parameter list generated by IXCMMSGIX:</p> <ul style="list-style-type: none"> <li>• The field, PART#, indicates the index of the element containing the ALET that was not valid. PART# values start with 1.</li> <li>• The field, ELEMENTPTR, contains the address of the element containing the ALET that was not valid.</li> </ul> <p><b>Action:</b> Correct the ALET.</p> <p>Check for errors such as the following:</p> <ul style="list-style-type: none"> <li>• You specified an incorrect number of buffers to receive the message.</li> <li>• The message data element is damaged or has not maintained the integrity of its data.</li> <li>• You specified an incorrect end-of-queue value.</li> </ul>
08	237	<p><b>Equate Symbol:</b> IXCMMSGIXRSNSTARTOFFSETBADVALUE</p> <p><b>Meaning:</b> Program error. The specified STARTOFFSET value places the starting read position beyond the range of available user message data.</p> <p><b>Action:</b> Ensure that STARTOFFSET indicates a value that is less than or equal to the available user message data to be returned. Note that if partial data was returned on a prior IXCMMSGIX call, the amount of available user message data is MEPLMLEN less the amount of data already returned.</p>
08	010Cxxxx	<p><b>Equate Symbol:</b> IXCMMSGIRSNPLISTNOPARTINFOBADSTG</p> <p><b>Meaning:</b> Program error. A parameter specifying information on a request with MULTIPART=YES was not valid. The system was unable to store the erroneous information in the PART# and ELEMENTPTR fields in the parameter list generated by IXCMMSGIX because it could not access the parameter list. The low-order half word of the reason code (xxxx) indicates the reason code for the error that would have been returned if the parameter list had been accessible.</p> <p><b>Action:</b> The control parameters must be in the primary address space. Make sure that the storage for the parameter list was not inadvertently freed.</p>
10	None.	<p><b>Meaning:</b> System error. XCF processing failed.</p> <p><b>Action:</b> Retry the request one or more times. If the problem persists, record the return and reason code and supply it to the appropriate IBM support personnel.</p>

## Example

*Operation:* Receive a message that another member of the XCF group sent. The token and length for the message are accessed from the parameter list at entry to the message user-routine (using the IXCYMEPL mapping macro). Register 4 is the length of the message; the STORAGE macro obtains an area equal to that length. Register 3 contains the address of this area. TOKENMSG contains the token of the message to be received. XCF is to store the return code and reason code into the variables RETURN and REASON. Although not shown here, you must obtain storage for the dynamic area (@DATD). See “Using the Cross-System Coupling Facility” in *z/OS MVS Programming: Sysplex Services Guide*, for a complete example showing the use of IXCMSGIX in a message user routine.

**Note:** Since more than one message user routine can be running at a time, **message user routines should be reentrant**. IBM recommends that you use the list and execute forms of IXCMSGIX.

```

L      R4,MEPLMLN          LENGTH OF INCOMING MESSAGE
STORAGE OBTAIN,LENGTH=(R4),SP=0  GET STORAGE FOR MESSAGE
LR     R3,R1               SAVE ADDRESS OF STORAGE
MVC    TOKENMSG(4),MEPLMTOK  MESSAGE TOKEN TO PASS TO XCF

IXCMSGIX MSGTOKEN=TOKENMSG,MSGBUF=(R3),RETCODE=RETURN,      X
        RSNCODE=REASON,MF=(E,MSGILSTD)  HAVE XCF PUT      X
                                           MESSAGE IN STORAGE X
                                           JUST OBTAINED

* DSECT used to map the dynamic area storage
@DATD      DSECT
           DS 0F
MSGILST2    IXCMSGIX MF=(L,MSGILSTD)  LIST FORM OF IXCMSGIX MACRO
RETURN      DS 1F                    RETURN CODE
REASON      DS 1F                    REASON CODE
TOKENMSG    DS CL4                  MESSAGE TOKEN TO PASS TO XCF

```

You can obtain the message token from the MEPLMTOK field in the parameter list passed to the message user routine. The message token identifies the message to be received. Do not confuse the message token with the member token. The member token identifies the member that sent the message. You can obtain the member token from the MEPLSRCE field in the parameter list.



---

## Chapter 15. IXCMMSGO — Send a Message to Another Member in the XCF Group

**Note:** IBM suggests using the IXCMMSGOX macro interface service for XCF message-out service requests. IXCMMSGOX contains functions that the IXCMMSGO macro interface does not have. For details about the IXCMMSGOX macro, see [Chapter 16, “IXCMMSGOX — Send a Message to Another Member in the XCF Group,” on page 183](#).



## Chapter 16. IXCMMSGOX – Send a Message to Another Member in the XCF Group

### Description

The IXCMMSGOX macro is the interface to the XCF (cross-system coupling facility) message-out service. This interface is the successor to the IXCMMSGO macro interface, contains all the functionality of the IXCMMSGO message-out macro interface and is the suggested XCF message-out service interface to use.

The IXCMMSGOX macro allows a member of an XCF group to send a message to one or more members in its XCF group. IXCMMSGOX, and its companion services, IXCMMSGIX (message-in service) and IXCMMSGC (message control service), comprise the XCF signalling services. The target XCF member can receive the message by issuing the IXCMMSGIX macro from within a message user routine it defines when it joins the XCF group. The target member can also save or discard the message by issuing the IXCMMSGC macro from within the message user routine. Between pairs of members, you can specify that XCF is to deliver messages to a target member in the same order in which they were sent.

Messages can vary in length up to 134 217 278 (128M) bytes. An IXCMMSGOX request to send a "large message" must specify MSGACCESS=ASYNCR or MSGACCESS=SYNCSUSPEND. A "large message" is defined to be one that is greater than 62 464 (61K) bytes. XCF handles large messages somewhat differently from messages of less than 61K bytes, and also imposes the following restrictions when sending large messages:

- Both sending and target **systems** must be running OS/390 Release 8 or higher. (This does not necessarily imply that all **members** residing on those systems support large message delivery.)
- Both sending and target **members** must have specified when they joined their XCF group that they supported large message delivery. (Each must have specified GT61KMSG=YES on the IXCJOIN invocation.)

The following IXCMMSGOX error notifications occur if either of the above conditions is not met:

- If a **target system** is not at the appropriate release level, requests to send a large message to a member on that system are rejected with return code X'8', reason code X'340'. If a **source system** is not at the appropriate release level, requests to send a large message from a member on that system are rejected with return code X'8', reason code X'40'.
- If a sending **member** has not specified support for large message delivery when joining the XCF group, a request to send a large message is rejected with return code X'8', reason code X'C'. If a target **member** has not specified support for large message delivery when joining the XCF group, a request to send a large message is rejected with return code X'8', reason code X'340'.

You pass the message to the IXCMMSGOX service using either a single buffer or multiple buffers. If you are using a single buffer, you need only:

- Identify the message buffer using the MSGBUF parameter.
- Identify the message length using the MSGLEN parameter.

If you are sending a message that resides in multiple buffers, you have several options for specifying the location and size of the message buffers. For each message buffer, you must describe a **message data element**. Message data elements can either contain the message buffer or provide a pointer to it. Message data elements can optionally contain the length of the buffer and an ALET to qualify the buffer address (if the buffer is not in message data element).

You can pass the message data elements either organized as a queue or a table. Message buffer lengths and message buffer ALETs can also be passed separately if you do not wish to include them in the message data elements. A message buffer can be in your primary address space, in an address space accessible through your dispatchable unit access list (DU-AL), or in a common area data space. z/OS MVS

*Programming: Sysplex Services Guide* provides a complete description of the different options for passing messages in multiple buffers.

The buffer storage in which the message resides is accessed by XCF when IXCMGSOX is invoked. XCF can access this storage in one of two ways, as specified by the MSGACCESS keyword.

- With MSGACCESS=SYNC, XCF accesses the storage synchronously with the message-out request. Once IXCMGSOX returns to the sender, the sender can dispose of the storage that contained either the message or the queue or table that defined the parts of the message.

Note that a request to send a message longer than 61K bytes will be rejected with return code X'8', reason code X'C' when MSGACCESS=SYNC is specified.

With MSGACCESS=SYNCSUSPEND, a request to send a message in the decimal range of 0 to 134 217 728 (128M) bytes long can be made synchronously. As needed, XCF can suspend the current unit of work for a specified amount of time in order to complete accessing storage areas associated with message text. As with MSGACCESS=SYNC, once IXCMGSOX returns to the sender, the sender can dispose of the storage that contained either the message or the queue or table that defined the parts of the message.

MSGACCESS=SYNCSUSPEND is useful for senders that want to send messages that are greater than 61K in length, can tolerate having their unit of work possibly suspended and that would like to release storage resources that contained message text, and queue or table elements that defined parts of the message prior to the completion of the message.

- With MSGACCESS=ASYN, XCF can access the storage even after IXCMGSOX returns to the sender. If IXCMGSOX returns with return code X'4', reason code X'410', the sender must preserve the storage containing the message text as well as the queue or table that defined the parts of the message until the message is completed. To determine when a message is complete, either
  - Wait for XCF to notify the sender via the message notify user routine.
  - Invoke the IXCMGSC service. (IXCYMQAA contains an indication as to whether the message has completed.)

With MSGACCESS=ASYN, you must specify a non-zero TIMEOUT value. See *z/OS MVS Programming: Sysplex Services Guide* for additional information about specifying a timeout value.

For any other return and reason codes, the sender can dispose of the storage as with MSGACCESS=SYNC or MSGACCESS=SYNCSUSPEND.

Messages you send using IXCMGSOX are delivered asynchronously. A return code of zero from IXCMGSOX indicates that XCF has accepted the message; it does not indicate that the message has been delivered. If it is necessary for the sender to know the message has been received, then it is up to the sender and receiver to maintain a protocol to indicate that the message has been received. Alternatively, the sender can request that the target member is to respond to the message and that XCF is to manage the response or the collection of responses, if the message is sent to more than one target member.

If you do not specify the #MSGPARTS parameter and, while looking for your message data, IXCMGSOX finds more than 65536 consecutive buffers of length zero, IXCMGSOX assumes an error has occurred. The message is not sent, and you receive a return code and reason code indicating the error.

Be sure to read the IXCMGSOX guidance information in *z/OS MVS Programming: Sysplex Services Guide*. The information presented here assumes you have done so. IXCMGSOX guidance information includes:

- Illustrations showing the message data element formats specified by the various IXCMGSOX parameters.
- Design considerations relating to the use of signalling services.
- Information on designing a protocol for sending and receiving multipart messages.

## Environment

---

The following are the environment requirements for the caller.



Environment	Environment requirement
Minimum authorization:	Supervisor state or PKM allowing key 0-7
Dispatchable unit mode:	Task or SRB
Cross memory mode:	Any HASN, any SASN. Unless MSGOUTASID=ANY was specified at the time the group was joined, the primary address space must equal the primary address space of the caller of the IXCJOIN macro when it was issued to join the XCF group; or you must be running in the master scheduler address space.  If MSGOUTASID=ANY was specified at the time the group was joined, IXCMGGOX can be issued from any address space.
AMODE:	31-bit or 64-bit. To pass keyword values that name and reference 64-bit virtual storage, specify SYSSTATE AMODE64=YES before invoking the IXCMGGOX macro service.  The following IXCMGGOX macro parameters may reside in either 31-bit or 64-bit addressable virtual storage: <ul style="list-style-type: none"> <li>• MSGBUF</li> <li>• MSGCNTL</li> <li>• ELEMENT</li> <li>• TARGETS</li> <li>• RETMSGOTOKEN</li> <li>• PARTLENTBL</li> <li>• PARTALETBL</li> <li>• The TABLE or QUEUE of message elements that describe the message parts of a MULTIPART message. In addition, the buffer addresses and next message data element address for a message data element queue contained within the message data elements may reference 31-bit or 64-bit storage</li> <li>• An area to be used for the parameter list of the IXCMGGOX macro</li> </ul>
ASC mode:	Primary or access register (AR).
Interrupt status:	Enabled for I/O and external interrupts
Locks:	No locks held
Control parameters:	Must be in the primary address space of the caller and might be in 31-bit or 64-bit addressable virtual storage.

## Programming Requirements

If the program is in AR mode, issue the SYSSTATE ASCENV=AR macro before IXCMGGOX. SYSSTATE ASCENV=AR tells the system to generate code appropriate for AR mode.

If the parameter list specified for the macro service call resides in 64-bit virtual storage above the 2-gigabyte bar, the caller must be executing in AMODE 64 when invoking the IXCMGGOX macro service.

For MULTIPART=YES, the invoker of IXCMGGOX is responsible for maintaining the integrity of the element structure. If the elements or their structure change while being processed by the message-out service, the message data actually sent may be corrupted.

- If MSGACCESS=SYNC or MSGACCESS=SYNCSUSPEND was specified, the caller is responsible for maintaining the integrity of the element structure until the message-out service returns.

- If MSGACCESS=ASYN was specified, the caller is responsible for maintaining the integrity of the element structure until the message is completed.

All message data elements (but not buffers whose addresses are located in the message data elements) must reside in the same address space or data space so they can be accessed using the same ALET.

## Restrictions

---

- The sending and receiving members must be active members of the same XCF group.
- When invoked from an address space resource manager such as the MASTER address space, some IXCMSGOX functions might not be available. Any request that requires the use of one of the member data spaces that XCF manages on behalf of the member is subject to rejection. XCF attempts to perform the requested function without use of these data spaces. If a member data space is required, the request is rejected. Do not use the following keywords if your application cannot tolerate such rejections: SENDTO(GROUP), GETRESPONSE(YES), NOTIFY(YES), or TIMEOUT.  
  
When MSGACCESS=ASYN or MSGACCESS=SYNCSUSPEND is specified, if the sending member's associated task or address space or both is undergoing termination, XCF will not be able to asynchronously access the sender's data areas. (See the MEMASSOC keyword on the IXCJOIN service.) Thus, an IXCMSGOX request issued from the sender's task or address space resource manager termination routine that completes with return code X'4', reason code X'410' will not be able to successfully complete the send of the message.
- When MSGACCESS=ASYN or MSGACCESS=SYNCSUSPEND is specified, the storage indicated by MSGBUF is subject to the following restrictions:
  - If the storage is in the caller's primary address space and this space is not the same space that was primary when the sending member joined its XCF group (that is, the joiner's address space), the caller's primary address space must be non-swappable.
  - If the storage is in a data space accessible via a public entry on the caller's DU-AL, the data space must either be owned by the joiner's address space, or be owned by a non-swappable address space, or be a common area data space.
  - If the storage is in an address space accessible via a public entry on the caller's DU-AL, that address space must either be the joiner's address space or be a non-swappable address space.
- Also when MSGACCESS=ASYN or MSGACCESS=SYNCSUSPEND is specified, the Dispatchable Unit Access List (DU-AL) under which the caller is running must adhere to the following restrictions. The DU-AL must never have had access to a subspace, must never have had access to more than 255 spaces of any kind at one time, and must not be full. If the message-out service is unable to process the request because the DU-AL is unsuitable, the IXCMSGOX request is rejected with return code X'C' and an appropriate reason code.
- When MSGACCESS=SYNCSUSPEND is specified, the following environmental restrictions apply:
  - The XCF message-out request cannot be issued from a SUSPEND exit routine or from an SRB routine that the system abended with a 47B system completion code.
  - The XCF message-out request using MSGACCESS=SYNCSUSPEND cannot be issued from an address space resource manager. An IXCMSGOX request issued from an address space resource manager termination routine completes with return code X'8", reason code X'354'.

## Input Register Information

---

Before issuing the IXCMSGOX macro, the caller does not have to place any information into any register unless using it in register notation for a particular parameter, or using it as a base register.

## Output Register Information

---

When control returns to the caller, the general purpose registers (GPRs) contain:

### Register Contents

**0**

Reason code, if GPR 15 contains a non-zero return code that has applicable reason codes.

**1**

Used as a work register by the system.

**2-13**

Unchanged.

**14**

Used as a work register by the system.

**15**

Return code.

When control returns to the caller, the access registers (ARs) contain:

### Register Contents

**0-1**

Used as work registers by the system

**2-13**

Unchanged

**14-15**

Used as work registers by the system

Some callers depend on register contents remaining the same before and after issuing a service. If the system changes the contents of registers on which the caller depends, the caller must save them before issuing the service, and restore them after the system returns control.

## Performance Implications

---

To prevent backlogs that can degrade system performance, a target must be able to process messages as fast, or faster than the sender creates them.

For the best performance, specify the message length using the MSGLEN parameter. If you omit the MSGLEN parameter, IXCMGGOX must perform processing to determine the message length.

The IXCMGGOX services that incur the least overhead when sending a message are:

- SENDTO=MEMBER
- NOTIFY=NO
- MSGBUF=x
- UNORDERED
- GETRESPONSE=NO

Other IXCMGGOX services require additional processing.

## Understanding IXCMGGOX Version Support

---

The IXCMGGOX macro supports versions in the range of 4 - 6.

- Keywords not specifically noted here are supported by version 4 and subsequent versions of the IXCMGGOX macro.
- The following keywords and functions are supported by version 5 and subsequent versions of the IXCMGGOX macro.
  - ATTRIBUTES
  - DELIVERMSG

- GETRESPONSE
- MEMBERS
- MSGACCESS
- NEXTTARGETOFF
- NOTIFY
- NOTIFYBY
- NOTIFYEXIT
- NOTIFYIF
- RESPONSEID
- RETMSGOTOKEN
- SENDTO
- STREAMID
- TARGETS
- TIMEOUT
- USERDATA
- #TARGETS
- The following keywords and functions are supported by version 6 and subsequent versions of the IXCMGGOX macro.
  - ELEMADDRMODE
  - ELEMENT
  - ELEMFORM
  - ENDOFQUEUE
  - MULTIPART
  - NEXTOFF
  - NEXTPTROFF
  - PARTALET
  - PARTALETTOFF
  - PARTALETTLBL
  - PARTLEN
  - PARTLENOFF
  - PARTLENTBL
  - PARTOFF
  - PARTPTROFF
  - #MSGPARTS
  - SENDTIME

Specify the version of the parameter list that you want generated with the PLISTVER keyword. See Chapter 2, [“Specifying a Macro Version Number,”](#) on page 5 for considerations when specifying the version of the parameter list with PLISTVER.

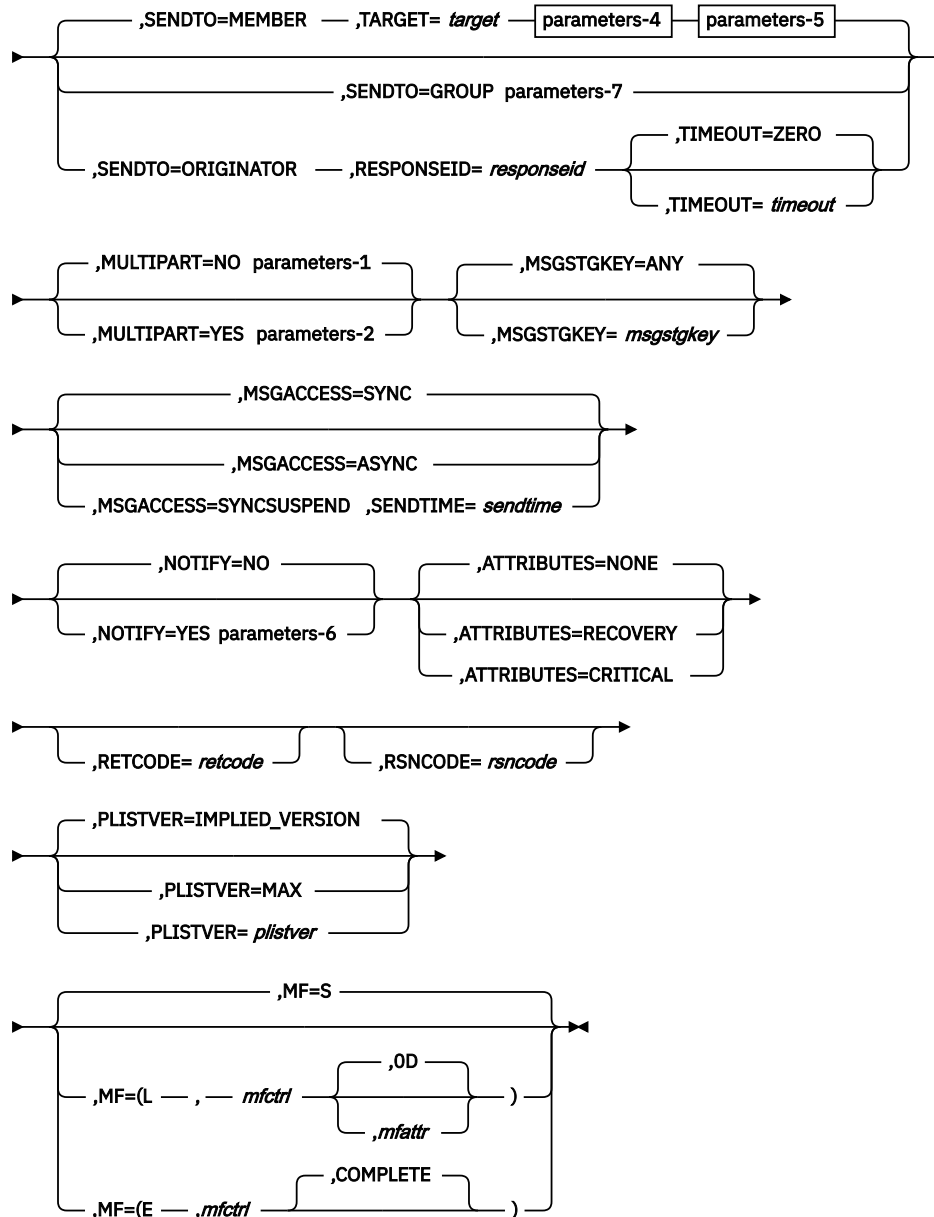
## Syntax Diagram

---

The syntax of the IXCMGGOX macro is as follows:

## main diagram

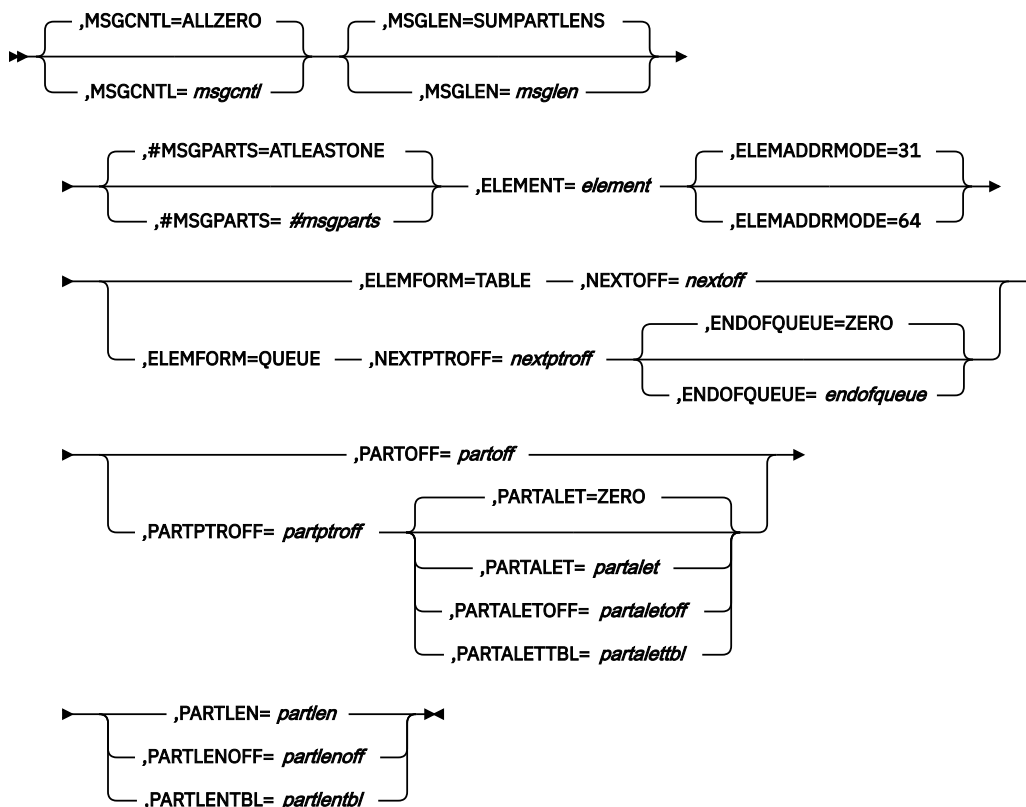
► IXCMSGOX — — MEMTOKEN= *memtoken* →



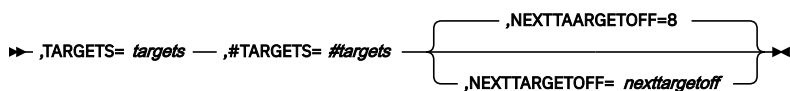
## parameters-1



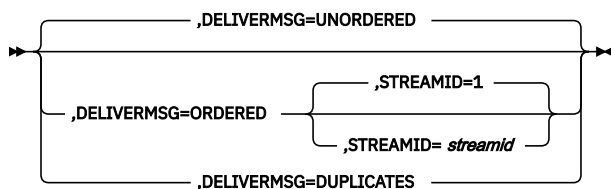
## parameters-2



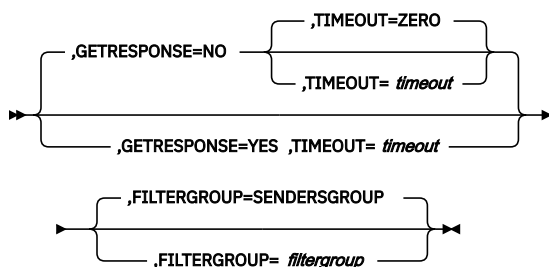
## parameters-3

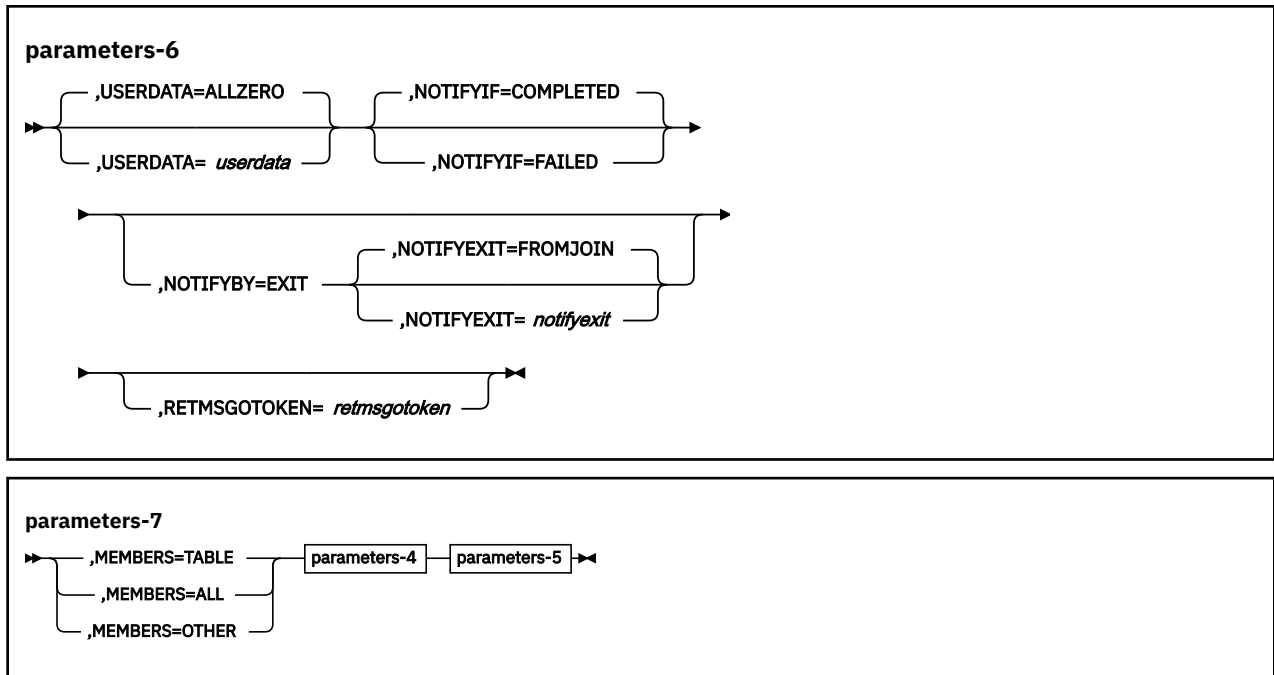


## parameters-4



## parameters-5





## Parameter Descriptions

The parameter descriptions are listed in alphabetical order. Default values are underlined:

**,ATTRIBUTES=NONE**

**,ATTRIBUTES=RECOVERY**

**,ATTRIBUTES=CRITICAL**

Use this parameter to specify the special attributes to be associated with the message being sent.

**ATTRIBUTES=NONE**

No special attributes will be associated with the message. The message will be delivered in accordance with normal message delivery practices.

**ATTRIBUTES=RECOVERY**

The message is a *recovery signal* being sent as part of a sysplex recovery protocol. The RECOVERY attribute indicates that it is important for the signal to be delivered in a timely manner to ensure that the sysplex recovery protocol can finish as soon as possible, and thereby enable the sysplex to resume normal operation.

**ATTRIBUTES=CRITICAL**

The message is to be treated as critical to the functionality of the XCF group by the target member.

**,DELIVERMSG=UNORDERED**

**,DELIVERMSG=ORDERED**

**,DELIVERMSG=DUPLICATES**

Use this input parameter to indicate whether the system is to deliver the message in sequential order, or whether the target member can tolerate receiving the message for multiple times.

**DELIVERMSG=UNORDERED**

The system is to deliver the message as soon as possible. No ordering is imposed on this message with respect to any other message sent by the sending member to the target member.

Given any other message sent by the sending member to the target member, no assumption can be made about the order in which the system will deliver that message and this message.

**DELIVERMSG=ORDERED**

The system is to impose ordering on this message with respect to other ordered messages sent by the sending member to the target member. The ordering is determined by the order in which the message-out service accepts the messages for delivery. The system presents the ordered messages to the message user routine of the target member in the same order that the messages

were accepted for delivery. Delivery of these messages might be delayed to ensure that proper ordering occurs.

Note that the ordering is imposed on messages sent between a particular **pair** of members. The ordered delivery of messages sent between one particular pair of members is independent of message delivery between any other pair of members.

Use the STREAMID keyword to send messages in independently ordered streams between a particular pair of members. The streams of any one pair of members are independent of the streams of any other pair of members. For a given pair of members, messages within a particular stream are delivered independently of any other stream.

For ordered message delivery to occur, the systems on which the sending member and the target member reside must both be running a release that supports ordered message delivery. If the message is sent to a target member that resides on a release that does not support ordered message delivery, unordered delivery occurs. Use the XCF Query Service (IXCQUERY) to determine whether the ordered delivery protocol is supported for a particular target member.

### **DELIVERMSG=DUPLICATES**

The target member can tolerate receiving the message for multiple times. In general, one should expect exactly one instance of the message to be delivered to each target member. However, if the signalling path over which the message has been sent is restarted or stopped before the signal is known to have been transferred to the target system, this option allows XCF to send the signal again to the target member through an alternate signalling path without first discovering whether the target system had already received the signal.

If the original signal has been transferred to the target system, both the original message and the resent copy are delivered to the target member.

If the signalling path used to send the second instance of the signal is similarly restarted or stopped, XCF will send yet another instance of the signal through an alternate signalling path. Thus specifying DELIVERMSG=DUPLICATES can potentially result in an arbitrary number of copies of the message being delivered to the targeted member.

It is up to the sender to determine whether or not the target member can tolerate duplicate messages. If the message is being broadcast to multiple targets, then every target member must tolerate duplicate copies of the message.

### **,ELEMADDRMODE=31**

### **,ELEMADDRMODE=64**

Use this parameter to specify the precision format of the addresses that is located within the message data element at the offsets specified by the PARTPTROFF and NEXTPTROFF keywords. The default is ELEMADDRMODE=31.

Use ELEMADDRMODE=31 to indicate that all of the addresses in message data elements are of 31-bit addressing precision (single word - 4 bytes) and reference only virtual storage below the 2-gigabyte virtual storage bar.

Use ELEMADDRMODE=64 to indicate that all of the addresses in message data elements are of 64-bit addressing precision (double word - 8 bytes) and can reference virtual storage below or above the 2-gigabyte virtual storage bar.

### **,ELEMENT=xelement**

Use this input parameter to specify the first element of the table or queue of message data elements. Message data elements contain either buffers or pointers to buffers that contain parts of the message to be sent.

Specifying the PARTOFF parameter indicates that each element contains a buffer. Specifying the PARTPTROFF parameter indicates that each element contains a pointer to a buffer.

Elements must all reside in the same address space or data space — the caller's primary address space, an address space or data space that is accessible through a public entry on the caller's dispatchable unit access list (DU-AL), or a common area data space.



**To Code:** Use a register from 2 to 12 to specify the RS-type name or address of a required character input area that is the first element containing the data that describes the parts of the message.

**,ELEMFORM=TABLE**

**,ELEMFORM=QUEUE**

Use ELEMFORM=TABLE to indicate that the message data elements are organized as a table.

Use ELEMFORM=QUEUE to indicate that the message data elements are organized as a queue.

**,ENDOFQUEUE=ZERO**

**,ENDOFQUEUE=endofqueue**

Use this input parameter to specify the address that marks the end of the queue. When the pointer to the next element contains this address, queue processing ends.

**Note:** The queue must have at least one element.

If you omit ENDOFQUEUE, or specify ENDOFQUEUE=ZERO, the default value for the end-of-queue address is zero.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword containing the end-of-queue address.

**,FILTERGROUP=SENDERGROUP**

**,FILTERGROUP=filtergroup**

Use this input parameter to further qualify the set of members to which the message is sent for a SENDTO=GROUP request.

**,FILTERGROUP=SENDERGROUP**

XCF will send the message to all active target group members according to the MEMBERS specification.

**,FILTERGROUP=filtergroup**

A FILTERGROUP specification is used to further qualify the set of members to which the message is sent. XCF will first construct the set of candidate target members according to the MEMBERS specification. When sending the message, XCF will apply the FILTERGROUP filter to each candidate target member as follows. If the FILTERGROUP group has an active member on the system where the candidate target member resides, the message is sent to the target member. If the FILTERGROUP group does not have an active member on that system, the target member is "skipped". The message will not be sent to that target member.

In effect, FILTERGROUP prevents the message from being sent to certain systems in the sysplex. Such filtering could be useful in cases where two different XCF groups need to cooperate to perform their function. It might be the case that a given message only needs to be processed by systems that have active members in both groups.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-byte character field containing an XCF group name to use as the filter group value. The 8 character group name must be left justified and padded on the right with blanks if necessary. The valid characters are A-Z, 0-9, \$, @, and #.

**,GETRESPONSE=NO**

**,GETRESPONSE=YES**

Use this input parameter to specify whether XCF is to manage the collection of responses to this message.

**GETRESPONSE=NO**

XCF is not to manage the collection of responses for this message. It is the responsibility of the user to handle any management of responses.

**GETRESPONSE=YES**

XCF is to manage the collection of responses for this message. When all responses are collected, XCF presents them to the message sender. Note that XCF recognizes situations for which it is not reasonable to expect a response, such as when a target member's system fails.

Response collection is a cooperative effort between XCF and the target members. When XCF presents the message to a target member's message user routine, the parameter list contains the following:

- A flag (MEPLNEEDSRESPONSE in IXCYMEPL) is set to indicate that XCF is managing response collection.
- A response identifier (MEPLRESPONSEID in IXCYMEPL) is provided to allow XCF to identify the message to which the response is to be associated.

After composing the response, the target member replies by invoking the message-out service to send the response to the ORIGINATOR of the message identified by the RESPONSEID. XCF recognizes the message as a response by the particular form of the message-out invocation used to make the reply. When collection of the response(s) has completed, XCF presents the collected response message(s) to the member that originated the GETRESPONSE message-out request through the member's message notify user routine.

For XCF to collect a response, the system on which a target member resides must be running a release that supports XCF-managed response collection. Also, the target member must be capable of participating in this protocol by invoking the message-out service to send its reply in a way that allows XCF to recognize it as a response. XCF sends the message to a target member even if it appears that the member is unable to participate in the XCF response collection. However, XCF does not expect a response from such a member. Use the XCF Query Service (IXCQUERY) to determine whether the XCF-managed response collection protocol is supported for a particular target member.

#### **,MEMBERS=TABLE**

#### **,MEMBERS=ALL**

#### **,MEMBERS=OTHER**

Use this input parameter to indicate how the collection of target members is to be determined.

#### **MEMBERS=TABLE**

The sender is providing a table that contains a member token for each intended target member.

The table is an array of entries. Each entry has the same fixed size, and can contain data other than a target member token. The storage location identified by the TARGETS keyword contains the first 64-bit target member token. Subsequent target member tokens are iteratively located by adding the value NEXTTARGETOFF to the address of each member token in turn. The keyword #TARGETS indicates the number of entries in the table.

The table can contain "holes", that is, entries that contain a target member token of X'0'. XCF will skip these entries, but will preserve the entries so that the TARGETS table has a one-to-one correspondence with any other table that XCF constructs for this request. For example, the "i'th" entry of the TARGETS table would correspond to the "i'th" MQATARGRESPENTRY (or MQATARGONLYENTRY) returned by the XCF Message Control Query Message service (IXCMGGO), or would correspond to the "i'th" MNPLTARGRESPENTRY (or MNPLTARGONLYENTRY) presented to a message notify user routine.

The IXCMGGOX invoker is responsible for maintaining the integrity of the TARGETS table until the Message-out service returns. If a table changes while being processed by the Message-out service, the message might be sent to a different set of targets than expected. Also, the content of the entries in tables constructed by XCF might no longer correspond to the content of the entries in the TARGET table.

#### **MEMBERS=ALL**

Send a copy of the message to every active member in the sender's XCF group (includes the sender).

XCF determines the collection of members that are currently active in the sender's XCF group throughout the sysplex. This collection is equivalent to the list of members that would be returned by invoking the XCF Query service IXCQUERY with REQTYPE=IMMEDIATE.

**MEMBERS=OTHER**

Send a copy of the message to every active member in the sender's XCF group except for the sender.

XCF determines the collection of members that are currently active in the sender's XCF group throughout the sysplex. This collection is equivalent to the list of members that would be returned by invoking the XCF Query service IXCQUERY with REQTYPE=IMMEDIATE, with the sender excluded.

**,MF=S**  
**,MF=(L,mfctrl)**  
**,MF=(L,mfctrl,mfattr)**  
**,MF=(L,mfctrl,0D)**  
**,MF=(M,mfctrl)**  
**,MF=(M,mfctrl,COMPLETE)**  
**,MF=(M,mfctrl,NOCHECK)**  
**,MF=(E,mfctrl)**  
**,MF=(E,mfctrl,COMPLETE)**  
**,MF=(E,mfctrl,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

**,mfctrl**

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

**,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

**,COMPLETE****,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

**COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if SMILE=*var* were an optional parameter and the default is SMILE=NO\_SMILE then it would not be documented. However, if the default was SMILE=:-), then it would be documented because a value would be the default.

**NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

**MEMTOKEN=memtoken**

Use this input parameter to specify the member token you received when you issued the IXCJOIN macro to join your XCF group. This token identifies the member sending the message.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 64-bit field that contains the member token of the sending member.

**,MSGACCESS=SYNC****,MSGACCESS=SYNCSUSPEND****,MSGACCESS=ASYN**

Use this input parameter to indicate how XCF can access the storage containing the text of the message.

**MSGACCESS=SYNC**

XCF accesses the storage containing the text of the message synchronously with the message-out request. After the message-out service returns, the sender can dispose of the storage containing the message. For multipart messages, the sender can dispose of the storage containing the elements or tables that define the parts of the message as well.

**MSGACCESS=ASYN**

XCF is allowed to access the storage containing the text of the message after the message-out request returns to the sender. Note that specifying MSGACCESS=ASYN does not necessarily imply that XCF will access the storage asynchronously. When possible, XCF will attempt to use synchronous access.

If IXCMSGOX returns with return code X'4' and reason code X'410', the sender must preserve the storage containing the message text until the message is completed. For multipart messages, the elements and tables or queues must be preserved as well. For any other return and reason code, the sender can dispose of the storage as if MSGACCESS=SYNC were specified.

If IXCMSGOX returns with return code X'0', implying that XCF has accepted the message for delivery, **and** you have specified that the message notify user routine is to receive control when the message is complete, you do not need to preserve the storage containing the message text or the elements and tables or queues. However, the application may have to coordinate responsibility for freeing the message storage area between the sender and the notify exit. The notify exit can determine whether the message storage area had to be preserved by checking the MNPLMSGOASYNMSGACCESS flag.

If the sending member becomes inactive before an ASYN message is completed, the message-out service might terminate processing for the message. For example, the message might not be delivered to the target member(s) once the sender becomes inactive. In contrast, with MSGACCESS=SYNC, delivery of messages that were accepted by the message-out service continues even after the sender becomes inactive.

For MSGACCESS= ASYN or MSGACCESS=SYNCSUSPEND, the length of the message can be in the decimal range of 0 to 134 217 728 (128M) bytes. If the sending member becomes inactive while the unit of work is suspended, the unit of work is to be released and control returned to the caller. The message-out service might end processing for the message. Specifically, the message might not be delivered to one or more of the target members when the sender becomes inactive.

MSGACCESS=ASYN or MSGACCESS=SYNCSUSPEND must be specified to send message longer than 62 464 (61K) bytes.

See [“Restrictions” on page 186](#) for additional information about using MSGACCESS=ASYN.

**MSGACCESS=SYNCSUSPEND**

The storage containing the text of the message must be accessed synchronously with the message-out request. As needed, XCF can suspend the current unit of work for a specified amount of time in order to complete accessing storage areas that are associated with message text.

After the message-out service returns, the sender is free to dispose of the storage containing the message. For multipart messages, the sender can dispose of the storage containing the elements and/or tables that define the parts of the message as well. When XCF returns, the message might not be complete.

MSGACCESS=SYNCSUSPEND is useful for senders that want to release storage resources that contained message text and queue or table elements that defined parts of the message before the message completes.

For MSGACCESS= SYNCSUSPEND, the length of the message can be in the decimal range of 0 to 134 217 728 (128M) bytes.

If the sending member becomes inactive while the unit of work is suspended, the unit of work is to be released and control returned to the caller. The message-out service might end processing for the message. Specifically, the message might not be delivered to one or more of the target members when the sender becomes inactive.

MSGACCESS=ASYN or MSGACCESS=SYNCSUSPEND must be specified to send message longer than 62 464 (61K) bytes.

See “Restrictions” on page 186 for additional information about using MSGACCESS=ASYN.

#### **,MSGBUF=msgbuf**

Use this input parameter with MULTIPART=NO to specify the buffer containing the message to be sent. The storage key of the buffer specified by MSGBUF must match the storage key specified with the MSGSTGKEY parameter (if you code it).

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the storage area containing the message.

#### **,MSGCNTL=ALLZERO**

#### **,MSGCNTL=msgcntl**

Use this input parameter to specify a storage area containing user-specified control information to help the receiving member process the message. Possibilities include:

- The type of message being sent
- The format of the message
- A sequence number to help detect missing signals when using ordered delivery.

The message control information is passed to the message user routine of the receiving member in the message exit parameter list (MEPL), which is mapped by the IXCYMEPL macro.

If you omit the MSGCNTL parameter, or if you specify MSGCNTL=ALLZERO, the message control information presented to the receiving member consists of zeros.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the 32-byte storage area containing the message control information.

#### **,MSGLEN=SUMPARTLENS**

#### **,MSGLEN=msglen**

Use this input parameter to specify the length of the message in bytes.

- For MSGACCESS=SYNC, the value must be in the decimal range of 0 to 62464 (61K) bytes.
- For MSGACCESS=ASYN or MSGACCESS=SYNCSUSPEND, the value can be in the decimal range of 0 to 134 217 728 (128M) bytes. If MSGLEN exceeds 61K, both the sending member and the target member must have specified GT61KMSG=YES when the IXCJOIN macro was invoked to join the group. Use the IXCQUERY service to determine whether a particular target member supports the large message delivery protocol.

The amount of buffer storage you provide must be equal to or greater than the value of MSGLEN.

If you specify MULTIPART=YES, then the sum of the lengths of the parts is the default value.

**Note:** SUMPARTLENS may be specified only when MULTIPART=YES is specified.

If you omit MSGLEN, IXCMGGOX determines the message length for you, but performance is reduced.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword containing the length of the message.

**,MSGSTGKEY=ANY****,MSGSTGKEY=msgstgkey**

Use this input parameter to specify the storage key to be used by the system when fetching the message from each buffer. XCF can fetch the message data successfully from a buffer only if the storage key of the buffer matches the storage key specified by MSGSTGKEY or the storage is not fetch protected.

If you omit the MSGSTGKEY parameter, or if you specify MSGSTGKEY=ANY, XCF does not verify that the buffers have any particular storage key.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-bit field formatted as **kkkkxxxx**, where the high-order four bits contain the storage key. The low-order halfword is ignored.

**,MULTIPART=NO****,MULTIPART=YES**

Use MULTIPART=NO to indicate that the message is in a single buffer.

Use MULTIPART=YES to indicate that the message is in one or more buffers.

**,NEXTOFF=nextoff**

Use this input parameter to specify the number of bytes to be added to the address of the current element to obtain the address of the next element. This value equals the size in bytes of an individual message data element. It is used when the message data elements are in table form.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword containing the number of bytes.

**,NEXTPTROFF=nextptroff**

Use this input parameter to specify the number of bytes to be added to the address of the current element, to locate within the current element either a 31- or 64-bit precision pointer to the next element. This value is used when the message data elements are in queue form.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword containing the number of bytes.

**,NEXTTARGETOFF=8****,NEXTTARGETOFF=nexttargetoff**

Use this input parameter to specify, in bytes, the relative location of the next member token in the TARGETS table. To locate the next member token, add the NEXTTARGETOFF value to the location of the current member token. Usually this value is the length of an individual entry in the TARGETS table.

The default value, if NEXTTARGETOFF is not specified, is 8 bytes, meaning that each table entry consists of a member token only.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the fullword field containing the number of bytes in an individual table entry.

**,NOTIFY=NO****,NOTIFY=YES**

Use this input parameter to indicate whether the member is to be notified of message completion.

A message is considered complete in the following circumstances:

- A message-out request times-out or is forced to completion by IXCMSGC with TYPE=COMPLETION.
- XCF has initiated the send of a message without response
- XCF has initiated the send of a broadcast message to each valid target member
- The response to a message with response has arrived.
- The expected responses to a broadcast message with response have arrived.

**NOTIFY=NO**

Specify this option when you do not require notification of message completion. This option provides compatibility with previous versions of IXCMSGOX.

**NOTIFY=YES**

Specify this option when you require notification of message completion. XCF will maintain status information about the message until the message is completed. Upon completion of the message, the state of the message and the NOTIFYIF keyword determine how XCF is to proceed. If a member is to be notified of message completion, XCF preserves status information and uses the NOTIFYEXIT to inform the member.

**,NOTIFYBY=EXIT**

Use this input parameter to indicate that the member is to be notified of message completion through a message notify user routine scheduled by XCF.

**,NOTIFYEXIT=FROMJOIN****,NOTIFYEXIT=notifyexit**

Use this input parameter to identify the message notify user routine to receive control when the message is complete. If omitted, the system uses the message notify user routine specified on IXCJOIN when the member joined the XCF group. Note that the system rejects the IXCMGGOX request if the NOTIFYEXIT defaults to FROMJOIN and no message notify user routine was specified on IXCJOIN.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 31-bit message notify user routine.

**,NOTIFYIF=COMPLETED****,NOTIFYIF=FAILED**

Use this input parameter to identify the conditions under which the system is to provide notification of message completion. See the NOTIFY keyword for the definition of message completion.

**NOTIFYIF=COMPLETED**

The system is to provide notification when the message is completed. Notification occurs whether the message succeeded or failed.

**NOTIFYIF=FAILED**

The system is to provide notification only if the message failed.

- A message without response is considered to have failed if the message was not sent to one of the possible target members.
- A message with response is considered to have failed if XCF does not receive a response from every possible target member. (For example, if a message with response is targeted to a member that is no longer active — and therefore, no response is possible — the message is considered to have failed.)

Note that skipped targets are ignored when determining whether the message failed. A skipped target is one whose member token in the TARGETS table is X'0'.

**,PARTALET=ZERO****,PARTALET=partalet**

Use this input parameter to specify a single ALET to be used to qualify every buffer address. The ALET must be zero, a public entry on your dispatchable unit access list (DU-AL), or an entry for a common area data space.

If you omit PARTALET, PARTALET OFF, and PARTALET TBL, the default is PARTALET with an ALET of zero.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword containing the ALET.

**,PARTALET OFF=partalet off**

Use this input parameter to specify the number of bytes to be added to the address of the current element, to locate within the element the fullword field that contains the ALET to be used to qualify the buffer address associated with that element. The ALET must be zero, a public entry on your dispatchable unit access list (DU-AL), or an entry for a common area data space.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword containing the number of bytes.

**,PARTALETTLBL=partalettbl**

Use this input parameter to specify a table in which each entry is a fullword containing the ALET to qualify a buffer address in the table or queue of message data elements. For instance, the 3rd entry in PARTALETTLBL must contain the ALET to qualify the buffer address in the 3rd message data element. *partalettbl* must begin on a fullword boundary. Each ALET must be zero, a public entry on your dispatchable unit access list (DU-AL), or an entry for a common area data space.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of *partalettbl*.

**,PARTLEN=partlen**

Use this input parameter to specify the length in bytes of each buffer (element) when all the buffers are the same length. The length of the entire message is equal to PARTLEN multiplied by the number of elements. All buffers must be at least as long as PARTLEN.

- When MSGACCESS=SYNC, PARTLEN must be in the decimal range of 0 to 62464 (61K) bytes. The length of the entire message must be in the decimal range of 0 to 62464 bytes.
- When MSGACCESS=ASYN or MSGACCESS=SYNCSUSPEND, PARTLEN can be in the decimal range of 0 to 134 217 728 (128M) bytes. The length of the entire message must be in the decimal range of 0 to 128M bytes. If the length of the entire message exceeds 61K bytes, both the sending member and the target member must have specified GT61KMSG=YES when they invoked the IXCJOIN macro to join the group. Use the IXCQUERY service to determine whether a particular target member supports the large message delivery protocol.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword containing the length in bytes of each buffer.

**,PARTLENOFF=partlenoff**

Use this input parameter to specify the number of bytes to be added to the address of the current element, to locate within the element the fullword field that contains the length in bytes of the buffer associated with that element. The length of each buffer can range from 0 to 62464 (61K) bytes in decimal when MSGACCESS=SYNC is specified and from 0 to 134 217 728 (128K) bytes in decimal when MSGACCESS=ASYN or MSGACCESS=SYNCSUSPEND is specified.

If you specify the length of a buffer as zero, XCF does not use that buffer.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword containing the number of bytes.

**,PARTLENTBL=partlentbl**

Use this input parameter to specify a table in which each entry is a fullword containing the length of a corresponding buffer. For instance, the 3rd entry in PARTLENTBL contains the length of the buffer whose address is associated with the 3rd message data element in the table or queue.

- When MSGACCESS=SYNC, the length of an individual message part must be in the decimal range of 0 to 62464 (61K) bytes. The length of the entire message must be in the decimal range of 0 to 62464 bytes.
- When MSGACCESS=ASYN or MSGACCESS=SYNCSUSPEND, the length of an individual message part must be in the decimal range of 0 to 134 217 728 (128M) bytes. The length of the entire message must be in the decimal range of 0 to 128M bytes. If the length of the entire message exceeds 61K bytes, both the sending member and the target member must have specified GT61KMSG=YES when they invoked the IXCJOIN macro to join the group. Use the IXCQUERY service to determine whether a particular target member supports the large message delivery protocol.

If you specify the length of a buffer as zero, XCF does not use that buffer.

The table specified by PARTLENTBL must begin on a fullword boundary.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of *partlentbl*.

**,PARTOFF=partoff**

Use this input parameter to specify the number of bytes to be added to the address of the current element to obtain the address within the element of the start of the buffer associated with it.



**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword containing the number of bytes.

**,PARTPTROFF=partptroff**

Use this input parameter to specify the number of bytes to be added to the address of the current element, to locate within the element either a 31- or 64-bit precision pointer to the buffer associated with it.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword containing the number of bytes.

**,PLISTVER=IMPLIED\_VERSION**

**,PLISTVER=MAX**

**,PLISTVER=plistver**

Use this input parameter to specify the version of the macro. See [“Understanding IXCMGGOX Version Support”](#) on page 187 for a description of the options available with PLISTVER.

**,RESPONSEID=responseid**

Use this input parameter to provide the XCF identifier for the originating message to which this request is making a response.

Obtain the RESPONSEID from one of the following:

- The MEPLRESPONSEID field in the IXCYMEPL that was presented to a message user routine.
- The MQAMIDRESPONSEID field in the IXCYMQAA that was returned by the XCF message control service (IXCMGGO REQUEST=QUERYMSG).

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the 24-byte character field containing the XCF response identifier.

**,RETCODE=retcode**

Use this output parameter to specify the location in which the system is to copy the return code from GPR 15.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword to contain the return code.

**,RETMSGOTOKEN=retmsgotoken**

Use this output parameter to specify a storage area to contain a token that you can use to identify this message to the XCF message control service (IXCMGGO).

When using MSGACCESS=SYNCSUSPEND where the calling work unit must be suspended by XCF, the token is stored before the work unit is actually suspended. If it becomes necessary for the sender to leave the suspend before the service routine resumes and returns, some other work unit can use this token to release, discard, or force the completion of the request by invoking the XCF message control service (IXCMGGO). Note that cancelling the send likely implies that the message will not be delivered to all of the intended targets.

The storage area must be in the caller's primary address space, in an address space or data space that is addressable through a public entry on the caller's dispatchable unit access list (DU-AL), or in a common area data space.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the 16-byte character output field to contain the token.

**,RSNCODE=rsncode**

Use this output parameter to specify the location in which the system is to copy the reason code from GPR 0.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword to contain the reason code.

**,SENDTIME=sendtime**

Use this input parameter to indicate the maximum number of seconds XCF might suspend the work unit to allow XCF to synchronously complete accessing caller's storage areas.

SENDTIME is a required input when MSGACCESS=SYNCSUSPEND. If the SENDTIME value expires before the processing that XCF paused for is complete, XCF cancels incomplete send processing, which might result in not all of the desired targets receiving the send request.

The SENDTIME value must be in the range 1 and 32 767 (X'7FFF'). If TIMEOUT is explicitly specified on the message-out request, the SENDTIME value must be less than or equal to the TIMEOUT keyword value.

**,SENDTO=MEMBER**

**,SENDTO=GROUP**

**,SENDTO=ORIGINATOR**

Use this input parameter to specify the member(s) to which the message should be sent.

**SENDTO=MEMBER**

Send the message to the (one) member indicated by TARGET. This type of send is the default.

**SENDTO=GROUP**

Send the message to a collection of members. This type of send is referred to as a "broadcast".

Note that a send to GROUP where the collection of members consists of only one member does not necessarily have the same behavior as a send to MEMBER. For example, suppose that in both cases the one target member specified was not a valid target. The system would reject the send to MEMBER request with return code X'08', reason code X'08', indicating that the target was not valid. The system would reject the send to GROUP request with return code X'04', reason code X'404', indicating that the broadcast completed but the send to at least one target was rejected. The send reason code reported for the individual target in the SENDTO=GROUP collection would indicate that the target was not valid.

**SENDTO=ORIGINATOR**

Send a response message. This is a response to a previously received message that indicated that the sender had requested that XCF manage the gathering of response(s) to that message.

**,STREAMID=1**

**,STREAMID=streamid**

Use this input parameter to identify the stream of ordered messages to which this message belongs. STREAMID must be a decimal value in the range 0 - 65 535. XCF assigns no meaning to the STREAMID other than for the purpose of providing independently ordered streams of messages between the sending member and target member. The default value is 1.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the fullword field containing the stream identification.

**,TARGET=target**

Use this input parameter to specify the member token of the member to receive the message. See [z/OS MVS Programming: Sysplex Services Guide](#) for information about how to obtain the member token of the receiving member.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 64-bit field that contains the member token.

**,TARGETS=targets**

Use this input parameter to specify the storage area containing the table of member tokens for each intended target member. The storage location named by TARGETS contains the first member token to be processed.

The storage area identified by TARGETS must be in the caller's primary address space, in an address space or data space that is addressable through a public entry on the caller's dispatchable unit access list (DU-AL), or in a common area data space.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the storage area containing the table of member tokens for each intended target member.

**,TIMEOUT=ZERO****,TIMEOUT=timeout**

Use this input parameter to specify the number of seconds the user is willing to allow for the message to complete. (See the NOTIFY keyword for a definition of message completion.) The timeout value specified must be between 1 and 32,767 (X'7FFF').

If the message has not completed before the expiration of the timeout value, the message is declared to be complete. The system then processes the message completion as specified by the NOTIFY keyword.

A nonzero timeout value is required when MSGACCESS=ASYNCR is specified.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the halfword field containing the timeout value.

**,USERDATA=ALLZERO****,USERDATA=userdata**

Use this input parameter to specify eight characters of user data to be associated with the message. The system presents this data in the message notification parameter list (IXCYMNPL) when the message is presented to the message notify user routine. The user data is also available through the IXCMSGC REQUEST=QUERYMSG service with DATATYPE=MSGOUT.

The user data is not presented to the targets of the message.

The default of ALLZERO consists of hexadecimal zeros.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the eight-character field containing the user data.

**,#MSGPARTS=ATLEASTONE****,#MSGPARTS=msgparts**

Use this input parameter with MULTIPART=YES to specify the number of buffers (1 or more) containing the message data to be sent.

If you specify the default value (#MSGPARTS=ATLEASTONE), one message part is the minimum, or more as needed to send MSGLEN bytes.

If you omit #MSGPARTS, the result depends on the following:

- If you specify MSGLEN, the buffer storage you provide must be equal to or greater than the value of MSGLEN.
- If you omit MSGLEN and specify a table of message data elements (ELEMFORM=TABLE), XCF processes only the first message data element.
- If you omit MSGLEN and specify a queue of message data elements (ELEMFORM=QUEUE), XCF processes message data elements until it reaches the end of the queue. See the ENDOFQUEUE parameter description to find out how XCF detects the end of the queue.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword containing the number of buffers.

**,#TARGETS=#targets**

Use this input parameter to specify the number of entries in the TARGETS table. The value specified must be between 1 and the maximum number of members per XCF group supported by the sysplex, inclusive.

The system programmer determines the maximum number of members per XCF group supported by the sysplex when using the XCF format utility (IXCL1DSU) to create the sysplex couple data set for the sysplex. In effect, the message can be broadcast to every member that can successfully invoke IXCJOIN to join the XCF group.

Note that #TARGETS indicates the number of potential targets. A TARGETS table entry that contains a member token of X'0' is considered a potential target and should be included in the #TARGETS count.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the fullword field containing the number of entries in the TARGET table.

## ABEND Codes

Abend codes an issuer of IXCMSGOX might receive are listed below. For detailed abend code information, see *z/OS MVS System Codes*.

X'073'

X'C78'

## Return and reason codes

When the IXCMSGOX macro returns control to your program:

- GPR 15 (and *retcode*, if you coded RETCODE) contains a return code.
- GPR 0 (and *rsncode*, if you coded RSNCODE) contains a reason code, if applicable.

Macro IXCYCON provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

**0**

IXCRETCODEOK

**4**

IXCRETCODEWARNING

**8**

IXCRETCODEPARMERROR

**C**

IXCRETCODEENVERROR

**10**

IXCRETCODECOMPERROR

The following table identifies the hexadecimal return and reason codes and the equate symbol associated with each reason code.

Table 13. Return and reason codes for the IXCMSGOX macro		
Hexadecimal return code	Hexadecimal reason code	Equate Symbol meaning and action
00	None.	<b>Meaning:</b> IXCMSGOX completed successfully; XCF accepts the message for delivery. <b>Action:</b> None.
04	401	<b>Equate Symbol:</b> IXMSGORSNSENDPENDING <b>Meaning:</b> Send message pending. The system could not initiate the send to the requested target member immediately and the send is pending. XCF continues trying to send the message until the specified TIMEOUT value expires. <b>Action:</b> Discontinue sending the message until the condition clears. The system will drive the message notify user routine when the send completes if notification was requested.

Table 13. Return and reason codes for the IXCMGGOX macro (continued)

Hexadecimal return code	Hexadecimal reason code	Equate Symbol meaning and action
04	402	<p><b>Equate Symbol:</b> IXCMGORSNBCEPENDINGNOREJECTS</p> <p><b>Meaning:</b> Broadcast pending, no rejections. The system could not initiate the send to one or more target members immediately and the send is pending. No send was rejected. XCF continues trying to send the pending message(s) until the specified TIMEOUT value expires.</p> <p><b>Action:</b> Discontinue sending the message until the condition clears. The system will drive the message notify user routine when the send completes.</p>
04	403	<p><b>Equate Symbol:</b> IXCMGORSNBCEPENDINGWITHREJECTS</p> <p><b>Meaning:</b> Broadcast pending, some send(s) rejected. The system could not initiate the send to one or more target members immediately and the send is pending. The send for at least one target member was rejected. XCF continues trying to send the pending message(s) until the specified TIMEOUT value expires.</p> <p><b>Action:</b> Discontinue sending the message until the condition clears. The system will drive the message notify user routine when the send completes.</p>
04	404	<p><b>Equate Symbol:</b> IXCMGORSNBCCOMPLETewithREJECTS</p> <p><b>Meaning:</b> Broadcast completed, some sends rejected. No sends to target members are pending. The send for at least one target member was rejected.</p> <p><b>Action:</b> None.</p>
04	0405xxxx	<p><b>Equate Symbol:</b> IXCMGORSNRETMSTOKENNOACCESS</p> <p><b>Meaning:</b> RETMSTOKEN is not accessible. IXCMGGOX was unable to store a message token in the area indicated by RETMSTOKEN. The low order halfword (xxxx) of the reason code has a value of zero if the return code was X'00' or indicates the reason code associated with return code X'04' that the system would have returned if the storage area indicated by RETMSTOKEN had been accessible. Note that even though the system was unable to store the RETMSTOKEN token, if the return code is X'00', the system has initiated the send of the message, or if the return code is X'04', the system has accepted the message for delivery.</p> <p><b>Action:</b> Verify that the storage area indicated by RETMSTOKEN is either in the caller's primary address space or in an address or data space that is addressable through a public entry on the caller's dispatchable unit address list (DU-AL) or is in a common area data space.</p>

Table 13. Return and reason codes for the IXCMSGOX macro (continued)

Hexadecimal return code	Hexadecimal reason code	Equate Symbol meaning and action
04	410	<p><b>Equate Symbol:</b> IXCMSGORSNASYNCSENDPENDING</p> <p><b>Meaning:</b> A send message is pending. The send to a requested target member could not be initiated immediately and is pending. XCF will continue trying to send the message until the specified TIMEOUT value expires. This reason code is applicable only when MSGACCESS=ASYN is specified.</p> <p><b>Action:</b> Preserve the message text until the send of the message is complete. For a multipart message, the queue or table elements that describe how to locate the message must also be preserved. If NOTIFY=YES is specified, the system will drive the message notify user routine when the message is completed.</p>
08	04	<p><b>Equate Symbol:</b> IXCMSGORSNSENDERNOTVALID</p> <p><b>Meaning:</b> Program error. One of the following happened:</p> <ul style="list-style-type: none"> <li>• The sending member token does not identify an active member associated with the primary address space that was current when the IXCMSGOX service was invoked.</li> <li>• The primary address space is not the master scheduler's address space.</li> <li>• A member has terminated or placed itself in a not-defined state.</li> </ul> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• Ensure that the correct sending member token was specified.</li> <li>• Ensure that the parameter list was not inadvertently overlaid and that it was correctly specified.</li> <li>• Ensure that the IXCMSGOX service is being called either from the address space that was current when the sending member issued IXCJOIN to join its group or from the master scheduler address space.</li> </ul> <p>Any further action depends on your application.</p>

Table 13. Return and reason codes for the IXCMGGOX macro (continued)

Hexadecimal return code	Hexadecimal reason code	Equate Symbol meaning and action
08	08	<p><b>Equate Symbol:</b> IXCMGORSNTARGETNOTVALID</p> <p><b>Meaning:</b> Program error. The token of the target member either:</p> <ul style="list-style-type: none"> <li>• Represents a member in a different XCF group.</li> <li>• Represents a member that is not active.</li> <li>• Is not a valid XCF member token.</li> </ul> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• Ensure that the target and source member tokens represent members of the same XCF group. Messages cannot be sent between members of different groups.</li> <li>• If the target member is not active, any action taken depends on your application.</li> <li>• Ensure that the correct target member token was specified.</li> <li>• Ensure that the parameter list was not inadvertently overlaid and that it was correctly specified.</li> </ul>
08	0C	<p><b>Equate Symbol:</b> IXCMGORSNMSGLENNOTVALID</p> <p><b>Meaning:</b> Program error. For MSGACCESS=SYNC, the total message length is not in the decimal range of 0 to 62464. For MSGACCESS=ASYN or MSGACCESS=SYNCSUSPEND, the total message length is not in the decimal range of 0 to 134 217 728 or the sending member did not specify YES for the GT61KMSG keyword when it invoked IXCJOIN to join the group.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• Ensure that the correct message length was specified.</li> <li>• The maximum message size is 62464 bytes. If your message is larger than 62464 bytes, you will need to break it up into parts no larger than 62464, and send each of them separately.</li> <li>• Ensure that the parameter list was not inadvertently overlaid and that it was correctly specified.</li> </ul>
08	10	<p><b>Equate Symbol:</b> IXCMGORSNMSGBUFBADSTG</p> <p><b>Meaning:</b> Program error. XCF could not access the message buffer.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• Ensure that the correct message buffer storage area was used.</li> <li>• If your program is running in AR mode, ensure that: <ul style="list-style-type: none"> <li>– The ALET for the message buffer is correct.</li> <li>– You specified SYSSTATE ASCENV=AR before issuing the IXCMGGOX macro.</li> </ul> </li> <li>• Ensure that the message buffer storage area was not inadvertently freed by your program.</li> </ul>

Table 13. Return and reason codes for the IXCMGGOX macro (continued)

Hexadecimal return code	Hexadecimal reason code	Equate Symbol meaning and action
08	14	<p><b>Equate Symbol:</b> IXCMGORSNMSGCNTLBADALET</p> <p><b>Meaning:</b> Program error. The address of the message control information (MSGCNTL) is qualified by an ALET that is not valid. The ALET must be zero, a public entry on your DU-AL, or an entry for a common area data space.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• If your program is running in AR mode, ensure that you specified SYSSTATE ASCENV=AR before issuing the IXCMGGOX macro.</li> <li>• Ensure that the ALET for the message control information is correct.</li> </ul>
08	18	<p><b>Equate Symbol:</b> IXCMGORSNMSGCNTLBADSTG</p> <p><b>Meaning:</b> Program error. The system cannot access the area that contains the message control information (MSGCNTL).</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• Ensure that the address of the message control information is correct.</li> <li>• If your program is running in AR mode, ensure that: <ul style="list-style-type: none"> <li>– The ALET for the message control information is correct.</li> <li>– You specified SYSSTATE ASCENV=AR before issuing the IXCMGGOX macro.</li> </ul> </li> </ul>
08	1C	<p><b>Equate Symbol:</b> IXCMGORSNTARGETNOMSGEXIT</p> <p><b>Meaning:</b> Program error. The member specified on the TARGET parameter does not have a message user routine.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• Ensure that the correct TARGET member token was specified.</li> <li>• Ensure that the TARGET member specified a message user routine (MSGEXIT keyword on IXCJOIN) when joining the XCF group.</li> </ul>
08	40	<p><b>Equate Symbol:</b> IXCMGORSNPLISTRSDNOTVALID</p> <p><b>Meaning:</b> Program error. A reserved field in the control parameter list was not zero. The parameter list has been corrupted or the release level of XCF on which the caller is running does not support the provided XCF message-out service parameter list.</p> <p><b>Action:</b> Check to see if your program inadvertently overlaid the parameter list storage, and that it was assembled with the correct macro library for the release of z/OS on which your program is running.</p>



Table 13. Return and reason codes for the IXCMGGOX macro (continued)

Hexadecimal return code	Hexadecimal reason code	Equate Symbol meaning and action
08	100	<b>Equate Symbol:</b> IXCMGORSNPLISTBADALET <b>Meaning:</b> Program error. The ALET that qualifies the address of the control parameter list was not valid. The ALET must be zero. <b>Action:</b> Ensure that: <ul style="list-style-type: none"> <li>• You specified SYSSTATE ASCENV=AR before issuing the IXCMGGOX macro.</li> <li>• The ALET for the control parameter list storage is correct.</li> </ul>
08	104	<b>Equate Symbol:</b> IXCMGORSNPLISTVERSIONNOTVALID <b>Meaning:</b> Program error. The version number in the control parameter list was not valid. <b>Action:</b> Check for errors such as the following: <ul style="list-style-type: none"> <li>• Your program overlaid the control parameter list storage.</li> <li>• Your program was assembled with the wrong macro library for the release of z/OS on which your program is running.</li> </ul>
08	10C	<b>Equate Symbol:</b> IXCMGORSNPLISTBADSTG <b>Meaning:</b> Program error. XCF could not access the control parameter list. <b>Action:</b> Ensure that: <ul style="list-style-type: none"> <li>• The correct parameter list storage area was specified.</li> <li>• The parameter list storage area was not inadvertently freed by your program.</li> </ul>
08	11C	<b>Equate Symbol:</b> IXCMGORSNNOTENABLED <b>Meaning:</b> Program error. The caller is not enabled. <b>Action:</b> Correct your program so that it does not issue IXCMGGOX while it is disabled.
08	12C	<b>Equate Symbol:</b> IXCMGORSNLOCKHELD <b>Meaning:</b> Program error. The caller of IXCMGGOX holds a lock. <b>Action:</b> Correct your program so that it does not issue IXCMGGOX while it is holding a lock.
08	208	<b>Equate Symbol:</b> IXCMGORSNMSGBUFBADALET <b>Meaning:</b> Program error. The ALET that qualifies the address of the buffer specified by MSGBUF must be zero, a valid public entry on your DU-AL, or a common area data space. <b>Action:</b> Check for errors such as the following: <ul style="list-style-type: none"> <li>• You did not specify SYSSTATE ASCENV=AR before issuing the IXCMGGOX macro.</li> <li>• You specified the wrong ALET for the buffer.</li> </ul>

Table 13. Return and reason codes for the IXCMGGOX macro (continued)

Hexadecimal return code	Hexadecimal reason code	Equate Symbol meaning and action
08	20C	<p><b>Equate Symbol:</b> IXCMGORSNMSGBUFKEYMISMATCH</p> <p><b>Meaning:</b> Program error. The service could not fetch the message data from the buffer area specified by MSGBUF using the storage key specified by MSGSTGKEY.</p> <p><b>Action:</b> Check for errors such as the following:</p> <ul style="list-style-type: none"> <li>• The buffer area was allocated in storage whose storage key does not match the key specified by the MSGSTGKEY parameter.</li> <li>• The storage key specified by MSGSTGKEY has been overlaid</li> <li>• The storage key specified by MSGSTGKEY is not valid.</li> <li>• The address of the buffer area is incorrect.</li> </ul>
08	210	<p><b>Equate Symbol:</b> IXCMGORSNPARTPTROFFBADSTG</p> <p><b>Meaning:</b> Program error. An element is not accessible. The address located within an element at the offset specified by PARTPTROFF was not valid. In the parameter list generated by IXCMGGOX :</p> <ul style="list-style-type: none"> <li>• The field, PART#, indicates the index of the element containing the invalid address. PART# values start with 1.</li> <li>• The field, ELEMENTPTR, contains the address of the element containing the invalid address.</li> </ul> <p><b>Action:</b> Check for errors such as the following:</p> <ul style="list-style-type: none"> <li>• You specified an incorrect element address.</li> <li>• Your program overlaid the element address.</li> <li>• The buffer had been previously freed.</li> <li>• You specified an incorrect number of message parts.</li> <li>• The message data element is damaged or has not maintained the integrity of its data.</li> <li>• You specified an incorrect end-of-queue value.</li> <li>• If your program is running in AR mode, check for the following: <ul style="list-style-type: none"> <li>– You specified the wrong ALET for the element.</li> <li>– You did not specify SYSSTATE ASCENV=AR before issuing the IXCMGGOX macro.</li> </ul> </li> </ul>
08	212	<p><b>Equate Symbol:</b> IXCMGORSNELEMENTBADALET</p> <p><b>Meaning:</b> Program error. The ALET that qualifies the address of an ELEMENT must be zero, a public entry on your DU-AL, or an entry for a common area data space.</p> <p><b>Action:</b> Check for errors such as the following:</p> <ul style="list-style-type: none"> <li>• You did not specify SYSSTATE ASCENV=AR before issuing the IXCMGGOX macro.</li> <li>• You specified the wrong ALET for the buffer.</li> </ul>

Table 13. Return and reason codes for the IXCMSGOX macro (continued)

Hexadecimal return code	Hexadecimal reason code	Equate Symbol meaning and action
08	213	<p><b>Equate Symbol:</b> IXCMSGORSNNEXTPTROFFBADSTG</p> <p><b>Meaning:</b> Program error. A pointer to the next queue element, at the offset specified by NEXTPTROFF, was not valid. In the parameter list generated by IXCMSGOX :</p> <ul style="list-style-type: none"> <li>• The field, PART#, indicates the index of the element containing the incorrect address. PART# values start with 1.</li> <li>• The field, ELEMENTPTR, contains the address of the element containing the incorrect address.</li> </ul> <p><b>Action:</b> Check for errors such as the following:</p> <ul style="list-style-type: none"> <li>• You specified an incorrect element address.</li> <li>• Your program overlaid the element address.</li> <li>• You specified an incorrect number of message parts.</li> <li>• The message data element is damaged or has not maintained the integrity of its data.</li> <li>• You specified an incorrect end-of-queue value.</li> </ul>
08	214	<p><b>Equate Symbol:</b> IXCMSGORSN#MSGPARTSZERO</p> <p><b>Meaning:</b> Program error. The value specified by the #MSGPARTS parameter was zero but must be greater than zero.</p> <p><b>Action:</b> Specify a value greater than zero for #MSGPARTS.</p>
08	215	<p><b>Equate Symbol:</b> IXCMSGORSNTOOMANYZEROLENPARTS</p> <p><b>Meaning:</b> Program error. The message data element table or queue is assumed to be damaged. You omitted the #MSGPARTS parameter, and XCF processed more than 65,536 consecutive elements with buffers of length zero.</p> <p><b>Action:</b> Check for errors such as the following:</p> <ul style="list-style-type: none"> <li>• You specified PARTLEN with a value of zero.</li> <li>• You specified PARTLENTBL but the address of the table was not correct.</li> <li>• You specified PARTLENOFF with an incorrect value.</li> <li>• Ensure that the elements have not been overlaid.</li> <li>• You specified an incorrect end-of-queue value.</li> </ul> <p>Specify the #MSGPARTS parameter when invoking IXCMSGOX.</p>

Table 13. Return and reason codes for the IXCMSGOX macro (continued)

Hexadecimal return code	Hexadecimal reason code	Equate Symbol meaning and action
08	218	<p><b>Equate Symbol:</b> IXCMSGORSNPARTPTROFF@BADSTG</p> <p><b>Meaning:</b> Program error. A buffer whose address was at the offset specified by PARTPTROFF could not be accessed. In the parameter list generated by IXCMSGOX :</p> <ul style="list-style-type: none"> <li>• The field, PART#, indicates the index of the element containing the incorrect address. PART# values start with 1.</li> <li>• The field, ELEMENTPTR, contains the address of the element containing the incorrect address.</li> </ul> <p><b>Action:</b> Check for following types of errors:</p> <ul style="list-style-type: none"> <li>• The buffer address was incorrect.</li> <li>• The buffer was previously freed.</li> <li>• You specified PARTPTROFF with an incorrect value.</li> <li>• If your program is running in AR mode, check for the following types or errors: <ul style="list-style-type: none"> <li>– You specified the wrong ALET for the buffer.</li> <li>– You did not specify SYSSTATE ASCENV=AR before issuing the IXCMSGOX macro.</li> </ul> </li> </ul>
08	219	<p><b>Equate Symbol:</b> IXCMSGORSNPARTOFFBADSTG</p> <p><b>Meaning:</b> Program error. A buffer located at the offset specified by PARTOFF could not be accessed. In the parameter list generated by IXCMSGOX :</p> <ul style="list-style-type: none"> <li>• The field, PART#, indicates the index of the element containing the buffer that could not be accessed. PART# values start with 1.</li> <li>• The field, ELEMENTPTR, contains the address of the element containing the buffer that could not be accessed.</li> </ul> <p><b>Action:</b> Check for following types of errors:</p> <ul style="list-style-type: none"> <li>• The storage containing the element has been freed.</li> <li>• The PARTOFF value is incorrect.</li> <li>• The storage containing the element has been overlaid.</li> <li>• You specified an incorrect number of message parts.</li> <li>• The message data element is damaged or has not maintained the integrity of its data.</li> <li>• You specified an incorrect end-of-queue value.</li> </ul>

Table 13. Return and reason codes for the IXCMGGOX macro (continued)

Hexadecimal return code	Hexadecimal reason code	Equate Symbol meaning and action
08	21C	<p><b>Equate Symbol:</b> IXCMGORSNPARTPTROFF@KEYMISMATCH</p> <p><b>Meaning:</b> Program error. XCF could not fetch the message data from a buffer whose address was at the offset specified by PARTPTROFF using the storage key specified by MSGSTGKEY. In the parameter list generated by IXCMGGOX :</p> <ul style="list-style-type: none"> <li>• The field, PART#, indicates the index of the element containing the address of the buffer from which the message data could not be fetched. PART# values start with 1.</li> <li>• The field, ELEMENTPTR, contains the address of the element containing the address of the buffer from which the message data could not be fetched.</li> </ul> <p><b>Action:</b> Check for errors such as the following:</p> <ul style="list-style-type: none"> <li>• The buffer was allocated in storage whose storage key did not match the key specified by the MSGSTGKEY parameter.</li> <li>• You specified PARTPTROFF with an incorrect value.</li> <li>• The address of the buffer is not correct.</li> </ul>
08	21D	<p><b>Equate Symbol:</b> IXCMGORSNPARTOFFKEYMISMATCH</p> <p><b>Meaning:</b> Program error. XCF could not fetch the message data from a buffer area located at the offset specified by PARTOFF using the storage key specified by MSGSTGKEY. In the parameter list generated by IXCMGGOX :</p> <ul style="list-style-type: none"> <li>• The field, PART#, indicates the index of the element containing the buffer from which the message data could not be fetched. PART# values start with 1.</li> <li>• The field, ELEMENTPTR, contains the address of the element containing the buffer from which the message data could not be fetched.</li> </ul> <p><b>Action:</b> Check for errors such as the following:</p> <ul style="list-style-type: none"> <li>• The buffer area was allocated in storage whose storage key did not match the key specified by the MSGSTGKEY parameter.</li> </ul>
08	220	<p><b>Equate Symbol:</b> IXCMGORSNPARTLENTBLBADSTG</p> <p><b>Meaning:</b> Program error. The table specified by PARTLENTBL could not be accessed.</p> <p><b>Action:</b> Check for errors such as the following:</p> <ul style="list-style-type: none"> <li>• PARTLENTBL was inadvertently freed.</li> <li>• The element was overlaid.</li> <li>• The address of the table specified by PARTLENTBL is not correct.</li> </ul>

Table 13. Return and reason codes for the IXCMGGOX macro (continued)

Hexadecimal return code	Hexadecimal reason code	Equate Symbol meaning and action
08	221	<p><b>Equate Symbol:</b> IXCMGORSNPARTLENTBLNOTWORDBDY</p> <p><b>Meaning:</b> Program error. The table specified by PARTLENTBL does not begin on a fullword boundary.</p> <p><b>Action:</b> Ensure that the table specified by PARTLENTBL begins on a fullword boundary.</p>
08	222	<p><b>Equate Symbol:</b> IXCMGORSNPARTLENTBLBADALET</p> <p><b>Meaning:</b> Program error. The ALET that qualifies the address of PARTLENTBL must be zero, a public entry on your DU-AL, or an entry for a common area data space.</p> <p><b>Action:</b> Check for errors such as the following:</p> <ul style="list-style-type: none"> <li>You did not specify SYSSTATE ASCENV=AR before issuing the IXCMGGOX macro.</li> <li>You specified the wrong ALET for PARTLENTBL.</li> </ul>
08	223	<p><b>Equate Symbol:</b> IXCMGORSNPARTLENOFFBADSTG</p> <p><b>Meaning:</b> Program error. The buffer length located at the offset specified by PARTLENOFF could not be accessed within an element. In the parameter list generated by IXCMGGOX :</p> <ul style="list-style-type: none"> <li>The field, PART#, indicates the index of the element containing the buffer length that could not be accessed. PART# values start with 1.</li> <li>The field, ELEMENTPTR, contains the address of the element containing the buffer length that could not be accessed.</li> </ul> <p><b>Action:</b> Check for errors such as the following:</p> <ul style="list-style-type: none"> <li>The storage containing the element has been freed.</li> <li>The PARTLENOFF value is incorrect.</li> <li>The storage containing the element has been overlaid.</li> <li>The message data elements were not properly constructed or serialized.</li> <li>You specified an incorrect number of message parts.</li> <li>The message data element is damaged or has not maintained the integrity of its data.</li> <li>You specified an incorrect end-of-queue value.</li> </ul>
08	224	<p><b>Equate Symbol:</b> IXCMGORSNMSGLENGTSUMPARTLEN</p> <p><b>Meaning:</b> Program error. The message length specified by MSGLEN was not valid. The message length specified by MSGLEN is larger than the sum of the buffer lengths.</p> <p><b>Action:</b> Ensure that the message length specified by MSGLEN is less than or equal to the sum of the buffer lengths.</p>

Table 13. Return and reason codes for the IXCMGGOX macro (continued)

Hexadecimal return code	Hexadecimal reason code	Equate Symbol meaning and action
08	225	<p><b>Equate Symbol:</b> IXCMGORSNPARTLENBADLEN</p> <p><b>Meaning:</b> Program error. The buffer length specified by PARTLEN was not valid. For MSGACCESS=SYNC, the value must be less than or equal to 62464. For MSGACCESS=ASYN or MSGACCESS=SYNCSUSPEND, the value must be less than or equal to 134217728.</p> <p>In the parameter list generated by IXCMGGOX :</p> <ul style="list-style-type: none"> <li>• The field, PART#, indicates the index of the element containing the buffer length that was not valid. PART# values start with 1.</li> <li>• The field, ELEMENTPTR, contains the address of the element containing the buffer length that was not valid.</li> </ul> <p><b>Action:</b> Ensure that the value specified by PARTLEN is less than or equal to 62464.</p>
08	226	<p><b>Equate Symbol:</b> IXCMGORSNPARTLENTBLBADLEN</p> <p><b>Meaning:</b> Program error. A buffer length contained in the table specified by PARTLENTBL was not valid. For MSGACCESS=SYNC, the buffer lengths must be less than or equal to 62464. For MSGACCESS=ASYN or MSGACCESS=SYNCSUSPEND, the buffer lengths must be less than or equal to 134 217 728.</p> <p>In the parameter list generated by IXCMGGOX :</p> <ul style="list-style-type: none"> <li>• The field, PART#, indicates the index of the element containing the buffer length that was not valid. PART# values start with 1.</li> <li>• The field, ELEMENTPTR, contains the address of the element containing the buffer length that was not valid.</li> </ul> <p><b>Action:</b> Ensure that all buffer lengths are less than or equal to 62464.</p>
08	227	<p><b>Equate Symbol:</b> IXCMGORSNPARTLENOFFBADLEN</p> <p><b>Meaning:</b> Program error. The buffer length located at the offset specified by PARTLENOFF was not valid. For MSGACCESS=SYNC, the buffer lengths must be less than or equal to 62464. For MSGACCESS=ASYN or MSGACCESS=SYNCSUSPEND, the buffer lengths must be less than or equal to 134217728. In the parameter list generated by IXCMGGOX:</p> <ul style="list-style-type: none"> <li>• The field, PART#, indicates the index of the element containing the invalid buffer length. PART# values start with 1.</li> <li>• The field, ELEMENTPTR, contains the address of the element containing the invalid buffer length.</li> </ul> <p><b>Action:</b> Ensure that the buffer length is less than or equal to 62464.</p>

Table 13. Return and reason codes for the IXCMGGOX macro (continued)

Hexadecimal return code	Hexadecimal reason code	Equate Symbol meaning and action
08	230	<p><b>Equate Symbol:</b> IXCMGORSNPARTALETTBLBADSTG</p> <p><b>Meaning:</b> Program error. The table specified by PARTALETTBL could not be accessed.</p> <p><b>Action:</b> Check for errors such as the following:</p> <ul style="list-style-type: none"> <li>• PARTALETTBL was inadvertently freed.</li> <li>• PARTALETTBL was overlaid or incorrect.</li> </ul>
08	231	<p><b>Equate Symbol:</b> IXCMGORSNPARTALETTBLNOTWORDBDY</p> <p><b>Meaning:</b> Program error. The table specified by PARTALETTBL does not begin on a fullword boundary.</p> <p><b>Action:</b> Ensure that the table specified by PARTALETTBL begins on a fullword boundary.</p>
08	232	<p><b>Equate Symbol:</b> IXCMGORSNPARTALETTBLBADALET</p> <p><b>Meaning:</b> Program error. The ALET that qualifies the address of PARTALETTBL must be zero, a public entry on your DU-AL, or an entry for a common area data space.</p> <p><b>Action:</b> Check for errors such as the following:</p> <ul style="list-style-type: none"> <li>• You did not specify SYSSTATE ASCENV=AR before issuing the IXCMGGOX macro.</li> <li>• You specified the wrong ALET for PARTALETTBL.</li> </ul>
08	233	<p><b>Equate Symbol:</b> IXCMGORSNPARTALETOFFBADSTG</p> <p><b>Meaning:</b> Program error. The ALET located at the offset specified by PARTALETOFF could not be accessed. In the parameter list generated by IXCMGGO:</p> <ul style="list-style-type: none"> <li>• The field, PART#, indicates the index of the element containing the ALET that could not be accessed. PART# values start with 1.</li> <li>• The field, ELEMENTPTR, contains the address of the element containing the ALET that could not be accessed.</li> </ul> <p><b>Action:</b> Check for errors such as the following:</p> <ul style="list-style-type: none"> <li>• The storage containing the element has been freed.</li> <li>• The PARTALETOFF value is incorrect.</li> <li>• The storage containing the element has been overlaid.</li> <li>• The message data elements were not properly constructed or serialized.</li> </ul>
08	234	<p><b>Equate Symbol:</b> IXCMGORSNPARTALET@BADALET</p> <p><b>Meaning:</b> Program error. The ALET specified by PARTALET was not valid. It must be zero, a public entry on your DU-AL, or an entry for a common area data space.</p> <p><b>Action:</b> Correct the ALET.</p>



Table 13. Return and reason codes for the IXCMGGOX macro (continued)

Hexadecimal return code	Hexadecimal reason code	Equate Symbol meaning and action
08	235	<p><b>Equate Symbol:</b> IXCMGORSNPARTALETTBL@BADALET</p> <p><b>Meaning:</b> Program error. An ALET specified in a PARTALETTBL entry was not valid. Each ALET must be zero, a public entry on your DU-AL, or an entry for a common area data space. In the parameter list generated by IXCMGGOX :</p> <ul style="list-style-type: none"> <li>• The field, PART#, indicates the index of the element containing the ALET that was not valid. PART# values start with 1.</li> <li>• The field, ELEMENTPTR, contains the address of the element containing the ALET that was not valid.</li> </ul> <p><b>Action:</b> Correct the ALET.</p>
08	236	<p><b>Equate Symbol:</b> IXCMGORSNPARTALETOFF@BADALET</p> <p><b>Meaning:</b> Program error. An ALET specified at the offset designated by PARTALETOFF was not valid. It must be zero, a public entry on your DU-AL, or an entry for a common area data space. In the parameter list generated by IXCMGGOX :</p> <ul style="list-style-type: none"> <li>• The field, PART#, indicates the index of the element containing the ALET that was not valid. PART# values start with 1.</li> <li>• The field, ELEMENTPTR, contains the address of the element containing the ALET that was not valid.</li> </ul> <p><b>Action:</b> Correct the ALET or check for errors such as the following:</p> <ul style="list-style-type: none"> <li>• The PARTALETOFF value is incorrect.</li> <li>• You specified an incorrect number of message parts.</li> <li>• The message data element is damaged or has not maintained the integrity of its data.</li> <li>• You specified an incorrect end-of-queue value.</li> </ul>
08	300	<p><b>Equate Symbol:</b> IXCMGORSNSENDERNONOTIFYEXIT</p> <p><b>Meaning:</b> Program error. NOTIFYEXIT is not defined. The value of NOTIFYEXIT defaulted to the address of the exit routine defined when the member joined the group, but no NOTIFYEXIT was defined when the IXCJOIN service was invoked.</p> <p><b>Action:</b> Code the NOTIFYEXIT explicitly or add the NOTIFYEXIT definition to the IXCJOIN invocation.</p>

Table 13. Return and reason codes for the IXCMGGOX macro (continued)

Hexadecimal return code	Hexadecimal reason code	Equate Symbol meaning and action
08	304	<p><b>Equate Symbol:</b> IXCMGORSNTARGETSBADALET</p> <p><b>Meaning:</b> Program error. The ALET that qualifies the address of the TARGETS table is neither zero or a valid entry on the caller's Dispatchable Unit Access List (DU-AL) nor a valid entry for a common area data space.</p> <p><b>Action:</b> Check for errors such as the following:</p> <ul style="list-style-type: none"> <li>• You did not specify SYSSTATE ASCENV=AR before issuing the IXCMGGOX macro.</li> <li>• You specified the wrong ALET for TARGETS.</li> </ul>
08	308	<p><b>Equate Symbol:</b> IXCMGORSNRETMMSGOTOKENBADALET</p> <p><b>Meaning:</b> Program error. The ALET that qualifies the address of the RETMSGOTOKEN is neither zero or a valid entry on the caller's Dispatchable Unit Access List (DU-AL) nor a valid entry for a common area data space.</p> <p><b>Action:</b> Check for errors such as the following:</p> <ul style="list-style-type: none"> <li>• You did not specify SYSSTATE ASCENV=AR before issuing the IXCMGGOX macro.</li> <li>• You specified the wrong ALET for RETMSGOTOKEN.</li> </ul>
08	30C	<p><b>Equate Symbol:</b> IXCMGORSNBADRESPONSEID</p> <p><b>Meaning:</b> Program error. The RESPONSEID is not valid. The RESPONSEID token has been corrupted.</p> <p><b>Action:</b> Ensure that the RESPONSEID token that was provided in the message exit parameter list (MEPL) is the token you are using to respond to the message.</p> <p>Also verify that the storage containing the RESPONSEID token has not been inadvertently freed or overlaid.</p>
08	310	<p><b>Equate Symbol:</b> IXCMGORSNBADSTREAMID</p> <p><b>Meaning:</b> Program error. The value specified for STREAMID is not valid. STREAMID must contain a decimal value in the range 0 to 65535.</p> <p><b>Action:</b> Verify that STREAMID contains a value in the range of 0 to 65535 (decimal).</p>
08	314	<p><b>Equate Symbol:</b> IXCMGORSNTARGETSBADSTG</p> <p><b>Meaning:</b> Program error. The TARGETS table is not accessible.</p> <p><b>Action:</b> Check for errors such as the following:</p> <ul style="list-style-type: none"> <li>• TARGETS was inadvertently freed.</li> <li>• TARGETS was overlaid or incorrect.</li> </ul>

Table 13. Return and reason codes for the IXCMGGOX macro (continued)

Hexadecimal return code	Hexadecimal reason code	Equate Symbol meaning and action
08	320	<p><b>Equate Symbol:</b> IXCMGORSNBAD#TARGETS</p> <p><b>Meaning:</b> Program error. The value specified for #TARGETS is not valid. The number of targets must be a nonzero value less than or equal to the maximum number of members per group supported by the sysplex.</p> <p>Note that the maximum number of members per XCF group is determined by the system programmer when the XCF format utility (IXCL1DSU) is used to create the sysplex couple data set that is being used by the sysplex. #TARGETS cannot be larger than the maximum number of members permitted to join the group.</p> <p><b>Action:</b> Correct the #TARGETS value to be within the allowable range.</p> <p>Use the IXCQUERY service to determine the number of members currently in the group.</p>
08	324	<p><b>Equate Symbol:</b> IXCMGORSNBADTIMEOUT</p> <p><b>Meaning:</b> Program error. The value specified for TIMEOUT is not valid. The timeout value must be in the range of 1 to X'7FFF', inclusive. If MSGACCESS=ASYNCR is coded, a nonzero timeout value must be provided.</p> <p><b>Action:</b> Verify that the value specified for TIMEOUT is in the range of 1 to 32,767 (X'7FFF'), inclusive.</p>
08	340	<p><b>Equate Symbol:</b> IXCMGORSNTARGETMAXMSGLEN61K</p> <p><b>Meaning:</b> Program error. The message length is not valid for the target. Either the target member or the system on which the target member resides does not support messages larger than 62462 bytes (decimal). The message was not sent.</p> <p><b>Action:</b> Use the IXCQUERY service to determine which members reside on systems capable of sending or receiving messages larger than 62462 bytes. For XCF to deliver a message greater than 62462 bytes in length, the target member must reside on a system that supports 128MB message delivery, and the target member must specify GT61KMSG=YES when it invokes the IXCJOIN macro to join its group.</p>
08	344	<p><b>Equate Symbol:</b> IXCMGORSNSENDERBECAMEINACTIVE</p> <p><b>Meaning:</b> The sending member became inactive during the message-out request. The message-out request is terminated. Some, none, or all of the targets may receive the message.</p> <p><b>Action:</b> None.</p>

Table 13. Return and reason codes for the IXCMGGOX macro (continued)

Hexadecimal return code	Hexadecimal reason code	Equate Symbol meaning and action
08	348	<p><b>Equate Symbol:</b> IXCMGGOXRSNBADSENDDTIME</p> <p><b>Meaning:</b> Program Error. The SENDTIME value specified is not valid. The SENDTIME value must be between 1 and 32767 inclusive. If TIMEOUT is coded, the SENDTIME value must be less than or equal to the TIMEOUT value specified for the request.</p> <p><b>Action:</b> Ensure that the SENDTIME value is in the range of 1 - 32767 and less than or equal to TIMEOUT if a non-zero TIMEOUT value was specified.</p>
08	34C	<p><b>Equate Symbol:</b> IXCMGGOXRSNSENDDTIMEEXPIRED</p> <p><b>Meaning:</b> The amount of time that XCF was allowed to suspend the unit of work to synchronously complete accessing caller's storage areas expired. XCF canceled the incomplete send processing and the message was not sent to the target member.</p> <p><b>Action:</b> Increase the SENDTIME value to allow XCF more time to access the storage containing message text.</p>
08	350	<p><b>Equate Symbol:</b> IXCMGGOXRSNPAUSEENVERROR</p> <p><b>Meaning:</b> An XCF message-out request using MSGACCESS=SYNCSUSPEND was unable to suspend because the IXCMGGOX request was issued from a SUSPEND exit routine or from an SRB routine that the system abended with a 47B system completion code.</p> <p><b>Action:</b> Reissue the IXCMGGOX request from an environment other than a SUSPEND exit routine or an SRB routine that the system abended with a 47B system completion code. Alternatively, re-issue the IXCMGGOX request using MSGACCESS=ASYNCR if possible.</p>
08	354	<p><b>Equate Symbol:</b> IXCMGGOXRSNRESOURCEMGRCALLING</p> <p><b>Meaning:</b> An XCF message-out request using MSGACCESS=SYNCSUSPEND can not be issued from an address space resource manager. When invoked from an address space resource manager such as the MASTER address space, not all system functions and required environments associated with MSGACCESS=SYNCSUSPEND are available.</p> <p><b>Action:</b> Correct the address space resource manager to arrange for IXCMGGOX to be issued from another unit of work</p>
08	358	<p><b>Equate Symbol:</b> IXCMGGOXRSNBADFILTERGROUP</p> <p><b>Meaning:</b> The group name specified via the FILTERGROUP keyword is not valid. The group name must be left justified and as needed, padded on the right with blanks to 8 characters. The valid characters are A-Z, 0-9, \$, @, and #.</p> <p><b>Action:</b> Correct the group name provided and retry the request.</p>

Table 13. Return and reason codes for the IXCMGGOX macro (continued)

Hexadecimal return code	Hexadecimal reason code	Equate Symbol meaning and action
08	876	<p><b>Equate Symbol:</b> IXCRSNCODELOCKED</p> <p><b>Meaning:</b> Program Error. IXCMGGOX was issued while the caller was holding a lock.</p> <p><b>Action:</b> Ensure that the macro usage meets the environmental requirements.</p>
08	010Cxxxx	<p><b>Equate Symbol:</b> IXCMGGOXSNPLISTNOPARTINFOBADSTG</p> <p><b>Meaning:</b> Program error. A parameter specifying information on a request with MULTIPART=YES was not valid. The system was unable to store the erroneous information in PART# and ELEMENTPTR fields in the parameter list generated by IXCMGGOX because it could not access the parameter list. The low-order halfword of the reason code (xxxx) contains the reason code for the error that would have been returned if the parameter list had been accessible. (The reason code applies to a X'8' return code.)</p> <p><b>Action:</b> The control parameters must be in the primary address space. Make sure the ALET for the control parameter list is correct.</p>

Table 13. Return and reason codes for the IXCMGGOX macro (continued)

Hexadecimal return code	Hexadecimal reason code	Equate Symbol meaning and action
0C	04	<p><b>Equate Symbol:</b> IXCMGORSNNOBUFFER</p> <p><b>Meaning:</b> Environmental error. The signalling facility is busy; message buffers are temporarily unavailable. Some of the reasons message buffer space might not be available are:</p> <ul style="list-style-type: none"> <li>• There was suddenly a large amount of message buffer space usage, which caused all the buffer space to be temporarily exhausted.</li> <li>• There is a lot of competition for message buffer space. It is possible that the installation should have allocated more message buffers for your particular transport class. The installation can use the message buffer limit to control how much of the total message buffer resource your application can use.</li> <li>• If the sending member specified (or defaulted to) MSGISO=NONE when the IXCJOIN macro was invoked to become an active member of its group, this reason code is also returned for the case where the target member is "message isolated". In effect, the sending system refuses to provide a message buffer for a target member that is "message isolated".</li> </ul> <p>The sending member can specify MSGISO=MSGORSN on the IXCJOIN if there is a need to distinguish the case where there is no buffer for a target member due to message isolation (ixcMsgoRsnTargetIsolated) from the case where XCF has no message buffer due to MAXMSG constraints imposed by the installation (ixcMsgoRsnNoBuffer).</p> <ul style="list-style-type: none"> <li>• Before running or installing your application, you should inform the system programmer of any XCF resources you might require.</li> </ul> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Try sending the message again after a short period of time.</li> <li>• Invoke IXCMGGOX again specifying the TIMEOUT parameter so that XCF will queue the message and resend it automatically when the condition clears. If this condition reoccurs when a nonzero TIMEOUT is specified, XCF will either: <ul style="list-style-type: none"> <li>– Accept the message with return code X'4', or</li> <li>– Reject the message with return code X'C', reason code X'C'</li> </ul> </li> <li>• Inform the system programmer of the space constraints.</li> </ul>

Table 13. Return and reason codes for the IXCMSGOX macro (continued)

Hexadecimal return code	Hexadecimal reason code	Equate Symbol meaning and action
0C	08	<p><b>Equate Symbol:</b> IXCMSGORSNNOPATH</p> <p><b>Meaning:</b> Environmental error. All signalling paths to the target member's system are temporarily unavailable. There could be a problem with the target member's system.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>Try sending the message again after a short period of time. If all paths are still unavailable, something might be wrong with the target member's system. The member can keep trying periodically to send the message until the target member is no longer active (return code 8, reason code 8) or the member group exit routine receives a system-status-update-missing or system-going notification for the target member.</li> </ul> <p>A system-status-update-missing notification indicates that the member's system is stopped for as long as the member's failure detection interval. If a system-status-update-missing notification is received, the member can try again when its group exit routine receives the system status update resumed notification.</p> <p>Once a system-going notification is received, the member's system is terminating. At that point, there is no need to send the message.</p> <ul style="list-style-type: none"> <li>Invoke IXCMSGOX again specifying the TIMEOUT parameter so that XCF will queue the message and resend it automatically when the condition clears. If this condition reoccurs when a nonzero TIMEOUT is specified, XCF will either: <ul style="list-style-type: none"> <li>Accept the message with return code X'4', or</li> <li>Reject the message with return code X'C', reason code X'i8'</li> </ul> </li> </ul> <p>See <i>z/OS MVS Programming: Sysplex Services Guide</i> for more information.</p>

Table 13. Return and reason codes for the IXCMGGOX macro (continued)

Hexadecimal return code	Hexadecimal reason code	Equate Symbol meaning and action
0C	0C	<p><b>Equate Symbol:</b> IXCMGORSNNOMSGSPACE</p> <p><b>Meaning:</b> Environmental error. The message space managed by XCF on behalf of the member has no more capacity. Generally this condition suggests that the volume of message traffic has exceeded processing capacity of the member or its group. Depending on how the member is using the signalling service, member message space might become available as the system continues to process work. The member might use the IXCMGSC service with REQUEST=COMPLETION, TYPE=FORCE to attempt to have the message notify user routine process the message and not save it, thus making more member message space storage available. The member might need to take steps to reduce its message traffic volume.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Retry the request after allowing some time for the condition to clear, or after taking appropriate actions to release member message space.</li> <li>• Try to free up space by discarding saved messages or pending messages.</li> </ul>
0C	10	<p><b>Equate Symbol:</b> IXCMGORSNSYSTEMNOSTORAGE</p> <p><b>Meaning:</b> Environmental error. A shortage of system storage has occurred. The IXCMGGOX request is rejected.</p> <p><b>Action:</b> Retry the request after allowing some time</p> <ul style="list-style-type: none"> <li>• Retry the request after allowing some time for the condition to clear. Subsequent requests might also be rejected until the storage shortage is relieved.</li> <li>• The member might use the IXCMGSC service with REQUEST=COMPLETION, TYPE=FORCE to attempt to have the message notify user routine process the message and not save it, thus making more member message space storage available.</li> <li>• Try to free up space by discarding saved messages or pending messages.</li> </ul>
0C	14	<p><b>Equate Symbol:</b> IXCMGORSNNOBUFFERNOTQUEUED</p> <p><b>Meaning:</b> Environmental error. Signalling delivery has been making no progress delivering messages to the target because XCF message buffers used for signalling are unavailable. The IXCMGGOX request is rejected and is not queued. The system also rejects any further requests that require queueing until the condition is resolved.</p> <p><b>Action:</b> Retry the request after allowing some time for the condition to clear.</p>



Table 13. Return and reason codes for the IXCMGGOX macro (continued)

Hexadecimal return code	Hexadecimal reason code	Equate Symbol meaning and action
0C	18	<p><b>Equate Symbol:</b> IXCMGORSNNOPATHNOTQUEUED</p> <p><b>Meaning:</b> Environmental error. XCF signalling paths are unavailable because of a lack of connectivity to the target system. The system rejects the IXCMGGOX request and does not queue it. The system also rejects subsequent requests until the condition is resolved.</p> <p><b>Action:</b> Retry the request after allowing some time for the condition to clear.</p>
0C	1C	<p><b>Equate Symbol:</b> IXCMGORSNMSGPENDINGMUSTQUEUE</p> <p><b>Meaning:</b> Environmental error. Messages pending; timeout not specified. The system could not initiate the send for this message because there are other message-out requests already pending that must be initiated first.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Try sending the message again after allowing some time for the condition to clear.</li> <li>• Invoke IXCMGGOX again specifying a nonzero TIMEOUT value so that XCF will queue the message and resend it automatically when the condition clears. If the condition reoccurs when a nonzero TIMEOUT value is specified, XCF will either: <ul style="list-style-type: none"> <li>– Accept the message with return code X'4', or</li> <li>– Reject the message with a different reason code with return code X'C'.</li> </ul> </li> </ul>
0C	20	<p><b>Equate Symbol:</b> IXCMGORSNDUALFULL</p> <p><b>Meaning:</b> Environmental error. DU-AL is full. The Dispatchable Unit Access List (DU-AL) is full. Applies only when MSGACCESS=ASYN or MSGACCESS=SYNCSUSPEND is specified.</p> <p><b>Action:</b> Retry the request after removing at least one space from the DU-AL.</p>
0C	24	<p><b>Equate Symbol:</b> IXCMGORSNDUALNOSTORAGE</p> <p><b>Meaning:</b> Environmental error. Unable to obtain DU-AL storage. The system could not obtain storage to create the Dispatchable Unit Access List needed to process the request. Applies only when MSGACCESS=ASYN or MSGACCESS=SYNCSUSPEND is specified.</p> <p><b>Action:</b> Retry the request after allowing some time for the condition to clear.</p>

Table 13. Return and reason codes for the IXCMGGOX macro (continued)

Hexadecimal return code	Hexadecimal reason code	Equate Symbol meaning and action
0C	28	<p><b>Equate Symbol:</b> IXCMGORSNDUALNOTSUITABLE</p> <p><b>Meaning:</b> Environmental error. DU-AL is unsuitable. The Dispatchable Unit Access List (DU-AL) is unsuitable. The DU=AL is too large or provided access to a subspace at some point during its existence. Applies only when MSGACCESS=ASYN or MSGACCESS=SYNCSUSPEND is specified.</p> <p><b>Action:</b> Retry the request under some other work unit.</p>
0C	2C	<p><b>Equate Symbol:</b> IXCMGGOXRSNALLOCPAUSEELEMERROR</p> <p><b>Meaning:</b> The system was unable to obtain resources required to allow XCF to pause the unit of work while accessing storage containing message text. The send request is not attempted. Applies only when SGACCESS=SYNCSUSPEND is specified.</p> <p><b>Action:</b> Retry the request after allowing some time for the condition to clear.</p>
0C	30	<p><b>Equate Symbol:</b> IXCMGGOXRSNFORCECOMPLETION</p> <p><b>Meaning:</b> An IXCMGSC REQUEST=COMPLETION, TYPE=FORCE service call was issued for the message while the message-out unit of work was suspended while XCF synchronously accessed caller's storage areas. XCF canceled the send processing. The message may not have been sent to all target members. Applies only when MSGACCESS=SYNCSUSPEND is specified.</p> <p><b>Action:</b> Perform application specific retry, recovery or error processing to handle the forced completion of the send request.</p>
0C	34	<p><b>Equate Symbol:</b> IXCMGGOXRSNRELEASEMSG</p> <p><b>Meaning:</b> An IXCMGSC REQUEST=RELEASEMSG service call was issued for the message while the message-out unit of work was paused while XCF synchronously accessed storage areas of the caller. XCF canceled the send processing. The message might not have been sent to all target members. Applies only when MSGACCESS= SYNCSUSPEND is specified.</p> <p><b>Action:</b> Perform application specific retry, recovery or error processing to determine if the send request needs to be retried.</p>
0C	36	<p><b>Equate Symbol:</b> IXCMGGOXRSNDISCARDMSG</p> <p><b>Meaning:</b> An IXCMGSC REQUEST=DISCARDMSG service call was issued for the message while the message-out unit of work was paused while XCF synchronously accessed storage areas of the caller. XCF canceled the send processing. The message might not have been sent to all target members. Applies only when MSGACCESS= SYNCSUSPEND is specified.</p> <p><b>Action:</b> Perform application specific retry, recovery or error processing to handle the forced completion of the send request.</p>

Table 13. Return and reason codes for the IXCMSGOX macro (continued)

Hexadecimal return code	Hexadecimal reason code	Equate Symbol meaning and action
0C	38	<b>Equate Symbol:</b> IXCMSGOXRSNASYNCSYNCSUSPENDABEND <b>Meaning:</b> An XCF message-out request using MSGACCESS=SYNCSUSPEND was cancelled due to an asynchronous abend in the system that affected the suspended unit of work. XCF could not complete sending the message to all targets. Applies only when MSGACCESS=SYNCSUSPEND is specified. <b>Action:</b> Perform application specific retry, recovery or error processing to determine if the send request needs to be retried.
0C	3C	<b>Equate Symbol:</b> ixcMsgRsnTargetIsolated <b>Meaning:</b> The target member is not processing messages in a timely manner and is currently "message isolated". Messages targeted to such a member will either be rejected or delayed by the sending system. If and when the target member makes sufficient progress, normal message flow will resume. This reason code only applies if the sending member specified MSGISO=MSGORSN when the IXCJOIN macro was invoked to become an active group member. If MSGISO=NONE is specified (or taken as the default), a request to send a message to target member that is "message isolated" is rejected with a "no buffer" reason code (ixcMgoRsnNoBuffer). <b>Action:</b> Retry the request after allowing time for the condition to clear. If TIMEOUT was not specified, consider retrying the request with a nonzero TIMEOUT value to have XCF try to handle the condition. If the target member becomes not isolated before the TIMEOUT expires, XCF will send the message.
10	None.	<b>Meaning:</b> System error. XCF processing failed. <b>Action:</b> Retry the request one or more times. If the problem persists, record the return and reason code and supply it to the appropriate IBM support personnel.

## Example

**Operation:** Send a message to another active member of the XCF group. You can obtain the token of the calling member and the target member from the member information field that IXCQUERY returns. XCF is to store the return code and reason code into the variables RETURN and REASON. The code is as follows:

```

IXCMSGOX MEMTOKEN=TOKEN1,MSGBUF=BUFFER1,MSGLEN=LENMSG,      X
          TARGET=TARGET2,RETCODE=RETURN,RSNCODE=REASON,MF=S

TOKEN1  DS  CL8          MEMBER TOKEN OF MEMBER SENDING X
                        THE MESSAGE
TOKEN2  DS  CL8          MEMBER TOKEN OF MEMBER TO      X
                        RECEIVE THE MESSAGE
RETURN  DS  1F           RETURN CODE
REASON  DS  1F           REASON CODE
BUFFER1 DC  CL256'THIS IS A TEST' MESSAGE TO BE SENT
LENMSG  DC  F'256'       LENGTH OF THE MESSAGE

```

You can obtain the member tokens from the QUAMTKN field in the area returned by IXCJOIN or IXCQUERY,



## Chapter 17. IXCNOTE — XCF Note Pad Interface

### Description

The XCF Note Pad macro (IXCNOTE) encapsulates the interfaces that allow an application to interact with the XCF Note Pad Services. With these interfaces, an application can create or delete a *note pad*. A note pad contains zero or more notes. A *note* contains 1024 bytes of application data and is identified by an application provided *name*. An application that is *connected* to a note pad can create, read, modify, or delete notes in the note pad. Each note has an *instance number* that is assigned by XCF whenever the note is created or modified. The instance number uniquely identifies an instance of a particular note.

Depending on the various attributes specified when the note pad is created, you can use a note pad for any of the following purposes:

- Store data that allows a backup instance of an application to resume processing on some other system in the sysplex after the primary instance fails.
- Share data between systems in the sysplex. You can use the instance number of a note to provide a simple "compare and swap" type serialization to ensure the integrity of updates to the shared notes.

To begin using a note pad, you must first create it. Once created, the note pad will persist until one of the following situations occurs:

- The exploiter explicitly deletes the note pad by using an IXCNOTE REQUEST=NOTEPAD REQTYPE=DELETE request.
- The note pad fails.
- The note pad is manually deleted through use of the note pad deletion utility IXCDELNP.

The creator of the note pad determines its attributes. For example, the creator specifies the number of notes that the note pad must hold and indicates whether the note pad can receive updates from multiple connections.

After a note pad is created, a program must *connect* to the note pad to use it. One or more systems in the sysplex can access a note pad. On any given system, each address space that uses the note pad might need its own connection. In other cases, a single connection from one particular address space might suffice. The design of the exploiting application determines what is needed.

A connection has either *task scope* or *address space scope*. In either case, the *connector address space* is the address space that is home when the connection is created.

#### Task scope

If the task that creates the connection has its own security environment, meaning that TCBSENV is nonzero, the connection has task scope. For a connection with task scope, the *connector* is the task itself.

#### Address space scope

If the connection is created by an SRB or by a task that does not have its own security environment, meaning that TCBSENV is zero, the connection has address space scope. For a connection with address space scope, the *connector* is any work unit whose home address space is the connector address space, and for which the work unit is either an SRB or a task that does not have its own security environment. Thus, an address space connection can have multiple work units serve as the connector.

A *valid user* is a work unit that is allowed to use a connection to a note pad. Each IXCNOTE request that requires a connection indicates the conditions under which the requester is deemed a valid user. The creator of the connection specifies the USAGE keyword to indicate the criteria that determine a valid user when accessing notes in the note pad. Notes in a note pad are accessed with an IXCNOTE invocation that specifies REQUEST=NOTE or REQUEST=NOTES.

After a connection is created, the SAF profile of the connector might change. XCF is not typically aware of such changes since SAF checking is not typically performed for a connection that is used by a work unit that is deemed a valid user. To ensure that changes to the SAF profile are observed by XCF and applied to the connector, you must delete the connection and recreate it. In some cases, you might need to bring down the connector address space in order for the changes to become effective.

A note pad resides in a coupling facility. In general, if a coupling facility that contains a note pad fails, the note pad also fails. XCF automatically deletes a failed note pad and all of its connections throughout the sysplex. As needed, the application is responsible for recovering the note pad by recreating it, reestablishing the necessary connections to the new note pad instance, and recreating the notes.

Under normal circumstances, a request to access a note in a note pad is processed as a synchronous coupling facility operation. As needed, the invoking work unit is suspended until the request completes. However, if conditions are such that the duration of the suspension might be significantly longer than normal, the note request is rejected. In such cases, XCF returns with a return and reason code that indicates the situation (IXCNOTERSNQUIESCED). Requests are rejected in this manner when, for example, the coupling facility structure that contains the note pad is quiesced for rebuild processing.

**Note:** A structure that is engaged in rebuild processing might quiesce for several seconds, perhaps even several minutes.

The connector can issue IXCNOTE REQUEST=CONNECTION REQTYPE=PAUSE to wait for the quiesce conditions to clear. The service routine that processes the PAUSE request might also return if the quiesce conditions change. If the note pad becomes accessible again, the connector try again to use IXCNOTE to manipulate notes in the note pad. If the note pad is still inaccessible, the requester can inspect the quiesce conditions to determine whether to reissue the PAUSE request to continue waiting, or to take some other appropriate recovery action such as disconnecting from or even deleting the note pad. As an alternative to using the PAUSE request, the connector can simply try the request again after allowing some time for the condition to clear (poll).

The IXCNOTE macro completes a parameter list with caller provided data and generates a stacking, space-switch program call to an XCF service routine.

**Note:** The calling work unit might suspend until the request completes.

## Environment

---

The following contains the requirements for the caller:

**Category****Description****Minimum authorization:**

Supervisor state or problem state

Any PSW-key mask (PKM)

Programs require appropriate SAF (System Authorization Facility) authorization to the FACILITY class resource IXCNOTE.*owner.application* when creating, deleting, querying, or modifying a note pad, and when creating a connection to a note pad. The *owner* and *application* are derived from the note pad name.

- To create or delete a note pad or modify note pad attributes, the program must have CONTROL access.
- To query a note pad, the program must have READ access.
- To create a connection to a note pad that can be used to create, read, write, replace, or delete notes, the program must have UPDATE access.
- To create a connection to a note pad that can only read notes, the program must have READ access.

A program that uses a connection must be recognized by XCF as a valid user of the connection. Each applicable request documents the conditions under which a program is deemed to be a valid user. If the program is not recognized as a valid user, XCF might perform a SAF check to verify that the user has the authority appropriate for the request.

Note that all SAF checking is performed against the security environment of the work unit. For a task that has a task specific security environment, the Access Control Environment Element (ACEE) anchored at offset TCBSENV in the TCB (macro IKJTCTB) is used. For an SRB, or a task for which TCBSENV is zero, the ACEE anchored at offset ASXBSENV in the ASXB (macro IHAASXB) for the home address space is used.

If SAF is not available, or if there is no IXCNOTE.*owner.application* resource defined for the note pad in the FACILITY class, a request to create, delete, query, or connect to a note pad is rejected if the program is running in problem state with a PKM allowing key 8-15 (IXCNOTERSNNOSECPROFILE). Such requests are accepted if the program is running in supervisor state or with a PKM allowing key 0-7. However, note that the program will need to run in supervisor state or with a PKM allowing key 0-7 when using a connection that was created under these circumstances.

Any program that specifies USAGE=SERVER or USAGE=CLIENT when creating a connection to a note pad must be running in supervisor state or with a PKM allowing key 0-7.

**Dispatchable unit mode:**

Task or SRB mode

Category	Description
<b>Cross memory mode:</b>	<p>Any PASN, any HASN, any SASN</p> <p>Problem state programs with a PKM allowing key 8-15 must be running task mode with home and primary being the same address space for the following requests:</p> <ul style="list-style-type: none"> <li>• REQUEST=NOTEPAD REQTYPE=CREATE</li> <li>• REQUEST=NOTEPAD REQTYPE=QUERY</li> <li>• REQUEST=NOTEPAD REQTYPE=DELETE</li> <li>• REQUEST=NOTEPAD REQTYPE=MODIFY</li> <li>• REQUEST=CONNECTION REQTYPE=CREATE</li> </ul> <p>Any program that specifies USAGE=SERVER or USAGE=CLIENT when creating a connection to a note pad must be running with home and primary being the same address space.</p>
<b>AMODE:</b>	<p>31-bit or 64-bit</p> <p>If in 64-bit mode, specify SYSSTATE AMODE64=YES before invoking this macro.</p>
<b>ASC mode:</b>	<p>Primary or access register (AR)</p> <p>If in Access Register ASC mode, specify SYSSTATE ASCENV=AR before invoking this macro.</p>
<b>Interrupt status:</b>	Enabled for I/O and external interrupts
<b>Locks:</b>	No locks held
<b>Control parameters:</b>	<p>Control parameters must be in the primary address space or, for AR-mode callers, must be in an address/data space that is addressable through a public entry on the caller's Dispatchable Unit Access List (DU-AL). The control parameters can also reside in a common area data space.</p> <p>The control parameters must be accessible using the PSW key of the program making the request. The service routine will always fetch the control parameters. It might also store into them.</p>

## Programming Requirements

---

The IXCYNOTE macro provides the format of the area to which the ANSAREA parameter points. Include the IXCYNOTE macro in your program.

## Restrictions

---

- Task mode callers cannot have an enabled, unlocked task (EUT) FRR set.
  - IXCNOTE cannot be used by tasks higher in the task tree than the cross memory resource owning task (the top, or first, job step task in the address space).
  - Address space and task resource managers are not permitted to issue IXCNOTE requests that process note pads or connections (REQUEST=NOTEPAD or REQUEST=CONNECTION).
- Note:** XCF will automatically delete any connection associated with a terminated address space or task.
- Callers running in SRB mode should refrain from invoking the IXCNOTE service under the following circumstances:
    - After the SRB receives a X'47B'abend



- When running in a suspend exit after invoking SUSPEND

In these cases, the IXCNOTE request can be rejected because the service routine will not be able to suspend the caller as needed (IXCNOTERSNBADSUSPENDENV).

- All storage areas provided to the IXCNOTE service must be accessible via the PKM of the caller. In particular, this restriction applies to the input control parameters as well as storage areas identified by the keywords ANSAREA, BUFFER, CRITERIA, DESCRIPTION, and INFO.
- When using REQUEST=NOTEPAD, REQTYPE=MODIFY to modify the number of notes (#NOTES) that a note pad is allowed to hold, IBM recommends that if connections to the note pad exist, then all connections to the note pad be from systems that support REQUEST=NOTEPAD, REQTYPE=MODIFY. To determine whether the support for REQUEST=NOTEPAD, REQTYPE=MODIFY is available on the system from which you are connecting, issue IXCQUERY REQINFO=FEATURES. QuReqRfIxcNoteResiliency indicates whether the support is available.

If a connection to the note pad resides on a system that does not support REQUEST=NOTEPAD, REQTYPE=MODIFY (a "down level" system), an IXCNOTE request issued to create, read, modify, or delete notes in the note pad from the down level system while a note pad #NOTES is being modified may render the note pad unusable from the down level system for the life of the IPL. Restoring access to the note pad from the down level system will require a re-IPL of the down level system or deleting and re-creating the note pad

- For REQUEST=NOTEPAD, REQTYPE=CREATE, LOSSCONNDELETE=YES, IBM recommends that if connections to the note pad exist, then all connections to the note pad be from systems that support LOSSCONNDELETE=YES.

If a connection to the note pad resides on a system that does not support LOSSCONNDELETE=YES (a "down level" system), an IXCNOTE request issued to create, read, modify, or delete notes in the note pad from the down level system may not be properly fenced when the note pad is automatically being deleted by LOSSCONNDELETE processing.

If a "down level" system exists in the sysplex, it is possible that the LOSSCONNDELETE=YES specification will not be honored. This can occur if an "up level" system needs help from another system in the sysplex to initiate the LOSSCONNDELETE processing and the available helping system(s) are "down level". An "up level" system may need help initiating the LOSSCONNDELETE processing if the system does not have access to the note pad catalog structure

## Input Register Information

---

Before issuing the IXCNOTE macro, the caller does not have to place any information into any register unless using it in register notation for a particular parameter, or using it as a base register.

Input general purpose registers (GPRs):

### Register Contents

- 0-1**  
Reserved
- 2-13**  
Undefined
- 14-15**  
Reserved

Input access registers (ARs):

### Register Contents

- 0-1**  
Reserved
- 2-13**  
Undefined

**14-15**

Reserved

## Output Register Information

---

When control returns to the caller, the general purpose registers (GPRs) contain:

**Register****Contents****0**

Reason code, if applicable, and if GPR15 return code is nonzero

**1**

Used as a work register by the system

**2-13**

Unchanged

**14**

Used as a work register by the system

**15**

Return code

When control returns to the caller, the access registers (ARs) contain:

**Register****Contents****0-1**

Used as work registers by the system

**2-13**

Unchanged

**14-15**

Used as work registers by the system

## Performance Implications

---

None.

## Understanding IXCNOTE Version Support

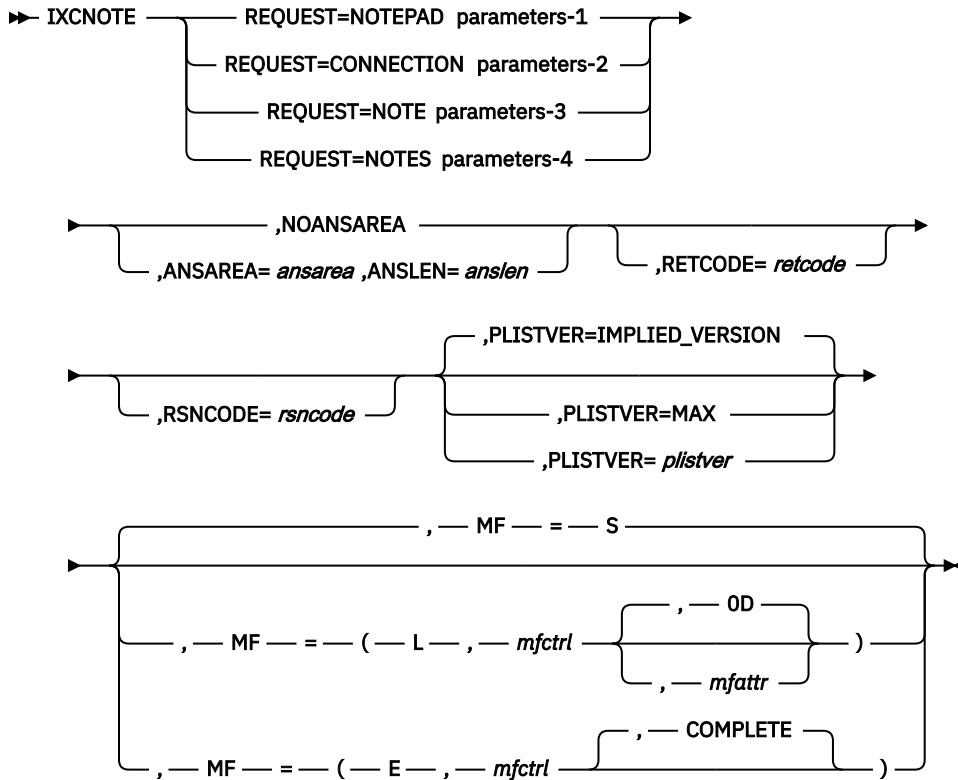
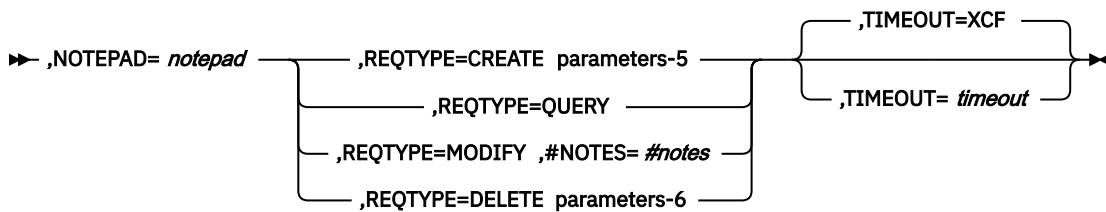
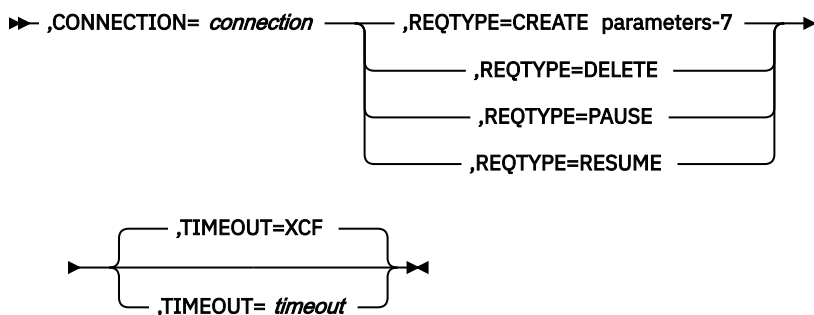
---

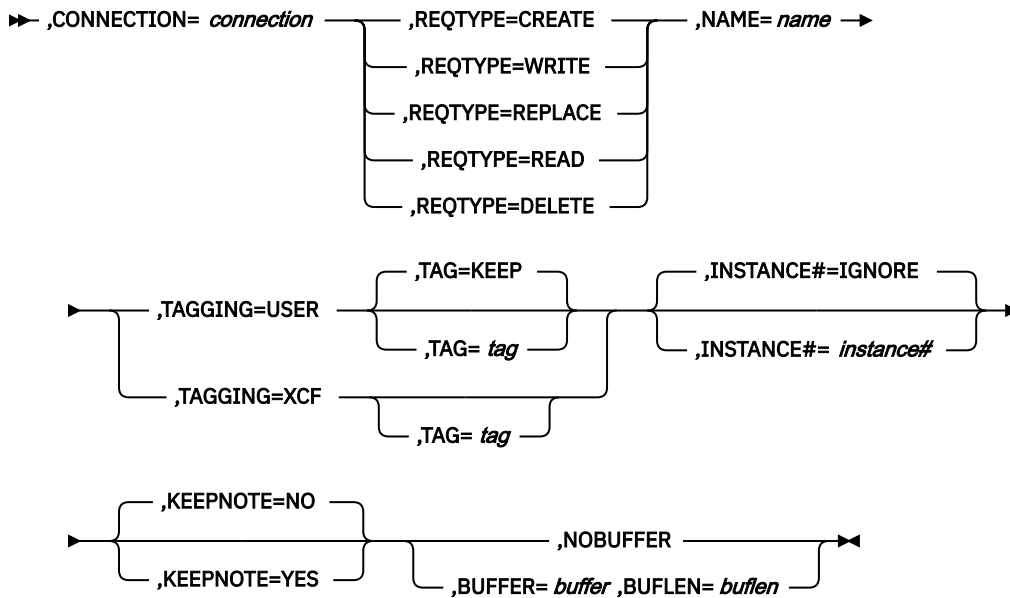
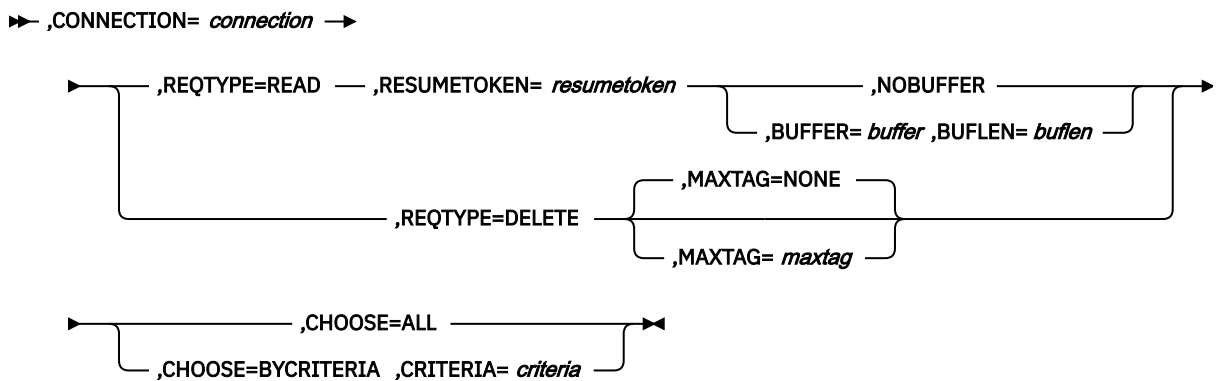
The IXCNOTE macro supports version 0. See [Chapter 2, “Specifying a Macro Version Number,” on page 5](#) for considerations when specifying the version of the parameter list with PLISTVER.

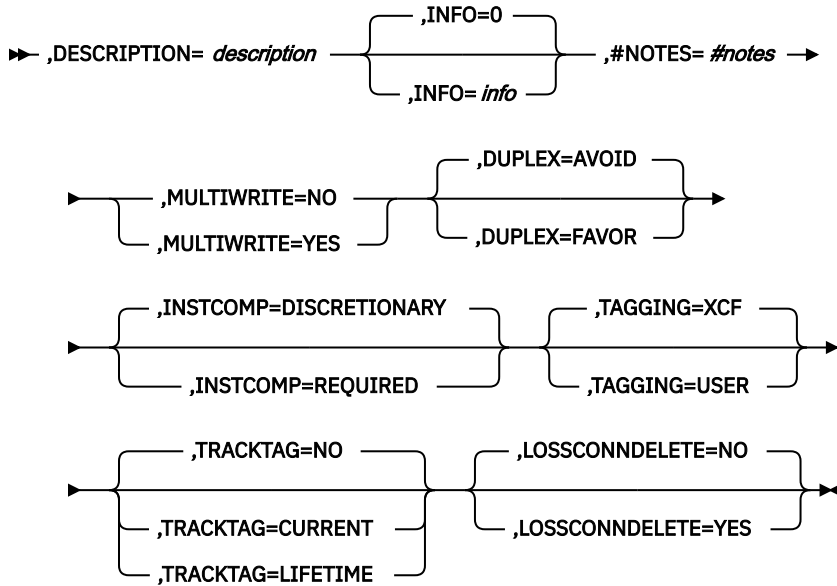
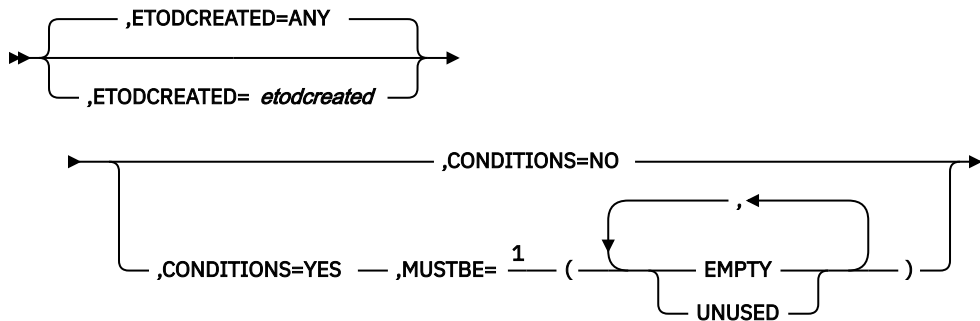
## Syntax

---

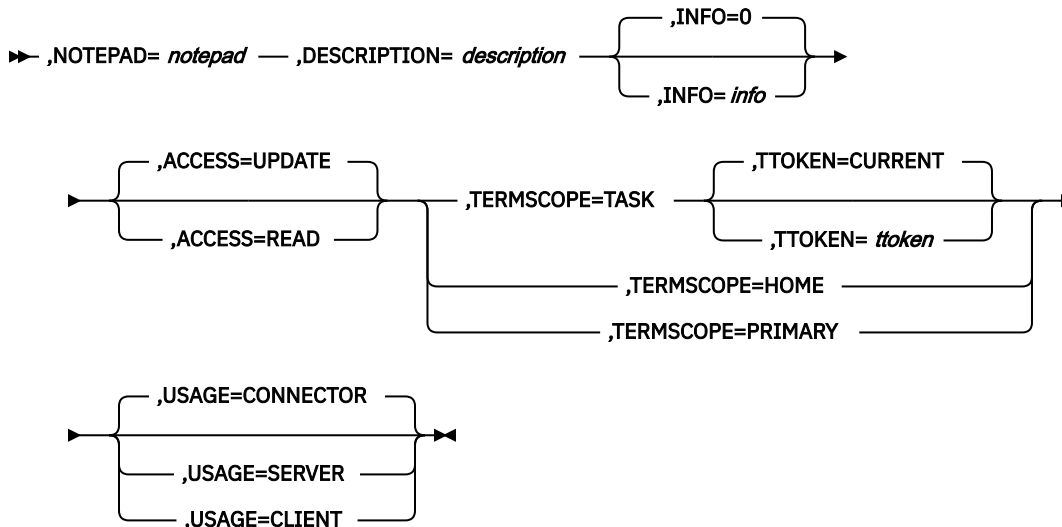
The IXCNOTE macro is written as follows:

**main diagram****parameters-1****parameters-2**

**parameters-3****parameters-4**

**parameters-5****parameters-6****Notes:**

<sup>1</sup> If only one parameter is selected, the surrounding parentheses are optional. Each option can only be specified once.

**parameters-7**

## Parameters

The parameters that are common to all requests follow in alphabetical order. The additional parameters specific to each request are described in [“Parameters for REQUEST=NOTEPAD”](#) on page 241, [“Parameters for REQUEST=CONNECTION”](#) on page 246, [“Parameters for REQUEST=NOTE”](#) on page 252, and [“Parameters for REQUEST=NOTES”](#) on page 256.

### Parameters Common to All Requests

**,ANSAREA=ansarea**

**,NOANSAREA**

Use this required input parameter to specify how XCF is to handle the results of the request.

**,ANSAREA=ansarea**

One of a set of mutually exclusive keywords indicating a storage area into which XCF is to store the results of the request.

If the ANSAREA is not large enough to hold the answer area header (mapped by IXCYNOTE\_TANSAREA), the request is rejected outright (IXCNOTERSNANSLENBADVAL). If the ANSAREA is large enough for the header, but not large enough to hold all the requested output data, the service routine returns with a return and reason code indicating that the ANSAREA needs to be bigger. In cases where the answer areas is not large enough for requests that return a static number of records, the request is rejected with reason code IXCNOTERSNANSLENMORE. In cases where the answer area is not large enough for requests that return a variable number of records, the answer area is filled with as many records as will fit and the service routine returns with a warning return code and a reason indicating that there is additional data available (IXCNOTERSNMOREDATA or IXCNOTERSNMORENOTES). For reason IXCNOTERSNMORENOTES, the caller can reissue the request with the resultant RESUMETOKEN value to continue reading the remaining notes. For the other reasons, the answer area header indicates how much storage is needed to contain the requested data (AA\_ANSAREASIZENEDED). As appropriate, the caller should obtain a sufficiently large answer area and reissue the request.

The storage containing the ANSAREA must reside in the primary address space of the caller, or in a space addressable via a public entry on the Dispatchable Unit Access List (DU-AL), or in a common area data space. The storage must be accessible using the PSW key of the program making the request.

Note that the answer area storage can be used by XCF as a work area. In particular, some or all of the answer area might be updated regardless of whether the request completes successfully or not. The answer area will be valid for use for any return and reason code that indicates that an answer area is stored.

**To code:** Specify the RS-type address, or address in register (2)-(12), of a character field.

#### **,NOANSAREA**

One of a set of mutually exclusive keywords indicating that no answer area is being provided. XCF will process the request and provide a suitable return and reason code, but an answer area will not be stored.

This specification is not permitted for REQUEST=NOTES REQTYPE=READ since an answer area is a required input for that request.

#### **,ANSLEN=*anslen***

When ANSAREA=*ansarea* is specified, use this required input parameter to specify a variable containing the size in bytes of the answer area provided.

**To code:** Specify the RS-type address, or address in register (2)-(12), of a fullword field, or specify a literal decimal value.

#### **,MF=S**

#### **,MF=(L,*mfctrl*)**

#### **,MF=(L,*mfctrl*,*mfattr*)**

#### **,MF=(L,*mfctrl*,0D)**

#### **,MF=(E,*mfctrl*)**

#### **,MF=(E,*mfctrl*,COMPLETE)**

Use this optional input parameter to specify the macro form.

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service. MF=S is the default.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter might be coded with the list form of the macro.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form of the macro stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

#### **,*mfctrl***

The name of a storage area to contain the parameters. For MF=S and MF=E, this can be an RS-type address or an address in register (1)-(12).

#### **,*mfattr***

An optional 1 to 60 character input string that you use to force boundary alignment of the parameter list. Use a value of 0F to force the parameter list to a word boundary, or 0D to force the parameter list to a doubleword boundary. If you do not code *mfattr*, the system provides a value of 0D.

#### **,COMPLETE**

Specifies that the system is to check for required parameters and supply defaults for omitted optional parameters.

#### **,PLISTVER=IMPLIED\_VERSION**

#### **,PLISTVER=MAX**

#### **,PLISTVER=*plistver***

Use this optional input parameter to specify the version of the macro. See [“Understanding IXCNOTE Version Support”](#) on page 234 for a description of the options available with PLISTVER.

**REQUEST=NOTEPAD**  
**REQUEST=CONNECTION**  
**REQUEST=NOTE**  
**REQUEST=NOTES**

Use this required parameter to specify the request to be processed.

**REQUEST=NOTEPAD**

Process a note pad. For parameter descriptions applicable to REQUEST=NOTEPAD, see [“Parameters for REQUEST=NOTEPAD” on page 241.](#)

**REQUEST=CONNECTION**

Process a connection to a note pad. For parameter descriptions applicable to REQUEST=CONNECTION, see [“Parameters for REQUEST=CONNECTION” on page 246.](#)

**REQUEST=NOTE**

Process a single note in the note pad associated with the indicated connection.

The keyword NAME identifies the note to be manipulated. For create, write, or replace requests, the note content will be fetched from the storage area identified by the BUFFER keyword. For read and delete requests, BUFFER indicates where the content of the note is to be stored.

A NOTE request can be issued by a work unit that can satisfy any of the following conditions:

- Requester is the connector
- Home is connector address space and has appropriate SAF authorization
- Supervisor state or PKM allowing key 0-7, and:
  - Connection created with USAGE=CLIENT, or
  - Connection created with USAGE=SERVER and primary is the connector space, or
  - Running as an address space resource manager

For parameter descriptions applicable to REQUEST=NOTE, see [“Parameters for REQUEST=NOTE” on page 252.](#)

**REQUEST=NOTES**

Process a collection of notes in the note pad.

The CHOOSE keyword determines the set of notes to be processed. One can either read or delete the selected notes.

Requests to process multiple notes could be long running.

A NOTES request can be issued by a work unit that can satisfy any of the following conditions:

- Requester is the connector
- Home is connector address space and has appropriate SAF authorization
- Supervisor state or PKM allowing key 0-7, and:
  - Connection created with USAGE=CLIENT, or
  - Connection created with USAGE=SERVER and primary is the connector space, or
  - Running as an address space resource manager

For parameter descriptions applicable to REQUEST=NOTES, see [“Parameters for REQUEST=NOTES” on page 256.](#)

**,RETCODE=retcode**

Use this optional output parameter to specify the location into which the return code is to be copied from GPR 15. If you specify 15, GPR15, REG15, or R15 (within or without parentheses), the value will be left in GPR 15.

**To code:** Specify the RS-type address of a fullword field, or register (2)-(12) or (15), (GPR15), (REG15), or (R15).



**,RSNCODE=rsncode**

Use this optional output parameter to specify the location into which the reason code is to be copied from GPR 0. If you specify 0, 00, GPRO, GPRO0, REG0, REG00, or R0 (within or without parentheses), the value will be left in GPR 0.

**To code:** Specify the RS-type address of a fullword field, or register (0) or (2)-(12), (00), (GPRO), (GPRO0), REG0), (REG00), or (R0).

## Parameters for REQUEST=NOTEPAD

**,#NOTES=#notes**

When REQTYPE=CREATE is specified, use this required input parameter to specify a variable containing the number of notes that the note pad needs to hold. The specified value must be at least one.

After the note pad is created, there can be times when XCF is unable to provide the requested number of notes. For example, the coupling facility structure containing the note pad could be full. In such cases, a request to process a note could be rejected with a return and reason code indicating that the note pad resources are constrained (IXCNOTERSNCONSTRAINED). If the note pad is not constrained, an attempt to create more than the requested number of notes is rejected with an indication that the note pad is full (IXCNOTERSN#NOTESEXCEEDED).

If the note pad is full, it might be necessary to delete one or more notes in order to successfully create a new note or update an existing note.

When REQTYPE=MODIFY is specified, use this required input parameter to change the maximum number of notes that a note pad is allowed to hold. The specified value may be greater than (increase) or less than (decrease) the current #NOTES value for the note pad. The specified value must be at least one (1).

**To code:** Specify the RS-type address, or address in register (2)-(12), of a fullword field, or specify a literal decimal value.

**,CONDITIONS=NO****,CONDITIONS=YES**

When REQTYPE=DELETE is specified, use this required parameter to indicate whether there are any conditions that must be satisfied in order to delete the note pad.

**,CONDITIONS=NO**

The delete of the note pad is to be performed unconditionally. The note pad is to be deleted even if it contains notes. The note pad is to be deleted even if it has connections.

**,CONDITIONS=YES**

The delete of the note pad is to proceed only if the conditions specified by the MUSTBE keyword are satisfied.

**,DESCRIPTION=description**

When REQTYPE=CREATE is specified, use this required input parameter to specify a variable containing a description of the note pad. The string can contain any alphanumeric (A-Z, a-z, 0-9), national (@, #, \$), or special (underscore or blank) character. Leading blanks and all blank descriptors are not permitted. Descriptions are case sensitive. The description will appear in various XCF messages and diagnostic data reports. The description is intended to help installations and service personnel understand the function, purpose, or role of the note pad.

The storage containing the DESCRIPTION must reside in the primary address space of the caller, or in a space addressable via a public entry on the Dispatchable Unit Access List (DU-AL), or in a common area data space. The storage must be accessible using the PSW key of the program making the request.

**To code:** Specify the RS-type address, or address in register (2)-(12), of a 32 character field.

**,DUPLEX=AVOID****,DUPLEX=FAVOR**

When REQTYPE=CREATE is specified, use this optional parameter to indicate a preference as to whether the note pad should be placed in a duplexed coupling facility structure. There is no guarantee that the specified preference can be satisfied. If the preference is satisfied initially, there is no guarantee that the preference will remain in effect for the life of the note pad.

A duplexed structure will generally provide greater availability since it is generally more resilient to failure than a simplex structure. A simplex structure will generally provide faster request response times than a duplexed structure since the note operations will not incur the overhead needed to replicate the request. The requirements of the exploiting application determine which option is to be preferred. The default is DUPLEX=AVOID.

**,DUPLEX=AVOID**

XCF should try to avoid placing the note pad in a duplexed structure.

**,DUPLEX=FAVOR**

XCF should try to place the note pad in a duplexed structure.

**,ETODCREATED=etodcreated****,ETODCREATED=ANY**

When REQTYPE=DELETE is specified, use this optional input parameter to specify a variable containing the extended time of day timestamp (ETOD) when the note pad was created. This timestamp identifies a unique instance of the note pad. Specify this keyword to ensure that the intended instance of the note pad is deleted. If not specified, or if the value of the indicated variable is zero, the current instance of the note pad will be processed.

The note pad data record contains the relevant ETOD (NPD\_ETODWHENCREATED). A note pad data record (IXCYNODE\_TNOTEPADDATA) is stored in the answer area when the note pad is created or queried. The connect data record also contains the relevant ETOD (CD\_ETODWHENNPCREATED). A connect data record (IXCYNODE\_TCONNECTDATA) is stored in the answer area when a connection to the note pad is created. The default is ANY.

**To code:** Specify the RS-type address, or address in register (2)-(12), of a 16-character field.

**,INFO=info****,INFO=0**

When REQTYPE=CREATE is specified, use this optional input parameter to specify a variable which contains information about the note pad. The content and interpretation of this information is determined by the creator of the note pad. XCF associates a copy of the indicated data with the note pad. A copy of the data is visible to other processes in the sysplex via queries that return information about the note pad. The creator might, for example, use INFO to define the protocols that users of the note pad are to follow. Alternatively, INFO could be used to describe the attributes of the note pad.

The storage containing the INFO must reside in the primary address space of the caller, or in a space addressable via a public entry on the Dispatchable Unit Access List (DU-AL), or in a common area data space. The storage must be accessible using the PSW key of the program making the request.

The default is 0.

**To code:** Specify the RS-type address, or address in register (2)-(12), of a 64-character field.

**,INSTCOMP=DISCRETIONARY****,INSTCOMP=REQUIRED**

When REQTYPE=CREATE is specified, use this optional parameter to indicate whether the users of the note pad are required to perform instance number comparisons when replacing or deleting a note (see [INSTANCE#](#) keyword for [REQUEST=NOTE](#)). The default is INSTCOMP=DISCRETIONARY.

**,INSTCOMP=DISCRETIONARY**

Instance number comparisons are not required.

When a note pad connector calls IXCNOTE REQUEST=NOTE to replace, write, or delete an existing note, any [INSTANCE#](#) specification is permitted. Instance number comparisons will be performed only if the connector so requests.

**,INSTCOMP=REQUIRED**

Instance number comparisons are required.

When a note pad connector calls IXCNOTE REQUEST=NOTE to replace, write, or delete an existing note, an instance number comparison must be performed to ensure that the correct instance of the note is being manipulated. For REQTYPE=REPLACE and REQTYPE=DELETE, the request is immediately rejected unless a nonzero INSTANCE# value is specified. For REQTYPE=WRITE, the request proceeds. If the note does not exist, the write request will create a new note (if the request is otherwise valid). If the note already exists, the write request is rejected.

**,LOSSCONNDELETE=NO****,LOSSCONNDELETE=YES**

When REQTYPE=CREATE is specified, use this optional parameter to indicate whether XCF can automatically initiate note pad delete processing when all systems with connectors to the note pad have lost connectivity to the coupling facility hosting the note pad. The default is NO.

**,LOSSCONNDELETE=NO**

XCF will not automatically initiate note pad delete processing when all systems with connectors to the note pad have lost connectivity to the coupling facility hosting the note pad.

**,LOSSCONNDELETE=YES**

XCF will automatically initiate note pad delete processing when all systems with connectors to the note pad have lost connectivity to the coupling facility hosting the note pad.

As a result of XCF automatically initiating delete processing the following will occur:

- Any existing notes will be deleted as part of deleting the note pad.
- Any existing connections will be deleted as part of deleting the note pad. Depending on the timing as to when the deletion is recognized, connections that are deleted in this manner might have their IXCNOTE requests rejected with a variety of reason codes (including for example, IXCNOTERSNNOTEPADNOTEXIST, IXCNOTERSNCONNECTIONNOTEXIST, and IXCNOTERSNNOTENOTEXIST).
- Any existing paused connections would be resumed and then deleted as part of deleting the note pad. If an answer area was provided on the PAUSE request, dr\_ResumeCode=ixcynote\_kResumeBeingDeleted would be set in the data area mapped by ixcynote\_tDetailsResumed.

**,MULTIWRITE=NO****,MULTIWRITE=YES**

When REQTYPE=CREATE is specified, use this required parameter to indicate whether the note pad can be updated by more than one connector at a time.

**,MULTIWRITE=NO**

The note pad is to have at most one connector with write access (ACCESS=UPDATE) connected to the note pad at any one time.

**,MULTIWRITE=YES**

The note pad can have two or more connectors with write access (ACCESS=UPDATE) connected to the note pad at the same time.

**,MUSTBE=EMPTY****,MUSTBE=UNUSED**

When CONDITIONS=YES and REQTYPE=DELETE are specified, use this required parameter to indicate the conditions that must be true in order for the note pad to be deleted.

One or more conditions can be specified. Only the specified conditions are considered when determining whether to delete the note pad. For example: Specifying MUSTBE=EMPTY means the delete request is rejected if the note pad contains notes, but the existence of connections is irrelevant. Specifying MUSTBE=UNUSED means the delete request is rejected if the note pad has connections, but the existence of notes is irrelevant. Specifying MUSTBE=(EMPTY,UNUSED) means the delete request is rejected if the note pad has notes or connections or both.

If there are competing delete requests with different conditions, or perhaps no conditions at all, the delete will be processed per the request with the least restrictive conditions.

**,MUSTBE=EMPTY**

The note pad must be empty. If the note pad contains notes, the delete request will be rejected (IXCNOTERSNNOTEPADINUSE).

**,MUSTBE=UNUSED**

The note pad must not have any users. If the note pad has a connection, the delete request will be rejected (IXCNOTERSNNOTEPADINUSE).

One or more values can be specified for the MUSTBE parameter. If more than one value is specified, group the values within parentheses.

**,NOTEPAD=notepad**

Use this required input parameter to specify a variable containing the name of the note pad to be processed.

Note pad names are mapped by IXCYNOTE\_TNOTEPADNAME. Note pad names consist of four 8 byte sections. The various sections identify the owner of the note pad, the application using the note pad, and its function. Each 8 byte section must be left justified, padded on the right with EBCDIC blanks as needed. Each section can contain any uppercase alphabetic (A-Z), numeric (0-9), national (@, #, \$), or underscore (\_) character. The first two sections, which identify the owner and application, must not be all blanks. The remaining sections can be all blank.

To avoid names used by IBM, do not begin note pad names (*owner*) with the letters A through I or the character string SYS. Names beginning with the string SYSXCF are reserved for use by XCF.

Note that if the installation so chooses, it can define coupling facility structure names of the form IXCNP\_ownership (where xx is the EBCDIC representation of a two digit hexadecimal number) to have XCF allocate note pads for the indicated *owner* in one of the designated structures. In the absence of any such definition, the note pad will be allocated in a default note pad structure defined for XCF.

**To code:** Specify the RS-type address, or address in register (2)-(12), of a 32 character field.

**,REQTYPE=CREATE**

**,REQTYPE=QUERY**

**,REQTYPE=DELETE**

**,REQTYPE=MODIFY**

Use this required parameter to indicate the type of request to be processed against the note pad.

**,REQTYPE=CREATE**

XCF is to create a new note pad.

**,REQTYPE=QUERY**

Get information about the note pad.

If an answer area (ANSAREA) is provided, one can for example, obtain the maximum number of notes supported by the note pad, the current number of notes in the note pad, or the maximum note tag value, as well as the INFO that was specified by the creator of the note pad. One can also obtain information about the systems that have connections to the note pad.

If an answer area is provided, it must be large enough to contain at least one note pad data record (IXCYNOTE\_TNOTEPADDATA). If not, the request is rejected (IXCNOTERSNANSLENMORE). If the answer area is large enough for the note pad data record but not large enough to hold all of the system connection records (IXCYNOTE\_TSYSCONNDATA), the answer area is filled with the note pad data record and as many system connection records as will fit. The request completes with an indication that more storage is needed to obtain all of the records (IXCNOTERSNMOREDATA).

If an answer area is not provided, one is in effect testing for the existence of the note pad.

**,REQTYPE=DELETE**

Delete the indicated note pad, provided the specified conditions (if any) are satisfied.

Any existing notes will be deleted as part of deleting the note pad.

Any existing connections will be deleted as part of deleting the note pad. Depending on the timing as to when the deletion is recognized, connections that are deleted in this manner can have their IXCNOTE requests rejected with a variety of reason codes (including for example, IXCNOTERSNNOTEPADNOTEXIST, IXCNOTERSNCONNECTIONNOTEXIST, and IXCNOTERSNNOTENOTEXIST).

#### **,REQTYPE=MODIFY**

Increase or decrease the number of notes a note pad is allowed to hold.

#### **,TAGGING=XCF**

#### **,TAGGING=USER**

When REQTYPE=CREATE is specified, use this optional parameter to indicate whether note tags are to be assigned by the user or by XCF.

Every note has a 16 byte note **tag** that is to be set (or preserved) whenever the note is manipulated (REQUEST=NOTE). The creator of the note pad determines whether note tags are to be assigned by the user or by XCF. Note tags must be assigned in the same way for the duration of the note pad. Thus all connectors to the note pad must adhere to the note tag assignment protocol specified by the creator of the note pad. The default is TAGGING=XCF.

#### **,TAGGING=XCF**

XCF is responsible for assigning tags to notes. XCF tag values are ever increasing. A new tag is assigned whenever a note is successfully created, replaced, or written. XCF does not assign a new tag when deleting a note.

#### **,TAGGING=USER**

The user is responsible for assigning tags to notes whenever a note is created, replaced, or written. One can also assign a new note tag when deleting a note.

User defined note tags can be used in any way that the exploiter deems suitable. For example, one could use tags to identify a collection of notes that are to be associated with each other, or one could use tags to assign a logical sequence number to each note. Alternatively, a tag could simply be additional metadata that is associated with the note.

Note that specifying TRACKTAG=CURRENT or TRACKTAG=LIFETIME imposes additional sequencing requirements on user assigned TAG values that are enforced when writing, replacing, or deleting existing notes.

#### **,TIMEOUT=timeout**

#### **,TIMEOUT=XCF**

Use this optional input parameter to indicate the number of seconds that the caller is willing to allow for the request to complete. The TIMEOUT specification imposes an upper bound on the duration of the request.

If the TIMEOUT keyword is not specified, or if TIMEOUT=XCF is specified, or if the specified value is zero, the maximum duration of the request is at the discretion of XCF.

The amount of time needed to complete a given REQTYPE will depend on the nature of the request as well as the state of the system and the note pad. In particular, there is the potential for any given request to be delayed while XCF processes previously accepted requests, some of which could be long running.

For REQUEST=NOTEPAD, XCF might need to access the Couple Data Set that contains the Coupling Facility Resource Manager (CFRM) policy that governs use of coupling facility resources in the sysplex. There can be a variety of conditions for which these accesses might require tens of seconds if not minutes to complete.

Choose TIMEOUT values accordingly.

In general, a REQUEST=NOTEPAD request requires multiple steps in the backend processing. If the request times out, the request might or might not have been processed successfully. Depending on the request and the design of the exploiting application, it might be appropriate to try the request again, perhaps with a longer timeout value. In some cases, one might need to issue a QUERY to determine whether the note pad exists.

The default is XCF.

**To code:** Specify the RS-type address, or address in register (2)-(12), of a halfword field, or specify a literal decimal value.

**,TRACKTAG=NO**

**,TRACKTAG=CURRENT**

**,TRACKTAG=LIFETIME**

When REQTYPE=CREATE is specified, use this optional parameter to indicate whether XCF needs to track the maximum tag value assigned to a note, and if so, how that value should be maintained and reported. The default is TRACKTAG=NO.

**,TRACKTAG=NO**

XCF need not track the maximum note tag values. When reporting the maximum tag value, XCF will always report zero. Requests to create, replace, write, or delete a note will not incur any additional overhead for tracking of the maximum note tag.

For user assigned tags (TAGGING=USER), any value can be assigned to the tag when writing, replacing, or deleting a note. In particular, the new tag value for the note can be less than, equal to, or greater than the current tag value of the note.

**,TRACKTAG=CURRENT**

XCF is to track the maximum tag of the notes that currently exist in the note pad. When reporting the maximum note tag value, XCF will report the maximum tag value assigned to any note that exists in the note pad when the query is made. If the note pad is empty, zero will be reported. Requests to create, replace, write, or delete a note will not incur any additional overhead for tracking of the maximum note tag.

For user assigned tags (TAGGING=USER), the tag value assigned when writing, replacing, or deleting a note must be greater than or equal to the current tag value of the note. In effect, TRACKTAG=CURRENT imposes a requirement that the tag values assigned to existing notes be nondecreasing. If the new tag value is less than the current tag value of the note, the request is rejected (IXCNOTERSNNOTELOWTAG). Any tag value can be assigned when creating a note.

**,TRACKTAG=LIFETIME**

XCF is to track the maximum tag of any note that ever existed in the note pad. When reporting the maximum tag, XCF will report the maximum tag value ever assigned to any note, including deleted notes.

For user assigned tags (TAGGING=USER), the tag value assigned when writing, replacing, or deleting a note must be greater than or equal to the current tag value of the note. In effect, TRACKTAG=LIFETIME imposes a requirement that the tag values assigned to existing notes be nondecreasing. If the new tag value is less than the current tag value of the note, the request is rejected (IXCNOTERSNNOTELOWTAG). Any tag value can be assigned when creating a note.

Tracking the maximum tag value for the life of the note pad is not without cost. When deleting a note, XCF might have to defer deletion of the note until the maximum tag value can be recorded. In such cases, the note is said to be **pending delete**. XCF will automatically finish deleting the note, but a subsequent request for a note that is still pending delete at the time of the request could be delayed until XCF finishes deleting the prior instance of the note.

In practice, the need to defer delete requests and the additional overhead or delays (if any) encountered by other requests will vary according to the use of the note pad, the tags that are assigned to the notes, the attributes of the note pad, and the dynamics of the system.

## Parameters for REQUEST=CONNECTION

**,ACCESS=UPDATE**

**,ACCESS=READ**

When REQTYPE=CREATE is specified, use this optional parameter to indicate what type of access the indicated connection is to enjoy. The default is ACCESS=UPDATE.

**,ACCESS=UPDATE**

The connection can create, write, replace, read, or delete notes in the note pad.

The calling program must have SAF authorization that permits UPDATE access to the FACILITY class resource IXCNOTE.*owner.application*. If SAF is not available, or if there is no IXCNOTE.*owner.application* resource defined for the note pad in the FACILITY class, a program running in supervisor state or with a PKM allowing key 0-7 is permitted to create a connection.

**,ACCESS=READ**

The connection can read notes in the note pad, but it cannot create, write, replace, or delete notes.

The calling program must have SAF authorization that permits READ access to the FACILITY class resource IXCNOTE.*owner.application*. If SAF is not available, or if there is no IXCNOTE.*owner.application* resource defined for the note pad in the FACILITY class, a program running in supervisor state or with a PKM allowing key 0-7 is permitted to create a connection.

**,CONNECTION=connection**

Use this required input/output parameter to specify a storage area. A token for the subject connection will either be fetched from or stored into this area.

For REQTYPE=CREATE, CONNECTION is an output variable and a token representing the connection will be stored in the indicated area. This token must be passed on subsequent IXCNOTE requests that manipulate the connection or access notes in the note pad. Note that the requester must carefully preserve the CONNECTION token because this is the only opportunity to acquire the token. XCF does not make the token available via any other service or IXCNOTE request.

For all other REQTYPE specifications, CONNECTION is an input variable and a token will be fetched from the indicated storage area. The token identifies the connection that is to be processed. The token would have been returned by a prior IXCNOTE request that was issued to create the connection on the local system.

**To code:** Specify the RS-type address, or address in register (2)-(12), of a 32 character field.

**,DESCRIPTION=description**

When REQTYPE=CREATE is specified, use this required input parameter to specify a variable containing a description of the connector. The description might indicate the function, service, purpose, or role of the connector. Or it might indicate the application with which the connector is associated. The string can contain any alphanumeric (a-z, A-Z, 0-9), national (@, #, \$), or special (underscore or blank) character. Leading blanks and all blank descriptors are not permitted. Descriptions are case sensitive. XCF presents the connector description when providing information about the connection. The description also appears in various XCF messages and diagnostic data reports. The intended purpose is to help installations and service personnel understand how the system (or sysplex) might be impacted if there are problems with this connector or its use of the note pad.

The storage containing the DESCRIPTION must reside in the primary address space of the caller, or in a space addressable via a public entry on the Dispatchable Unit Access List (DU-AL), or in a common area data space. The storage must be accessible using the PSW key of the program making the request.

**To code:** Specify the RS-type address, or address in register (2)-(12), of a 32 character field.

**,INFO=info****,INFO=0**

When REQTYPE=CREATE is specified, use this optional input parameter to specify a variable which contains information about the connector. The content and interpretation of this information is determined by the creator of the connection. XCF associates a copy of the indicated data with the connector. The connector might, for example, use INFO to document the protocols that it supports.

The storage containing the INFO must reside in the primary address space of the caller, or in a space addressable via a public entry on the Dispatchable Unit Access List (DU-AL), or in a common area data space. The storage must be accessible using the PSW key of the program making the request.

The default is 0.

**To code:** Specify the RS-type address, or address in register (2)-(12), of a 64-character field.

**,NOTEPAD=notepad**

When REQTYPE=CREATE is specified, use this required input parameter to specify a variable containing the name of an existing note pad to which a connection is to be created.

Note pad names are mapped by IXCYNOTE\_TNOTEPADNAME. Note pad names consist of four 8 byte sections. The various sections identify the owner of the note pad, the application using the note pad, and its function. Each 8 byte section must be left justified, padded on the right with EBCDIC blanks as needed. Each section can contain any uppercase alphabetic (A-Z), numeric (0-9), national (@, #, \$), or underscore (\_) character.

**To code:** Specify the RS-type address, or address in register (2)-(12), of a 32 character field.

**,REQTYPE=CREATE**

**,REQTYPE=DELETE**

**,REQTYPE=PAUSE**

**,REQTYPE=RESUME**

Use this required parameter to indicate the type of request to be processed for the connection.

**,REQTYPE=CREATE**

Create a connection to the note pad.

**,REQTYPE=DELETE**

Delete the connection to the note pad.

If there is no further need for the note pad, one should explicitly delete the note pad (by issuing IXCNOTE REQUEST=NOTEPAD REQTYPE=DELETE) so that system resources can be released in a timely manner.

A DELETE request can be issued by a work unit that can satisfy any of the following conditions:

- Requester is the connector
- Home is connector address space and the requester has SAF authorization appropriate for the ACCESS specified by the creator of the connection
- Running in supervisor state or with a PKM allowing key 0-7 and either:
  - The creator of the connection specified USAGE=SERVER and primary is the connector address space, or
  - The creator of the connection specified USAGE=CLIENT

**,REQTYPE=PAUSE**

Wait for a condition to clear. The service routine is to suspend the caller until the relevant condition is resolved, or it is determined that the condition cannot be resolved. The service routine will also return when the indicated TIMEOUT value expires. The service routine might also return if the state of any relevant condition changes.

For example, a prior IXCNOTE request might have been rejected with a return and reason code indicating that the note pad is quiesced. While quiesced, all attempts to access notes in the note pad will be rejected. The connector can use REQTYPE=PAUSE to determine when typical note activity can be resumed.

A connection could be quiesced as the result of several different conditions occurring simultaneously. While paused, there could be a change in the set of quiesce conditions. Some conditions might clear, while new ones might occur. The service routine always returns to the caller when the set of quiesce conditions clears and the note pad is once again accessible. The service routine might also return to the caller when the set of quiesce conditions changes, even though the connection remains quiesced.

The paused unit of work will also be resumed if the connection is deleted or if the note pad fails. Note that "connection is deleted" includes the case where the connection is deleted due to termination of the TERMSCOPE entity, in which case the paused unit of work will be resumed if



the work unit is still viable. One can also resume the work unit by invoking IXCNOTE to process a REQTYPE=RESUME request (from some other unit of work).

The return and reason code presented to the caller of the PAUSE request indicate whether the note pad is accessible upon return from the PAUSE. Return code 0 implies that the connection no longer has any conditions that would preclude typical use. Return code 4 is used when the connection still has conditions for which a PAUSE might be appropriate. Return code 8 is used if the connection no longer exists.

Thus upon return from the service routine, the connection might still be quiesced. The connector can elect to reissue the PAUSE in order to continue waiting for the connection to become unquiesced, or it might take some other relevant recovery action according to the specific quiesce conditions that remain.

For a given connection, at most one unit of work can be paused for a REQTYPE=PAUSE request.

A PAUSE request can be issued by a work unit that can satisfy any of the following conditions:

- Requester is the connector
- Home address space is the connector address space
- Primary address space is the connector address space
- Running in supervisor state or a PKM allowing key 0-7

If an answer area is provided (ANSAREA), information related to the pause conditions is stored in the field AA\_DETAILS when the request completes with return code 0 or 4. The details are mapped by IXCYNODE\_TDETAILSRESUMED.

#### **,REQTYPE=RESUME**

Resume the work unit that is currently paused for the indicated connection. Use this service to release a paused connection (REQTYPE=PAUSE) before the quiescing conditions clear or before the connection is deleted.

If a work unit is currently paused for the connection, the suspended work unit will be resumed and the PAUSE request will return to its caller. If a work unit is not currently paused for the connection, the next PAUSE request to be accepted will be resumed immediately.

A RESUME request can be issued by a work unit that can satisfy any of the following conditions:

- Requester is the connector
- Home address space is the connector address space
- Primary address space is the connector address space
- Running in supervisor state or a PKM allowing key 0-7

#### **,TERMSCOPE=TASK**

#### **,TERMSCOPE=HOME**

#### **,TERMSCOPE=PRIMARY**

When REQTYPE=CREATE is specified, use this required parameter to indicate an entity with which the connection is to be associated for the purposes of termination processing. If the indicated entity terminates, XCF deletes the connection. Note that a connection is always deleted when the local system terminates, or when the connector address space terminates. For a connection with task scope (TCBSENV is nonzero), the connection is always deleted if the connector task terminates.

The TERMSCOPE keyword enables the creator of the connection to bind the existence of the connection to a designated task or space. A connection cannot be associated with a task or address space that is in the midst of being (or has already) terminated.

#### **,TERMSCOPE=TASK**

The connection is to be terminated when the indicated task or the address space in which it resides terminates.

#### **,TERMSCOPE=HOME**

The connection is terminated when the home address space of the caller terminates.

To be precise, the connection will be deleted when the task that owns the cross-memory resources in the home address space terminates. The Task Control Block (macro IKJTCTB) for that task is anchored in the ASCBXTCTB field of the Address Space Control Block (macro IHAASCB) for the address space.

**,TERMSCOPE=PRIMARY**

The connection is terminated when the primary address space of the caller terminates.

To be precise, the connection will be deleted when the task that owns the cross-memory resources in the primary address space terminates. The Task Control Block (macro IKJTCTB) for that task is anchored in the ASCBXTCTB field of the Address Space Control Block (macro IHAASCB) for the address space.

**,TIMEOUT=*timeout***

**,TIMEOUT=XCF**

Use this optional input parameter to indicate the number of seconds that the caller is willing to allow for the request to complete. The TIMEOUT specification imposes an upper bound on the duration of the request.

If the TIMEOUT keyword is not specified, or if TIMEOUT=XCF is specified, or if the specified value is zero, the amount of time allotted to allow for the request to complete is at the discretion of XCF.

The amount of time needed to complete a given REQTYPE will depend on the nature of the request as well as the state of the system and the note pad. In particular, there is the potential for any given request to be delayed while XCF processes previously accepted requests, some of which could be long running.

For REQTYPE=PAUSE, note that many quiescing conditions are not likely to be resolved for several minutes. In some cases, manual intervention can be required to resolve the problem.

Choose TIMEOUT values accordingly.

For a REQTYPE other than PAUSE, the request is processed asynchronously. If the request times out, XCF will not be able to provide the results of the request. However, the request might or might not have been processed successfully. Depending on the request and the design of the exploiting application, it might be appropriate to try the request again, perhaps with a longer timeout value. If a REQTYPE=CREATE request times out, XCF will make sure that the connection does not exist before returning.

The default is XCF.

**To code:** Specify the RS-type address, or address in register (2)-(12), of a halfword field, or specify a literal decimal value.

**,TTOKEN=*ttoken***

**,TTOKEN=CURRENT**

When TERMSCOPE=TASK and REQTYPE=CREATE are specified, use this optional input parameter to contain the task token of the relevant task. The TCBTOKEN macro is typically used to obtain task tokens.

If TTOKEN=CURRENT is specified or taken as the default, or if the content of the TTOKEN variable is hexadecimal zero, the system uses the task token of the current task. SRB mode callers must explicitly pass a nonzero TTOKEN.

For a program running in problem state with a PKM allowing key 8-15, the TTOKEN must represent a task that resides in the home address space. The specified task must be the job step program task or a descendent thereof. The specified task must also be the current task or an ancestor thereof.

For a program running in supervisor state or with a PKM allowing key 0-7, the TTOKEN must represent a task that resides in either the home address space or the primary address space of the caller. The default is CURRENT.

**To code:** Specify the RS-type address, or address in register (2)-(12), of a 16-character field.

**,USAGE=CONNECTOR****,USAGE=SERVER****,USAGE=CLIENT**

When REQTYPE=CREATE is specified, use this optional parameter to specify the intended use of the connection. When the connection is subsequently used to process notes in the note pad (REQUEST=NOTE or REQUEST=NOTES), USAGE determines the conditions under which the requester is deemed to be a valid user.

To specify USAGE=SERVER or USAGE=CLIENT, the creator of the connection must be running in supervisor state or with a PKM allowing key 0-7, and the primary address space must be the home address space. In addition, for USAGE=SERVER, a task mode caller must not have a task specific security environment (TCBSENV must be zero).

The default is USAGE=CONNECTOR.

**,USAGE=CONNECTOR**

The connection is to be used by work units running out of the connector address space.

When processing notes, a work unit is deemed to be a valid user of the connection if it can satisfy any of the following conditions:

- Requester is the connector
- Home is connector space and user has SAF authorization appropriate for the REQTYPE
- Supervisor state or PKM allowing key 0-7, and running as an address space resource manager

**,USAGE=SERVER**

The connection is to be used by a server that processes notes while running in its own address space.

When processing notes, a work unit is deemed to be a valid user of the connection if it can satisfy any of the following conditions:

- Requester is the connector
- Home is connector space and user has SAF authorization appropriate for the REQTYPE
- Supervisor state or PKM allowing key 0-7, and running as an address space resource manager
- Primary is connector space, and requester is supervisor state or PKM allowing key 0-7<sup>1</sup>

USAGE=SERVER is typically used in a client/server application where a client running in an arbitrary address space calls the server to perform some service that executes while running in the server address space. The server address space creates the note pad connection for its own internal use. Since the client is neither the connector nor running out of the connector address space, XCF would typically deny the server request to access its own note pad from the client thread. With the SERVER option, XCF allows the client work unit to access the note pad if the server (connector) address space is primary at the time of the request, and the program is running supervisor state or with a PKM allowing key 0-7.

**,USAGE=CLIENT**

The connection is to be used by a server that processes notes while running in the client address space.

When processing notes, a work unit is deemed to be a valid user of the connection if it can satisfy any of the following conditions:

- Requester is the connector
- Home is connector space and user has SAF authorization appropriate for the REQTYPE

---

<sup>1</sup> Note that in this case, XCF has not verified that the requester is permitted to access the note pad. So the exploiter, who must be running authorized, is responsible for ensuring that use of the note pad connection is appropriate. Typically this condition is satisfied because the server is processing notes for its own internal purposes and is not giving its clients access to the notes. If the server gives a client access to the notes, it might well be necessary for the server to ensure that the client is authorized to make such access.

- Supervisor state or PKM allowing key 0-7, and running as an address space resource manager
- Requester is supervisor state or PKM allowing key 0-7<sup>1</sup>

USAGE=CLIENT is typically used in a client/server application where a client running in an arbitrary address space calls the server to perform some service that executes while running in the client address space. The server address space creates the note pad connection for its own internal use. Since the client is neither the connector nor running out of the connector address space, XCF would typically deny the server request to access its own note pad from the client thread. With the CLIENT option, XCF allows the client work unit to access the note pad, provided the program is running supervisor state or with a PKM allowing key 0 to 7.

## Parameters for REQUEST=NOTE

**,BUFFER=buffer**

**,NOBUFFER**

Use this required input parameter to specify how XCF is to handle the contents of the note.

**,BUFFER=buffer**

One of a set of mutually exclusive keywords indicating the buffer into/from which the note content is stored/fetched. When creating, replacing or writing a note, the buffer names an input area that contains the data that is to be written as a note in the note pad. When reading or deleting a note, the buffer is an output area into which XCF is to place a copy of the note that was fetched from the note pad. Refer to the description of BUFLLEN for additional details.

The storage containing the BUFFER must reside in the primary address space of the caller, or in a space addressable via a public entry on the Dispatchable Unit Access List (DU-AL), or in a common area data space. The storage must be accessible using the PSW key of the program making the request.

**To code:** Specify the RS-type address, or address in register (2)-(12), of a character field.

**,NOBUFFER**

One of a set of mutually exclusive keywords indicating that no buffer is being provided for the note request.

When creating a note, the note will have no content (the size of the note will be zero). A note with no content is called a **null note**. When replacing a note, the current content of the note will not be changed. If writing a note, the content of the note will not be changed if the note exists, and the note will be null if it does not exist. Use BUFFER with a BUFLLEN of zero if an existing note is to be replaced or written with null content.

When reading or deleting a note, the request will be processed but the content of the note will not be copied into storage.

**,BUFLLEN=buflen**

When BUFFER=buffer is specified, use this required input parameter to specify a variable containing the size in bytes of the buffer. The specified value must be a multiple of 1024 (1K).

If BUFLLEN is zero, then:

- When reading or deleting a note, the request is rejected unless the note to be processed is a null note (has no content).
- When creating, replacing, or writing a note, the resulting note will be a null note. In particular, note that when writing or replacing an existing note, the existing content of the note (if any) will be lost when the BUFLLEN value is zero. Use NOBUFFER if the content of the note is to be preserved.

If BUFLLEN is nonzero, then:

- When reading or deleting a note, the nonzero BUFLLEN must be greater than or equal to the size of the note being processed. The number of bytes stored in the buffer will be the actual note size. If the buffer is larger than the note size, XCF does not update any bytes in the buffer beyond the end of the note. If the buffer is smaller than the actual note size, the request is rejected.

- When creating, replacing, or writing a note, the nonzero BUFLen indicates the size of the note. The nonzero value must be less than or equal to the maximum note size defined by the creator of the note pad. If BUFLen exceeds the maximum note size defined by the creator of the note pad, the request is rejected.

**To code:** Specify the RS-type address, or address in register (2)-(12), of a fullword field, or specify a literal decimal value.

**,CONNECTION=connection**

Use this required input parameter to specify a variable containing the token representing the connection to the note pad.

This token was returned by a previous invocation of IXCNOTE REQUEST=CONNECTION REQTYPE=CREATE. The connection represented by the token must have been created with an ACCESS specification that is appropriate to the type of note request to be processed (REQTYPE). If created with ACCESS=UPDATE, the connection can create, write, replace, read, or delete a note. If created with ACCESS=READ, the connection can read a note.

**To code:** Specify the RS-type address, or address in register (2)-(12), of a 32 character field.

**,INSTANCE#=instance#**

**,INSTANCE#=IGNORE**

Use this optional input/output parameter to specify a variable for the instance number of the note. XCF sets the instance number to a value of its own choosing whenever a note is created, written, or replaced. The exploiter specifies the INSTANCE# keyword to indicate whether instance number comparisons are to be performed when reading, writing, replacing, or deleting a note (such comparisons are not done when creating a note). The instance number allows for a simple compare and swap like serialization that can be used to ensure that the expected copy of the note is being manipulated in the note pad. If the specified (nonzero) instance number equals the current instance number of the note, the request is processed. If not, the request is rejected.

If the creator of the note pad specified INSTCOMP=REQUIRED, a nonzero INSTANCE# must be specified when deleting, replacing, or writing an existing note.

If INSTANCE#=IGNORE is specified, or if the specified instance number equals zero, the request is processed without performing instance number comparisons. That is, the note will be read, written, replaced, or deleted without regard to its current instance number. If the specified instance number is nonzero, the read, write, replace, or delete of the note in the note pad will be performed only if the specified instance number equals the current instance number of the note.

For an INSTANCE# specification other than INSTANCE#=IGNORE, INSTANCE# could be an output variable or an input/output variable.

- When creating a note, INSTANCE# is an output variable into which the instance number assigned to the note by XCF is to be stored. Zero might be stored if the request is not successful.
- When writing, replacing, reading, or deleting a note, INSTANCE# is an input/output variable. A copy of the variable will be put in the IXCNOTE parameter list. If the note exists and the specified instance number is not zero, the service routine will use it for instance number comparisons. Upon return from the service routine, an instance number is stored in the indicated variable. Typically this value will be the current instance number of the note, regardless of whether the request was successfully processed or whether it failed due to an instance number mismatch. The original input value might be stored if the request is not successful for other reasons. In the case where the request fails due to an instance number mismatch, the output value would be the instance number to specify on a subsequent request in order to pass the instance number comparison check for the note (assuming the note was not updated by some other request in the meantime).

The default is IGNORE.

**To code:** Specify the RS-type address, or address in register (2)-(12), of an 8 character field.

**,KEEPNOTE=NO****,KEEPNOTE=YES**

Use this optional parameter to indicate whether the note should be kept or deleted when the connection associated with the note is deleted. A note is associated with the connection if the connection created the note and it was never replaced, or if the connection was the last to replace the note (regardless of which connection might have created the note).

The KEEPNOTE specification is an attribute of the note that gets set whenever the note is created, written, or replaced. Thus the KEEPNOTE keyword must not be specified when reading or deleting a note since these operations cannot change the attribute.

Note that the KEEPNOTE attribute is subject to change whenever a note is replaced. Thus if the KEEPNOTE attribute is to be preserved, the connector must take care to code the appropriate KEEPNOTE specification each time the note is replaced. The default is KEEPNOTE=NO.

**,KEEPNOTE=NO**

The note is to be deleted when the associated connection is deleted.

**,KEEPNOTE=YES**

The note is to be kept when the associated connection is deleted.

The note will remain in the note pad until it is explicitly deleted by a subsequent IXCNOTE request, or until the note pad is deleted. Also note that the note will be eligible for deletion if KEEPNOTE=NO is specified (possibly as a default) when the note is subsequently replaced.

**,REQTYPE=CREATE****,REQTYPE=WRITE****,REQTYPE=REPLACE****,REQTYPE=READ****,REQTYPE=DELETE**

Use this required parameter to indicate the type of request to be processed for the indicated note.

**,REQTYPE=CREATE**

Create a new note in the note pad. The request is rejected if the named note already exists.

**,REQTYPE=WRITE**

If the named note does not already exist, create it. If the named note does exist, replace it.

**,REQTYPE=REPLACE**

Replace an existing note in the note pad. The request is rejected if the named note does not already exist.

**,REQTYPE=READ**

Read an existing note from the note pad. The request is rejected if the named note does not already exist.

**,REQTYPE=DELETE**

Delete an existing note from the note pad. The request is rejected if the named note does not already exist.

If a buffer area (BUFFER) is provided, a copy of the deleted note will be stored in the buffer if the request is successful.

If the creator of the note pad specified TRACKTAG=LIFETIME, deletion of the note might be deferred so that XCF can ensure that the maximum note tag value is preserved. In such cases the note is said to be **pending delete**. XCF will automatically complete deletion of the note asynchronously to the caller. Deferring the delete will not directly impact the delete request.

However, notes that are pending delete will continue to consume note pad resources until XCF finishes deleting the note. They could, for example, be included in counts of the number of notes that exist in the note pad. A subsequent request to process a given note can incur additional overhead if the named note is still pending delete when the request is made. In such cases, XCF might need to finish the pending delete before it can proceed with the request.

If the creator of the note pad specified TAGGING=USER and TRACKTAG=LIFETIME, be aware that the likelihood of the delete request being deferred increases if one assigns a new note tag that is greater than the current tag value for the note. Assigning a new tag value that would be the new maximum tag value of the note pad will certainly induce deferral. Consider using TAG=KEEP when deleting notes in order to minimize the potential for deferring of the request.

Note that return code 0 is provided regardless of whether the delete request is performed synchronously or deferred. The answer area (ANSAREA), if any, will indicate whether the delete request was deferred.

#### **,NAME=name**

Use this required input parameter to specify a variable containing the name of the note to be processed. The note name uniquely identifies a note in the note pad. There are no restrictions or requirements for the note name other than it be unique for each note in the note pad. Different note pads can use the same note names without conflict.

**To code:** Specify the RS-type address, or address in register (2)-(12), of an 8 character field.

#### **,TAG=tag**

#### **,TAG=KEEP**

Usage of the TAG parameter depends on the value that is coded for the TAGGING parameter.

- When TAGGING=USER is specified, use this optional input/output parameter to specify a variable into/from which the connector assigned note tag is stored/fetched.

- For a TAG specification other than TAG=KEEP

When reading a note, TAG is an output variable into which the current tag of the note is to be stored. Zero might be stored if the request is not successful.

When creating a note, TAG is an input/output variable. The tag value to be assigned to the note will be fetched from the indicated variable. If the request is rejected because the note already exists (IXCNOTERSNNOTEEXISTS), the current tag value for the note is stored. Otherwise the tag value stored will be the original input value, which for a successful request, would be the tag value that was set for the newly created note.

When replacing, writing, or deleting a note, TAG is an input/output variable. For these requests, the tag value to be assigned to the note will be fetched from the indicated variable. If the request is rejected due to an instance number mismatch (IXCNOTERSNNOTEBADINSTANCE#), or because the specified tag value was less than the current tag value of the note (IXCNOTERSNNOTELOWTAG), the current tag value for the note is stored. Otherwise the tag value stored will be the original input value, which for a successful request, would be the tag value that was set for the note.

When writing a note, the indicated TAG value will be assigned to the note regardless of whether the write request causes the note to be created or replaced.

When deleting a note from a note pad for which the creator of the note pad specified TRACKTAG=LIFETIME, the delete request could be deferred if XCF needs to ensure that the maximum note tag value is preserved.

- For TAG=KEEP

If TAG=KEEP is specified (or taken as a default), a tag value is neither fetched nor stored. When creating a note, the tag value assigned to the note will be zero. When reading, replacing, or deleting a note, the note tag will not be changed. When writing a note, the note tag will be set to zero if the note is created, but will not be changed if the note is replaced.

The default is KEEP.

- When TAGGING=XCF is specified, an optional output parameter variable into which the XCF assigned tag value for the note is to be stored. If TAG is specified and the request is not successful, zero might be stored.

**To code:** Specify the RS-type address, or address in register (2)-(12), of a 16-character field.

**,TAGGING=USER****,TAGGING=XCF**

Use this required parameter to indicate how the connector expects (or desires) tags to be processed.

Every note has a 16 byte note **tag** that is set (or preserved) whenever the note is manipulated. If TAGGING=USER was specified by the creator of the note pad, the tags are to be assigned by the connector. If TAGGING=XCF was specified by the creator of the note pad, the tags are assigned by XCF.

**,TAGGING=USER**

The connector claims to be responsible for assigning tags to the notes. The request is rejected if the creator of the note pad specified TAGGING=XCF.

For user assigned tags (TAGGING=USER) where the creator of the note pad specified TRACKTAG=CURRENT or TRACKTAG=LIFETIME, the tag value assigned when writing, replacing, or deleting a note must be greater than or equal to the current tag value of the note. If the new tag value is less than the current tag value of the note, the request is rejected (IXCNOTERSNNOTELOWTAG). Any tag value can be assigned when creating a note.

**,TAGGING=XCF**

The connector claims that XCF is responsible for assigning tags to the notes. The request is rejected if the creator of the note pad specified TAGGING=USER.

XCF will assign a new tag when creating, replacing, or writing a note. XCF does not change the note tag when reading or deleting a note. XCF note tags are ever increasing.

## Parameters for REQUEST=NOTES

**,BUFFER=buffer****,NOBUFFER**

When REQTYPE=READ is specified, use this required input parameter to specify how XCF is to handle the contents of the notes.

**,BUFFER=buffer**

One of a set of mutually exclusive keywords indicating the buffer into which the content of the selected notes is to be stored. Only complete notes will be stored in the buffer. When the buffer is filled, the read request returns. In the answer area, the data record for each note (IXCYNOTE\_TNOTEDATA) will indicate whether the content of the note was stored in the BUFFER area, and if so, where.

The storage containing the BUFFER must reside in the primary address space of the caller, or in a space addressable via a public entry on the Dispatchable Unit Access List (DU-AL), or in a common area data space. The storage must be accessible using the PSW key of the program making the request.

**To code:** Specify the RS-type address, or address in register (2)-(12), of a character field.

**,NOBUFFER**

One of a set of mutually exclusive keywords indicating that the content of the notes are not to be stored. Information about the notes will be stored in the answer area (ANSAREA).

**,BUFLEN=buflen**

When BUFFER=buffer and REQTYPE=READ are specified, use this required input parameter to specify a variable containing the size in bytes of the buffer. BUFLLEN must be a multiple of 1024. If BUFLLEN is less than the size needed to hold the first stored note, the request is rejected. If one or more notes are already stored but the remaining specified space is not enough to store the next note, the request completes with only the notes that fit. In general, the specified BUFLLEN value should be greater than or equal to the largest note supported by the note pad. BUFLLEN can be zero, but the request will be rejected if any note other than a null note is selected. A null note has no content (zero length).

**To code:** Specify the RS-type address, or address in register (2)-(12), of a fullword field, or specify a literal decimal value.



**,CHOOSE=ALL****,CHOOSE=BYCRITERIA**

Use this required parameter to indicate how the set of notes to be processed is to be determined.

**,CHOOSE=ALL**

All notes in the note pad are to be selected.

**,CHOOSE=BYCRITERIA**

The notes that meet the indicated selection criteria are to be processed. Various selection criteria can be specified. For example, one can select notes whose tag values are within a particular range.

**,CONNECTION=connection**

Use this required input parameter to specify a variable containing the token that represents the connection to the note pad.

This token was returned by a previous invocation of IXCNOTE REQUEST=CONNECTION REQTYPE=CREATE. The connection represented by the token must have been created with an ACCESS specification that is appropriate to the type of request to be processed (REQTYPE). If created with ACCESS=UPDATE, the connection can read or delete notes. If created with ACCESS=READ, the connection can read notes.

**To code:** Specify the RS-type address, or address in register (2)-(12), of a 32 character field.

**,CRITERIA=criteria**

When CHOOSE=BYCRITERIA is specified, use this required input parameter to specify a storage area containing selection criteria which is mapped by IXCNOTE\_TSELECTIONCRITERIA. A selection element contains data that defines the criteria that are to be used to select the notes of interest. See the IXCNOTE macro for additional details on how to specify various selection criteria.

The storage containing the CRITERIA must reside in the primary address space of the caller, or in a space addressable via a public entry on the Dispatchable Unit Access List (DU-AL), or in a common area data space. The storage must be accessible using the PSW key of the program making the request.

**To code:** Specify the RS-type address, or address in register (2)-(12), of a character field.

**,MAXTAG=maxtag****,MAXTAG=NONE**

When REQTYPE=DELETE is specified, use this optional input parameter to specify a variable containing a note tag value. If MAXTAG=NONE is specified (or taken as the default), all of the selected notes are deleted without regard to tag value. If a MAXTAG value other than NONE is specified, a selected note will be deleted if its tag value is less than or equal to the indicated MAXTAG value.

If the creator of the note pad specified TRACKTAG=LIFETIME, these additional considerations apply:

- If notes are deleted, XCF will have atomically preserved the maximum tag value of the note pad if any of the deleted notes had the maximum tag value ever assigned for the life of the note pad.
- If the creator of the note pad specified TAGGING=USER, then after the selected notes are deleted, XCF determines whether the specified MAXTAG value is a new maximum tag value for the note pad. If so, XCF preserves the indicated MAXTAG value as the new maximum note tag value for the note pad.

**Note:** Although preservation of the maximum tag value assigned to the deleted notes is atomic, the setting of the specified MAXTAG value as the new maximum note tag for the note pad is not atomic. Thus as the result of a failure, the specified MAXTAG value might not get preserved in the note pad. If the exploiter cannot tolerate a failure to record the specified MAXTAG value as the new maximum tag value, consider using REQUEST=NOTE to delete the notes one at a time. Preservation of the maximum note tag is guaranteed to be atomic when deleting a single note.

The default is NONE.

**To code:** Specify the RS-type address, or address in register (2)-(12), of a 16-character field.

**,REQTYPE=READ**  
**,REQTYPE=DELETE**

Use this required parameter to indicate the type of request to be processed.

**,REQTYPE=READ**

Read one or more notes from the note pad. Metadata for each of the selected notes will be stored in the answer area (ANSAREA). The note content will be stored in the buffer area (BUFFER). An answer area is required. A buffer area is optional. A successful read request returns to the caller when all of the selected notes have been read, or when the storage areas provided for output are filled with as many notes as will fit, whichever comes first.

If a read request completes prematurely because one or both of the provided storage areas are full, the service routine provides a warning return and reason code (IXCNOTERSNMORENOTES). If so, a nonzero resume token (RESUMETOKEN) is also stored. After processing the notes that were read, one can continue reading the remaining notes by reissuing the same request, passing the aforementioned nonzero resume token via the RESUMETOKEN keyword to continue reading from where the prior read request left off. When all of the notes have been read, the service routine sets return code zero. Note that the number of notes returned in this case could be zero. Also note that a nonzero RESUMETOKEN is returned for the return code zero case as well. One could for example, later issue a new read notes request with this token to read subsequently created notes.

XCF inspects the notes in the order in which they were originally created. As a result, be aware of the following:

- If an existing note is replaced one or more times while a read notes request is being processed, at most one of the instances will be returned by the request. A note is created by a create note request, or by a write request when the note does not exist. Replacing the content of an existing note does not cause the note to be created. Thus after a read operation inspects a note, that note will not be inspected again by the ongoing read operation or any subsequent resume of the read request, regardless of how many times the note is replaced.
- If a note is created while a read notes request is being processed, the newly created note might not be returned by the ongoing read operation. In some cases, the timing could be such that a note would not be returned even when the read request is resumed using the RESUMETOKEN returned by the read request that was active when the note was created.
- If a note is deleted and created anew while a read notes request is being processed, there is the potential for the note to be reported multiple times by the read request. For example, the read request could fetch a given note and store it in the user ANSAREA. Meanwhile, some other process could delete the note and create a new instance of the same note (one with the same name). That new instance of the note could be visible to the ongoing read operation, or to a subsequent resume of the read request, and so could be returned again.

An answer area (ANSAREA) must be provided when reading notes. For each of the selected notes, a data record containing information about the note (such as its name and tag) will be stored in the answer area. The data record is mapped by IXCYNOTE\_TNOTEDATA.

The answer area must be large enough to allow information for at least one note to be returned. If the answer area is not large enough to allow information for the required minimum number of notes to be returned, the service routine rejects the request (IXCNOTERSNANSLENMORE).

The content of the notes will be returned in the indicated buffer area (BUFFER). The buffer area is optional. If a buffer is not provided, the content of the notes will not be read. If a buffer area is provided, the data record stored in the answer area for each note will indicate the location within the buffer area where the content of the corresponding note was stored (if any).

The BUFFER area must be large enough to allow the content of at least one note to be stored. If the BUFFER area is not large enough for the minimum required number of notes, the service routine rejects the request (IXCNOTERSNBUFLENBADVAL).

**,REQTYPE=DELETE**

Delete the indicated set of notes from the note pad. The request will not return until all of the selected notes are processed.

If provided, the answer area will indicate how many notes were deleted. No note information is stored for any of the deleted notes.

Note that the note pad itself persists until it is deleted. Deleting all the notes in the note pad does not delete the note pad.

#### **,RESUMETOKEN=resume token**

When REQTYPE=READ is specified, use this required input/output parameter to specify a variable that contains a resume token for resuming a read request that completes prematurely. Specifying a resume token initialized to all zeros causes the read request to consider all the notes in the note pad. Specifying a nonzero resume token causes the read request to continue reading from where a prior request left off.

When specifying a nonzero resume token, only use the value that was returned by the previous invocation of a read request initiated by the subject connection for the subject note pad. Otherwise XCF might reject the request with an indication that the resume token is not valid (IXCNOTERSNRESUMETOKENBADVAL), or if the request is in fact processed, the request might not return the expected set of notes.

When passing a nonzero RESUMETOKEN, the caller should in general also pass the same selection criteria as was passed when the request was first initiated, and continue doing so for each iteration of the read request until it is complete. Otherwise the request will likely not return the intended set of notes. For example, the same note could be read twice or a note that meets the selection criteria might not be read at all.

Typically, the reading of multiple notes will use logic similar to the following:

```
Set ResumeToken to zero
Do until all notes have been read (rc=0)
  IXCNOTE REQUEST=NOTES,REQTYPE=READ,
    CHOOSE=ALL,
    RESUMETOKEN=ResumeToken,
    CONNECTION=connect_token,
    ANSAREA=ansarea,ANSLEN=anslen,
    BUFFER=buffer,BUFLen=buflen,
    RETCODE=rc,RSNCODE=rsn
  If rc/rsn indicate error
    Deal with error
    Terminate loop
  Else (rc=0 or rc=4)
    Process returned notes (if any)
End Do
```

**To code:** Specify the RS-type address, or address in register (2)-(12), of a 32 character field.

## ABEND Codes

Abend codes that an issuer of IXCNOTE might receive are listed below. For detailed abend code information, see [z/OS MVS System Codes](#).

- Abend X'073' - Environment not valid. The caller held a lock.

## Return and Reason Codes

When the IXCNOTE macro returns control to your program:

- GPR 15 (and *retcode*, when you code RETCODE) contains a return code.
- When the value in GPR 15 is not zero, GPR 0 (and *rsncode*, when you code RSNCODE) contains a reason code. Note that bits 0-15 of the reason code might contain component diagnostic data for use by IBM service personnel. XCF provides the IXCNOTERSNCODEMASK constant to mask off the component-diagnostic data.

The IXCYCON macro provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

**0** IXCRETCODEOK  
**4** IXCRETCODEWARNING  
**8** IXCRETCODEPARMERROR  
**C** IXCRETCODEENVERROR  
**10** IXCRETCODECOMPERROR

The following table identifies the hexadecimal return and reason codes. IBM support personnel might request the entire reason code, including the **xxxx** value.

Table 14. Return and Reason Codes for the IXCNOTE Macro		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0	None.	<b>Meaning:</b> Successful completion.
4	xxxx0401	<p><b>Equate Symbol:</b> IXCNOTERSNMOREDATA</p> <p><b>Meaning:</b> The request was processed and some of the requested data was stored in the answer area. The ANSAREA is large enough to hold the header (IXCYNOTE_TANSAREA) but does not have room for all the requested data. The AA_ANSAREASIZENEEDD returned in the ANSAREA indicates how much storage is required in order for the ANSAREA to hold all the requested data.</p> <p>For requests that produce varying quantities of output data due to the dynamics of the system (for example, query of a note pad), the actual amount of storage needed for the answer area when the request is reissued can differ from the indicated size. One might choose to obtain more storage than indicated by AA_ANSAREASIZENEEDD in order to reduce the likelihood of getting this return and reason code when the request is reissued.</p> <p><b>Action:</b> As needed, obtain storage for the answer area that is at least as large as the indicated size and reissue the request.</p>
4	xxxx0402	<p><b>Equate Symbol:</b> IXCNOTERSNMORENOTES</p> <p><b>Meaning:</b> A request to read notes in the note pad completed prematurely. The answer area, and as applicable, the buffer area, were filled with as many notes as would fit.</p> <p><b>Action:</b> After processing these results, continue reading the remaining notes by reissuing the IXCNOTE request specifying the RESUMETOKEN that was returned in the RESUMETOKEN output variable.</p>
4	xxxx0403	<p><b>Equate Symbol:</b> IXCNOTERSNRESUMED</p> <p><b>Meaning:</b> As a result of an IXCNOTE REQUEST=CONNECTION REQTYPE=PAUSE request, the connector was being paused until the note pad was no longer quiesced. The pause request was resumed due to some condition other than the note pad becoming unquiesced. For example, some other work unit called IXCNOTE REQUEST=CONNECTION REQTYPE=RESUME to resume the paused work unit, or the connection terminated, or the note pad failed.</p> <p>If an answer area was provided, the field AA_DETAILS provides additional information. The data is mapped by IXCYNOTE_TDETAILSRESUMED.</p> <p><b>Action:</b> As needed, reissue the IXCNOTE REQUEST=CONNECTION REQTYPE=PAUSE request to wait for the quiesce conditions to clear. If an answer area was provided, examine the data returned in the answer area for additional information.</p>

Table 14. Return and Reason Codes for the IXCNOTE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	xxxx0441	<p><b>Equate Symbol:</b> IXCNOTERSNPENDING</p> <p><b>Meaning:</b> The request is pending.</p> <ul style="list-style-type: none"> <li>If REQUEST=CONNECTION REQTYPE=CREATE was specified, the connection was successfully created. However, the connection is currently in a quiesced state. If an answer area was provided, the field aa_Details provides additional information. The data is mapped by IXCYNODE_TDETAILSQUIESCED.</li> <li>If REQUEST=CONNECTION REQTYPE=RESUME was specified, no connection was found to be paused. The resume request was recorded so that the next pause request will be immediately resumed.</li> <li>If REQUEST=CONNECTION REQTYPE=DELETE was specified, the connection was marked for deletion. However, not all of the necessary cleanup could be accomplished because XCF was not able to access the relevant coupling facility structures. XCF will automatically complete the deletion when access to the structures is restored. If an answer area was provided, the field aa_Details provides additional information. The data is mapped by IXCYNODE_TDETAILSQUIESCED or by IXCYNODE_TDETAILSNORESOURCES.</li> <li>If REQUEST=NOTEPAD REQTYPE=DELETE was specified, the note pad was marked for deletion. However, not all of the necessary cleanup could be accomplished because XCF was not able to access the relevant coupling facility structures. XCF will automatically complete the deletion when access to the structures is restored. While deletion of the note pad is pending, some requests can get unexpected results. For example, you might be able to create a new connection to the note pad, or an attempt to create a new instance of the note pad might be rejected because the old instance still exists.</li> <li>If REQUEST=NOTEPAD, REQTYPE=MODIFY was specified to modify the #NOTES attribute for a note pad, not all of the necessary updates could be accomplished because XCF was not able to access the relevant coupling facility structures. XCF will automatically complete the modify request when access to the structures is restored.</li> </ul> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>If REQUEST=CONNECTION REQTYPE=CREATE or REQTYPE=DELETE was specified, consider using IXCNOTE REQUEST=CONNECTION REQTYPE=PAUSE to wait until the connection is no longer quiesced. If an answer area was provided, examine the data returned in the answer area for additional information.</li> <li>If REQUEST=NOTEPAD REQTYPE=DELETE was specified, consider using REQUEST=NOTEPAD REQTYPE=QUERY to verify that the note pad has been deleted before proceeding. You might need to be sensitive to the particular instance of the note pad (see NPD_ETODWHENCREATED in macro IXCYNODE).</li> <li>If REQUEST=NOTEPAD, REQTYPE=MODIFY was specified, consider one of the following: <ul style="list-style-type: none"> <li>When a connection to the note pad exists, consider using IXCNOTE REQUEST=CONNECTION REQTYPE=PAUSE to be resumed when XCF completes the MODIFY request.</li> <li>IXCNOTE REQUEST=NOTEPAD REQTYPE=QUERY can also be used to determine whether a MODIFY request is pending against the note pad. npd_Modifying#Notes is set ON if the note pad #NOTES is in the midst of being modified. When npd_Modifying#Notes is set OFF, npd_Required#Notes reflects the number of notes that the note pad is allowed to hold. npd_Required#Notes will not reflect the requested change to the #NOTES attribute while the modify request is pending.</li> </ul> </li> </ul>

Table 14. Return and Reason Codes for the IXCNOTE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0001	<p><b>Equate Symbol:</b> IXCNOTERSNPLISTBADSTG</p> <p><b>Meaning:</b> Program error. The storage containing the parameter list is not accessible. It must be both addressable and accessible using the PSW key of the caller.</p> <p>The error occurred while trying to store output data in the parameter list after the IXCNOTE request had been processed. The request might or might not have completed successfully. The note pad might or might not have been altered. In the specific case of REQUEST=CONNECTION REQTYPE=CREATE, the connection that was created, if any, has been deleted by XCF.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that the correct parameter list storage area was specified, and the storage area was not inadvertently freed.</li> <li>• As needed, verify whether the request had been processed using REQTYPE=QUERY or REQTYPE=READ.</li> <li>• Take an appropriate recovery action.</li> </ul>
8	xxxx0003	<p><b>Equate Symbol:</b> IXCNOTERSNPLISTBADRSVD</p> <p><b>Meaning:</b> Program error. A reserved field in the parameter list is not zero. Either the parameter list was corrupted or the program is running on a system that does not have the necessary support.</p> <p><b>Action:</b> Check to see if your program inadvertently overlaid the parameter list storage, and that it was assembled with the correct macro library for the release of z/OS your program is running on.</p>
8	xxxx0004	<p><b>Equate Symbol:</b> IXCNOTERSNPLISTBADVERSION</p> <p><b>Meaning:</b> Program error. The parameter list version number value is not valid. Either the parameter list was corrupted or the program is running on a system that does not have the necessary support.</p> <p><b>Action:</b> Check to see if your program inadvertently overlaid the parameter list storage, and that it was assembled with the correct macro library for the release of z/OS your program is running on.</p>
8	xxxx0005	<p><b>Equate Symbol:</b> IXCNOTERSNPLISTBADREQUEST</p> <p><b>Meaning:</b> Program error. The parameter list content for the REQUEST keyword is not valid. Either the parameter list was corrupted or the program is running on a system that does not have the necessary support.</p> <p><b>Action:</b> Check to see if your program inadvertently overlaid the parameter list storage, and that it was assembled with the correct macro library for the release of z/OS your program is running on.</p>
8	xxxx0006	<p><b>Equate Symbol:</b> IXCNOTERSNPLISTBADREQTYPE</p> <p><b>Meaning:</b> Program error. The parameter list content for the REQTYPE keyword is not valid. Either the parameter list was corrupted or the program is running on a system that does not have the necessary support.</p> <p><b>Action:</b> Check to see if your program inadvertently overlaid the parameter list storage, and that it was assembled with the correct macro library for the release of z/OS your program is running on.</p>
8	xxxx0008	<p><b>Equate Symbol:</b> IXCNOTERSNPLISTBADTAGGING</p> <p><b>Meaning:</b> Program error. The parameter list content for the TAGGING keyword is not valid. Either the parameter list was corrupted or the program is running on a system that does not have the necessary support.</p> <p><b>Action:</b> Check to see if your program inadvertently overlaid the parameter list storage, and that it was assembled with the correct macro library for the release of z/OS your program is running on.</p>

Table 14. Return and Reason Codes for the IXCNOTE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0010	<p><b>Equate Symbol:</b> IXCNOTERSNNOTEPADBADVAL</p> <p><b>Meaning:</b> Program error. The parameter list content for the NOTEPAD keyword is not valid. The indicated note pad name does not conform to the requirements for a valid name. Note pad names consist of four 8 byte sections. Each 8 byte section must be left justified, padded on the right with EBCDIC blanks as needed. Each section can contain any uppercase alphabetic (A-Z), numeric (0-9), national (@, #, \$), or underscore (_) character.</p> <p><b>Action:</b> Verify that the note pad name specified for the NOTEPAD keyword meets the requirements of the keyword.</p>
8	xxxx0011	<p><b>Equate Symbol:</b> IXCNOTERSNNOTEPADNOTEXIST</p> <p><b>Meaning:</b> The specified note pad does not exist. For REQUEST=NOTEPAD or for REQUEST=CONNECTION with REQTYPE=CREATE, the note pad named by the NOTEPAD keyword does not exist. For any other request, the note pad to which the CONNECTION token applies no longer exists. The connection token is no longer valid for use.</p> <p><b>Action:</b> As needed, create a new instance of the note pad, establish a new connection, and reconstruct the notes.</p>
8	xxxx0012	<p><b>Equate Symbol:</b> IXCNOTERSNNOTEPADFAILED</p> <p><b>Meaning:</b> The note pad failed and could not be recovered by XCF. It no longer exists. The connection token is no longer valid for use.</p> <p>This condition will typically be detected by an in-flight request. Subsequent attempts to issue IXCNOTE requests with the input connection token might be rejected with a return and reason code indicating that the connection no longer exists. IXCNOTE requests issued by other connectors might be rejected with a return and reason code indicating that the note pad no longer exists (IXCNOTERSNNOTEPADNOTEXIST) or that the connection no longer exists (IXCNOTERSNCONNECTIONNOTEXIST).</p> <p><b>Action:</b> As needed, create a new instance of the note pad, establish a new connection, and reconstruct the notes.</p>
8	xxxx0013	<p><b>Equate Symbol:</b> IXCNOTERSNNOTEPADEXISTS</p> <p><b>Meaning:</b> The specified note pad already exists.</p> <p>If an answer area was provided, the note pad data record (IXCYNOTE_TNOTEPADDATA) stored therein describes the existing note pad.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If an answer area was provided, review the note pad data returned in the answer area.</li> <li>• If an answer area was not provided, consider querying the note pad data using REQUEST=NOTEPAD REQTYPE=QUERY.</li> </ul>
8	xxxx0014	<p><b>Equate Symbol:</b> IXCNOTERSNNOTEPADINUSE</p> <p><b>Meaning:</b> Program error. A request to delete the note pad is rejected because the specified note pad is still in use. It either has connectors, or contains notes, or both.</p> <p>If an answer area was provided, the field AA_DETAILS provides additional information. The data is mapped by IXCYNOTE_TDETAILSDELETENP.</p> <p><b>Action:</b> Try the request again after the relevant connectors, notes, or both have been deleted, or specify the appropriate CONDITIONS when issuing the delete request.</p>
8	xxxx0015	<p><b>Equate Symbol:</b> IXCNOTERSNNOTEPADMULTIWRITENO</p> <p><b>Meaning:</b> A request to create a connection with ACCESS=UPDATE is rejected. Since the creator of the note pad specified MULTIWRITE=NO, at most one connector with ACCESS=UPDATE can be connected to the note pad. Such a connection already exists.</p> <p><b>Action:</b> Consider using REQUEST=NOTEPAD REQTYPE=QUERY to find out which system in the sysplex owns the connection that was created with ACCESS=UPDATE.</p>

Table 14. Return and Reason Codes for the IXCNOTE Macro (continued)		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0018	<p><b>Equate Symbol:</b> IXCNOTERSNCONNECTIONBADVAL</p> <p><b>Meaning:</b> Program error. The value specified for CONNECTION is not a valid connection token.</p> <p><b>Action:</b> Ensure the correct connection token was specified.</p>
8	xxxx0019	<p><b>Equate Symbol:</b> IXCNOTERSNCONNECTIONNOTEXIST</p> <p><b>Meaning:</b> The specified CONNECTION does not exist.</p> <p>In the specific case of REQUEST=CONNECTION REQTYPE=CREATE, the connection was created but then deleted before the service routine could return to the caller. You might see this situation, for example, if the entity indicated by TERMScope terminated while the connection was being created. You might also see this result if the work unit that requested the create fails to thrive. XCF deletes the connection if the requesting work unit appears to be unresponsive. Increasing the TIMEOUT can help overcome cases where the dynamics of the system are such that the requesting work unit does not get enough system resources to make timely progress.</p> <p><b>Action:</b> As needed, establish a new connection and retry the request.</p>
8	xxxx001B	<p><b>Equate Symbol:</b> IXCNOTERSNCONNECTIONBADTERM</p> <p><b>Meaning:</b> Program error. The TERMScope specification is not valid.</p> <p>A program running in SRB mode must either specify TERMScope=HOME, TERMScope=PRIMARY, or TERMScope=TASK with a nonzero TToken. When a program specifies TERMScope=TASK, the task indicated by the TToken keyword must be a task in either the home address space or the primary address space of the caller. When a program running in problem state with a PKM allowing key 8-15 specifies TERMScope=TASK, the relevant task must be the job step task or one of its descendents and it must be the current task or one of its ancestors.</p> <p><b>Action:</b> Verify that the TERMScope meets the requirements for the keyword.</p>
8	xxxx001C	<p><b>Equate Symbol:</b> IXCNOTERSNCONNECTIONBADPAUSE</p> <p><b>Meaning:</b> Program error.</p> <p>For REQTYPE=PAUSE, a work unit is already paused for the indicated connection. At most one work unit can be paused for a given connection.</p> <p><b>Action:</b> As an alternative to using the PAUSE request, the connector can simply retry a request after allowing some time for the condition to clear.</p>



Table 14. Return and Reason Codes for the IXCNOTE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx001D	<p><b>Equate Symbol:</b> IXCNOTERSNCONNECTIONBADUSER</p> <p><b>Meaning:</b> Program error. The specified CONNECTION token is not valid for use by the caller.</p> <p>When processing notes (REQUEST=NOTE or REQUEST=NOTES), the requester must satisfy at least one of the following conditions:</p> <ul style="list-style-type: none"> <li>Requester is the connector</li> <li>Home is connector address space and the requester has SAF authorization appropriate for the REQTYPE</li> <li>Running in supervisor state or with a PKM allowing key 0-7 and either: <ul style="list-style-type: none"> <li>Running as an address space resource manager, or</li> <li>The creator of the connection specified USAGE=SERVER and primary is the connector address space, or</li> <li>The creator of the connection specified USAGE=CLIENT</li> </ul> </li> </ul> <p>When deleting a connection (REQUEST=CONNECTION REQTYPE=DELETE), the requester must satisfy at least one of the following conditions:</p> <ul style="list-style-type: none"> <li>Requester is the connector</li> <li>Home is connector address space and the requester has SAF authorization appropriate for the ACCESS specified by the creator of the connection</li> <li>Running in supervisor state or with a PKM allowing key 0-7 and either: <ul style="list-style-type: none"> <li>The creator of the connection specified USAGE=SERVER and primary is the connector address space, or</li> <li>The creator of the connection specified USAGE=CLIENT</li> </ul> </li> </ul> <p>When pausing or resuming a connection (REQUEST=CONNECTION with REQTYPE=PAUSE or REQTYPE=RESUME), the requester must satisfy at least one of the following conditions:</p> <ul style="list-style-type: none"> <li>Requester is the connector</li> <li>Home is the connector address space</li> <li>Primary is the connector address space</li> <li>Running in supervisor state or with a PKM allowing key 0-7</li> </ul> <p><b>Action:</b> Verify that the caller is a valid user of the connection.</p>
8	xxxx001E	<p><b>Equate Symbol:</b> IXCNOTERSNCONNECTIONBADACCESS</p> <p><b>Meaning:</b> Program error. The specified CONNECTION token is not valid for use with the specified request.</p> <p>If the connection was created with ACCESS=READ, one cannot use the token to create, write, replace, or delete notes.</p> <p>If an answer area was provided, the field AA_DETAILS provides additional information. The data is mapped by IXCYNODE_TDETAILSACCESS.</p> <p><b>Action:</b> Verify that the connection token has the right type of access for the request.</p>
8	xxxx001F	<p><b>Equate Symbol:</b> IXCNOTERSNCONNECTIONBADAUTH</p> <p><b>Meaning:</b> Program error.</p> <p>When creating a connection with USAGE=SERVER or USAGE=CLIENT, the program was not running in supervisor state or with a PKM allowing key 0-7.</p> <p>For any other request, the specified connection was created when SAF was not available or when there was no IXCNOTE.owner.application resource defined for the note pad in the FACILITY class. It can only be used by programs running in supervisor state or with a PKM allowing key 0-7.</p> <p><b>Action:</b> Verify that a proper SAF profile was set up for the note pad resources, and that the caller has the proper authorization to issue the request.</p>

Table 14. Return and Reason Codes for the IXCNOTE Macro (continued)		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0020	<p><b>Equate Symbol:</b> IXCNOTERSNDESCBADSTG</p> <p><b>Meaning:</b> Program error. The storage identified by the DESCRIPTION keyword is not accessible. It must be both addressable and accessible using the PSW key of the caller.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• Ensure that the correct storage area was used.</li> <li>• If your program is running in AR mode, ensure that: <ul style="list-style-type: none"> <li>– The ALET for the storage area is correct.</li> <li>– You specified SYSSTATE ASCENV=AR before issuing the request.</li> </ul> </li> <li>• Ensure that the storage area was not inadvertently freed by your program.</li> </ul>
8	xxxx0021	<p><b>Equate Symbol:</b> IXCNOTERSNDESCBADALET</p> <p><b>Meaning:</b> Program error. The ALET that qualifies the address of the storage area indicated by the DESCRIPTION keyword is neither zero, nor a valid entry on the Dispatchable Unit Access List (DU-AL) of the caller, nor a valid entry for a common area data space.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• If your program is running in AR mode, ensure that you specified SYSSTATE ASCENV=AR before issuing the request.</li> <li>• Ensure that the ALET for the storage area is correct.</li> </ul>
8	xxxx0022	<p><b>Equate Symbol:</b> IXCNOTERSNDESCBADVAL</p> <p><b>Meaning:</b> Program error. The storage identified by the DESCRIPTION keyword does not contain a valid description. Descriptions can contain any alphanumeric (A-Z, a-z, 0-9), national (@, #, \$), or special (underscore or blank) character. Leading blanks and all blank descriptors are not permitted.</p> <p><b>Action:</b> Verify that the DESCRIPTION meets the requirements of the keyword.</p>
8	xxxx0023	<p><b>Equate Symbol:</b> IXCNOTERSNINFOBADSTG</p> <p><b>Meaning:</b> Program error. The storage identified by the INFO keyword is not accessible. It must be both addressable and accessible using the PSW key of the caller.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• Ensure that the correct storage area was used.</li> <li>• If your program is running in AR mode, ensure that: <ul style="list-style-type: none"> <li>– The ALET for the storage area is correct.</li> <li>– You specified SYSSTATE ASCENV=AR before issuing the request.</li> </ul> </li> <li>• Ensure that the storage area was not inadvertently freed by your program.</li> </ul>
8	xxxx0024	<p><b>Equate Symbol:</b> IXCNOTERSNINFOBADALET</p> <p><b>Meaning:</b> Program error. The ALET that qualifies the address of the storage area indicated by the INFO keyword is neither zero, nor a valid entry on the Dispatchable Unit Access List (DU-AL) of the caller, nor a valid entry for a common area data space.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• If your program is running in AR mode, ensure that you specified SYSSTATE ASCENV=AR before issuing the request.</li> <li>• Ensure that the ALET for the storage area is correct.</li> </ul>

Table 14. Return and Reason Codes for the IXCNOTE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0025	<p><b>Equate Symbol:</b> IXCNOTERSNCRITERIABADSTG</p> <p><b>Meaning:</b> Program error. The storage identified by the CRITERIA keyword is not accessible. It must be both addressable and accessible using the PSW key of the caller. In the case of a delete request, zero or more notes might have been deleted before the problem was detected.</p> <p>If an answer area was provided, the field AA_DETAILS provides additional information to identify the problem. The data is mapped by IXCYNODE_TDETAILSCRITERIA.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• Ensure that the correct storage area was used.</li> <li>• If your program is running in AR mode, ensure that: <ul style="list-style-type: none"> <li>– The ALET for the storage area is correct.</li> <li>– You specified SYSSTATE ASCENV=AR before issuing the request.</li> </ul> </li> <li>• Ensure that the storage area was not inadvertently freed by your program.</li> </ul>
8	xxxx0026	<p><b>Equate Symbol:</b> IXCNOTERSNCRITERIABADALET</p> <p><b>Meaning:</b> Program error. The ALET that qualifies the address of the storage area indicated by the CRITERIA keyword is neither zero, nor a valid entry on the Dispatchable Unit Access List (DU-AL) of the caller, nor a valid entry for a common area data space.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• If your program is running in AR mode, ensure that you specified SYSSTATE ASCENV=AR before issuing the request.</li> <li>• Ensure that the ALET for the storage area is correct.</li> </ul>
8	xxxx0027	<p><b>Equate Symbol:</b> IXCNOTERSNCRITERIABADVAL</p> <p><b>Meaning:</b> Program error. The specified CRITERIA do not contain valid selection criteria. In the case of a delete request, zero or more notes might have been deleted before the problem was detected.</p> <p>If an answer area was provided, the field AA_DETAILS provides additional information to identify the problem. The data is mapped by IXCYNODE_TDETAILSCRITERIA.</p> <p><b>Action:</b> Verify that the CRITERIA meets the requirements of the keyword.</p>
8	xxxx0029	<p><b>Equate Symbol:</b> IXCNOTERSNBUFFERBADSTGNP</p> <p><b>Meaning:</b> Program error. The storage identified by the BUFFER keyword is not accessible. It must be both addressable and accessible using the PSW key of the caller. The note pad was not altered. XCF might have altered the content of the storage containing the buffer and the answer area.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• Ensure that the correct storage area was used.</li> <li>• If your program is running in AR mode, ensure that: <ul style="list-style-type: none"> <li>– The ALET for the storage area is correct.</li> <li>– You specified SYSSTATE ASCENV=AR before issuing the request.</li> </ul> </li> <li>• Ensure that the storage area was not inadvertently freed by your program.</li> </ul>
8	xxxx002A	<p><b>Equate Symbol:</b> IXCNOTERSNBUFFERBADALET</p> <p><b>Meaning:</b> Program error. The ALET that qualifies the address of the storage area indicated by the BUFFER keyword is neither zero, nor a valid entry on the Dispatchable Unit Access List (DU-AL) of the caller, nor a valid entry for a common area data space.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• If your program is running in AR mode, ensure that you specified SYSSTATE ASCENV=AR before issuing the request.</li> <li>• Ensure that the ALET for the storage area is correct.</li> </ul>

Table 14. Return and Reason Codes for the IXCNOTE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx002B	<p><b>Equate Symbol:</b> IXCNOTERSNBUFLNBADVAL</p> <p><b>Meaning:</b> Program error. The value specified by the BUFLN keyword is not valid. BUFLN must be a multiple of 1024. If an answer area was provided, the field AA_DETAILS provides additional information. The data is mapped by IXCYNOTE_TDETAILSBUFLN.</p> <p>For REQUEST=NOTE with REQTYPE of READ or DELETE, BUFLN must be greater than or equal to the size of the designated note. For REQUEST=NOTE with REQTYPE of CREATE, WRITE, or REPLACE, BUFLN must be less than or equal to the size of the largest note supported by the note pad.</p> <p>When reading multiple notes with REQUEST=NOTES REQTYPE=READ, BUFLN must be greater than or equal to the length in bytes of the largest note, or at least the first note, selected.</p> <p><b>Action:</b> Verify that the BUFLN meets the requirements of the keyword.</p>
8	xxxx002C	<p><b>Equate Symbol:</b> IXCNOTERSNBUFFERBADSTG</p> <p><b>Meaning:</b> Program error. The storage identified by the BUFFER keyword is not accessible. It must be both addressable and accessible using the PSW key of the caller. The error occurred while trying to store note data in the buffer. The request might or might not have completed successfully. In the particular case of a delete note request (IXCNOTE REQUEST=NOTE REQTYPE=DELETE), the note was successfully deleted.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that the correct buffer storage area was specified, and the storage area was not inadvertently freed.</li> <li>• As needed, verify whether the request had been processed using REQTYPE=READ.</li> <li>• Take an appropriate recovery action.</li> </ul>
8	xxxx002E	<p><b>Equate Symbol:</b> IXCNOTERSNRESUMETOKENBADVAL</p> <p><b>Meaning:</b> Program error. The value specified by the RESUMETOKEN keyword is not valid. The value must either be zero or a token that was returned by a previous IXCNOTE request.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• Ensure that the correct resume token was specified.</li> <li>• Ensure that the parameter list was not inadvertently overlaid and that it was correctly specified.</li> </ul>
8	xxxx0030	<p><b>Equate Symbol:</b> IXCNOTERSNANSAREAREQUIRED</p> <p><b>Meaning:</b> Program error. An answer area is required for the particular IXCNOTE request that was issued.</p> <p><b>Action:</b> Reissue the request with a valid answer area.</p>
8	xxxx0031	<p><b>Equate Symbol:</b> IXCNOTERSNANSAREABADSTG</p> <p><b>Meaning:</b> Program error. The storage identified by the ANSAREA keyword is not accessible. It must be both addressable and accessible using the PSW key of the caller. The error occurred while trying to store output data after the IXCNOTE request had been processed. The request might or might not have completed successfully. In the specific case of REQUEST=CONNECTION REQTYPE=CREATE, the connection that was created, if any, has been deleted by XCF.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that the correct storage area was specified, and the storage area was not inadvertently freed.</li> <li>• As needed, verify whether the request had been processed.</li> <li>• Take an appropriate recovery action.</li> </ul>

Table 14. Return and Reason Codes for the IXCNOTE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0032	<p><b>Equate Symbol:</b> IXCNOTERSNANSAREABADALET</p> <p><b>Meaning:</b> Program error. The ALET that qualifies the address of the storage area indicated by the ANSAREA keyword is neither zero, nor a valid entry on the Dispatchable Unit Access List (DU-AL) of the caller, nor a valid entry for a common area data space.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• If your program is running in AR mode, ensure that you specified SYSSTATE ASCENV=AR before issuing the request.</li> <li>• Ensure that the ALET for the storage area is correct.</li> </ul>
8	xxxx0033	<p><b>Equate Symbol:</b> IXCNOTERSNANSLENBADVAL</p> <p><b>Meaning:</b> Program error. The value specified by the ANSLEN keyword is not valid. The length of the answer area must be at least as long as the number of bytes in the answer area header.</p> <p><b>Action:</b> Obtain storage for the answer area that is at least as long as the number of bytes in the answer area header and reissue the request.</p>
8	xxxx0034	<p><b>Equate Symbol:</b> IXCNOTERSNANSLENMORE</p> <p><b>Meaning:</b> The request was not processed because the ANSAREA is too small. The ANSAREA is large enough to hold the header (IXCYNOTE_TANSAREA) but does not have room for the requested data. The AA_ANSAREASIZENEDED returned in the ANSAREA indicates how much storage is required in order for the ANSAREA to hold the requested data. In the case of a REQUEST=NOTES REQTYPE=READ request, AA_ANSAREASIZENEDED will indicate the amount of space needed for the required minimum number of notes.</p> <p><b>Action:</b> Obtain storage for the answer area that is at least as large as the indicated size and reissue the request.</p>
8	xxxx0035	<p><b>Equate Symbol:</b> IXCNOTERSNANSAREABADSTGNP</p> <p><b>Meaning:</b> Program error. The storage identified by the ANSAREA keyword is not accessible. It must be both addressable and accessible using the PSW key of the caller. The note pad was not altered. XCF might have altered the content of the storage containing the answer area, and as applicable, the buffer area.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• Ensure that the correct storage area was used.</li> <li>• If your program is running in AR mode, ensure that: <ul style="list-style-type: none"> <li>– The ALET for the storage area is correct.</li> <li>– You specified SYSSTATE ASCENV=AR before issuing the request.</li> </ul> </li> <li>• Ensure that the storage area was not inadvertently freed by your program.</li> </ul>
8	xxxx0037	<p><b>Equate Symbol:</b> IXCNOTERSN#NOTESEXCEEDED</p> <p><b>Meaning:</b> The note request cannot be processed because the note pad already contains the maximum number of notes.</p> <p>If an answer area was provided, the field AA_DETAILS provides additional information. The data is mapped by IXCYNOTE_TDETAILSCONSTRAINED.</p> <p><b>Action:</b> Try the request again after deleting some notes, or after issuing an IXCNOTE REQUEST=NOTEPAD REQTYPE=MODIFY request that successfully increased the number of notes that the note pad is allowed to hold.</p>
8	xxxx0038	<p><b>Equate Symbol:</b> IXCNOTERSN#NOTESBADVAL</p> <p><b>Meaning:</b> Program error. The value specified by the #NOTES keyword is not valid. It must be a number greater than or equal to one.</p> <p><b>Action:</b> Verify that the #NOTES meets the requirements of the keyword.</p>

Table 14. Return and Reason Codes for the IXCNOTE Macro (continued)		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0040	<p><b>Equate Symbol:</b> IXCNOTERSNNOTENOTEXIST</p> <p><b>Meaning:</b> The specified note does not exist.</p> <p><b>Action:</b> Verify that the input note name was validly specified.</p>
8	xxxx0041	<p><b>Equate Symbol:</b> IXCNOTERSNNOTEEXISTS</p> <p><b>Meaning:</b> An attempt to create a new note (REQUEST=NOTE REQTYPE=CREATE) is rejected because the named note already exists.</p> <p>As applicable, the current tag value of the note was stored in the TAG variable. As applicable, the current instance number for the note was stored in the INSTANCE# variable.</p> <p><b>Action:</b> Verify that the input note name was unique in the note pad.</p>
8	xxxx0042	<p><b>Equate Symbol:</b> IXCNOTERSNNOTEBADINSTANCE#</p> <p><b>Meaning:</b> The specified note was not processed due to an instance number mismatch (INSTANCE#). The current instance number of the note was stored in the INSTANCE# variable. As applicable, the tag value of the note was stored in the TAG variable.</p> <p><b>Action:</b> As needed, retry the request with the updated INSTANCE#.</p>
8	xxxx0043	<p><b>Equate Symbol:</b> IXCNOTERSNNOTEBADTAGGING</p> <p><b>Meaning:</b> Program error. The TAGGING specification does not match the TAGGING specified by the creator of the note pad. If TAGGING=XCF was specified by the creator of the note pad, the connector must also specify TAGGING=XCF when processing notes. Similarly, if TAGGING=USER was specified by the creator of the note pad, the connector must also specify TAGGING=USER.</p> <p><b>Action:</b> Reissue the request with a suitable tagging specification.</p>
8	xxxx0044	<p><b>Equate Symbol:</b> IXCNOTERSNNOTELOWTAG</p> <p><b>Meaning:</b> Program error. The specified tag value was less than the current tag value of the note. The current tag value of the note was stored in the TAG variable. As applicable, the current instance number of the note was stored in the INSTANCE# variable.</p> <p>When the maximum user assigned tag values are being tracked for the note pad (the creator of the note pad specified TAGGING=USER and either TRACKTAG=CURRENT or TRACKTAG=LIFETIME), the tag value to be assigned to an existing note must be greater than or equal to the current tag value of the note. The write, replace, or delete request was rejected because the new tag value (TAG) specified for the note was not greater than or equal to the current tag value of the note.</p> <p><b>Action:</b> As appropriate, reissue the request with a tag value that is greater than the current tag value of the note.</p>
8	xxxx0045	<p><b>Equate Symbol:</b> IXCNOTERSNNOTENOINSTANCE#</p> <p><b>Meaning:</b> Program error. The creator of the note pad indicated that an instance number comparison must be performed (INSTCOMP=REQUIRED) when invoking IXCNOTE REQUEST=NOTE to replace, write, or delete a note. The specified note was not processed because the required instance number comparison value was not provided. Either INSTANCE#=IGNORE was specified, or the value of the INSTANCE# variable was zero.</p> <p><b>Action:</b> As appropriate, obtain the current instance number of the relevant note and try the request again, specifying the nonzero instance number for comparison. The current instance number of a note can be obtained using an IXCNOTE REQUEST=NOTE REQTYPE=READ request with a valid answer area specified.</p>

Table 14. Return and Reason Codes for the IXCNOTE Macro (continued)		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx00A4	<p><b>Equate Symbol:</b> IXCNOTERSNROCONFLICT</p> <p><b>Meaning:</b> Program error. The request data in the parameter list is not consistent with the request data in register 0 on entry to the IXCNOTE service routine. The parameter list or the register contents were corrupted before the service routine received control.</p> <p><b>Action:</b> Check to see if your program inadvertently overlaid the parameter list storage, and that it was assembled with the correct macro library for the release of z/OS on which your program is running.</p>
8	xxxx0128	<p><b>Equate Symbol:</b> IXCNOTERSNTASKTERM</p> <p><b>Meaning:</b> Program error. The specified request is not valid when called from a task that is terminating. Furthermore, when creating a connection to a note pad, TERMSCOPE cannot be assigned to a task that is terminating.</p> <p><b>Action:</b> As appropriate, reissue the request from a task that is not terminating. If applicable, verify that the TERMSCOPE meets the requirements of the keyword.</p>
8	xxxx0129	<p><b>Equate Symbol:</b> IXCNOTERSNSPACETERM</p> <p><b>Meaning:</b> Program error. The specified request is not valid when called from an address space that is terminating. Furthermore, when creating a connection to a note pad, TERMSCOPE cannot be assigned to an address space that is terminating.</p> <p><b>Action:</b> As appropriate, reissue the request from an address space that is not terminating. If applicable, verify that the TERMSCOPE meets the requirements of the keyword.</p>
8	xxxx012A	<p><b>Equate Symbol:</b> IXCNOTERSNRESMGR</p> <p><b>Meaning:</b> Program error. The specified request is not valid when called by a program running as a resource manager.</p> <p><b>Action:</b> As appropriate, reissue the request from a program that is not running as a resource manager.</p>
8	xxxx0130	<p><b>Equate Symbol:</b> IXCNOTERSNBADSENV</p> <p><b>Meaning:</b> Program error. When creating a connection with USAGE=SERVER, a program running in task mode must not have a task specific security environment. That is, TCBSENV must be zero.</p> <p><b>Action:</b> Verify that the program has the proper security environment set up.</p>
8	xxxx0801	<p><b>Equate Symbol:</b> IXCNOTERSNPLISTBADSTGNP</p> <p><b>Meaning:</b> Program error. The storage containing the parameter list is not accessible. It must be both addressable and accessible using the PSW key of the caller.</p> <p>The note pad was not altered.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• Ensure that the correct storage area was used.</li> <li>• If your program is running in AR mode, ensure that: <ul style="list-style-type: none"> <li>– The ALET for the storage area is correct.</li> <li>– You specified SYSSTATE ASCENV=AR before issuing the request.</li> </ul> </li> <li>• Ensure that the storage area was not inadvertently freed by your program.</li> </ul>
8	xxxx0802	<p><b>Equate Symbol:</b> IXCNOTERSNPLISTBADALETNP</p> <p><b>Meaning:</b> Program error. The parameter list ALET is not valid. The ALET that qualifies the address of the parameter list is neither zero nor a valid entry on the caller's Dispatchable Unit Access List (DU-AL), nor a valid entry for a common area data space.</p> <p>The IXCNOTE request was not processed.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• If your program is running in AR mode, ensure that you specified SYSSTATE ASCENV=AR before issuing the request.</li> <li>• Ensure that the ALET for the storage area is correct.</li> </ul>

Table 14. Return and Reason Codes for the IXCNOTE Macro (continued)		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0804	<p><b>Equate Symbol:</b> IXCNOTERSNNOTINSTALLEDVN</p> <p><b>Meaning:</b> Program error.</p> <p>This reason is returned only if the program issues an IXCNOTE request while running on a system that does not have IXCNOTE support installed. Note that the down level system can reject an IXCNOTE request with a variety of failures. This particular reason applies when the down level system determines that the parameter list version number is not one that it supports.</p> <p><b>Action:</b> As appropriate, verify that IXCNOTE support is properly installed and configured on the system, or retry the request on a system that does have IXCNOTE support.</p>
8	xxxx0805	<p><b>Equate Symbol:</b> IXCNOTERSNTASKTOOHIGH</p> <p><b>Meaning:</b> Program error. IXCNOTE cannot be used by tasks higher in the task tree than the cross memory resource owning task (the top, or first, job step task in the address space).</p> <p><b>Action:</b> Verify that the work unit issuing the IXCNOTE request meets the environmental restrictions.</p>
8	xxxx0806	<p><b>Equate Symbol:</b> IXCNOTERSNNOTTASKMODE</p> <p><b>Meaning:</b> Program error. The specified request is only valid when called from a work unit that is running in task mode.</p> <p>This reason can also be issued if the program is running in SRB mode on a system that does not have IXCNOTE support installed.</p> <p><b>Action:</b> Ensure that macro usage meets environmental requirements.</p>
8	xxxx0807	<p><b>Equate Symbol:</b> IXCNOTERSNNOTENABLED</p> <p><b>Meaning:</b> Program error. Caller is not running enabled.</p> <p><b>Action:</b> Ensure that macro usage meets environmental requirements.</p>
8	xxxx0808	<p><b>Equate Symbol:</b> IXCNOTERSNMASTERAS</p> <p><b>Meaning:</b> Program error. This reason is issued only if the program issues an IXCNOTE request from the MASTER address space while running on a system that does not have IXCNOTE support installed.</p> <p><b>Action:</b> Ensure that macro usage meets environmental requirements.</p>
8	xxxx0809	<p><b>Equate Symbol:</b> IXCNOTERSNPRIMARYNOTHOME</p> <p><b>Meaning:</b> Program error. The home address space of the caller is not the same as the primary address space.</p> <p>Problem state programs with a PKM allowing key 8-15 must be running with home and primary being the same address space for all REQUEST=NOTEPAD requests, and for REQUEST=CONNECTION requests with REQTYPE=CREATE.</p> <p>Authorized programs must be running with home and primary being the same address space when creating a connection with USAGE=SERVER or USAGE=CLIENT.</p> <p><b>Action:</b> Ensure that macro usage meets environmental requirements.</p>
8	xxxx0812	<p><b>Equate Symbol:</b> IXCNOTERSNBADSUSPENDENV</p> <p><b>Meaning:</b> Program error. The IXCNOTE request was issued from a SUSPEND exit routine or from an SRB routine that the system abended with a 47B system completion code. The caller cannot be suspended while running in this environment. The request cannot be processed because XCF needs to suspend the caller.</p> <p><b>Action:</b> Ensure that macro usage meets environmental requirements.</p>



Table 14. Return and Reason Codes for the IXCNOTE Macro (continued)		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx084C	<p><b>Equate Symbol:</b> IXCNOTERSNNOSAFAUTH</p> <p><b>Meaning:</b> The caller does not have the SAF authorization needed to perform the request.</p> <p>If an answer area was provided, the field AA_DETAILS provides additional information. The data is mapped by IXCYNOTE_TDETAILSNOSAFAUTH.</p> <p><b>Action:</b> Ensure that appropriate SAF profiles are set up for the note pad resources, and that the unauthorized application has the proper SAF authorization needed to perform the request.</p>
8	xxxx0857	<p><b>Equate Symbol:</b> IXCNOTERSNHASFRR</p> <p><b>Meaning:</b> Program error. Caller is running with an FRR established. This reason applies only to task mode callers.</p> <p><b>Action:</b> Ensure that macro usage meets environmental requirements.</p>
8	xxxx0876	<p><b>Equate Symbol:</b> IXCNOTERSNLOCKED</p> <p><b>Meaning:</b> Program error. Caller holds locks.</p> <p>Note that for some cases, an X'073' abend might be issued instead.</p> <p><b>Action:</b> Ensure that macro usage meets environmental requirements.</p>
8	xxxx08B2	<p><b>Equate Symbol:</b> IXCNOTERSNBADSERVICENUM</p> <p><b>Meaning:</b> Program error. The parameter list service code value is not valid. Either the parameter list was corrupted or the program is running on a system that does not have IXCNOTE support installed.</p> <p><b>Action:</b> Check to see if your program inadvertently overlaid the parameter list storage, and that it was assembled with the correct macro library for the release of z/OS on which your program is running.</p>
C	xxxx0C01	<p><b>Equate Symbol:</b> IXCNOTERSNQUIESCED</p> <p><b>Meaning:</b> The note pad is quiesced. Requests to process notes in the note pad (REQUEST=NOTE or REQUEST=NOTES) are rejected when a note pad is quiesced.</p> <p>A note pad could be quiesced for a variety of reasons. For example, a note pad will be quiesced if the CF structure containing the note pad is being rebuilt. It will also be quiesced when a system loses connectivity to the coupling facility that contains the note pad.</p> <p>If an answer area was provided, the field AA_DETAILS provides additional information. The data is mapped by IXCYNOTE_TDETAILSQUIESCED.</p> <p><b>Action:</b> Either invoke IXCNOTE REQUEST=CONNECTION REQTYPE=PAUSE to pause until the note pad is no longer quiesced or retry the request after allowing time for the condition to clear.</p>
C	xxxx0C02	<p><b>Equate Symbol:</b> IXCNOTERSNCONSTRAINED</p> <p><b>Meaning:</b> XCF is unable to honor the request due to constraints on the available number of notes.</p> <p>When processing a note, XCF is unable to satisfy the request because the CF structure is full. XCF is unable to provide the number of notes requested by the creator of the note pad.</p> <p>If an answer area was provided, the field AA_DETAILS provides additional information. The data is mapped by IXCYNOTE_TDETAILSCONSTRAINED.</p> <p><b>Action:</b> Try the request again after allowing time for the condition to clear.</p>

Table 14. Return and Reason Codes for the IXCNOTE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C03	<p><b>Equate Symbol:</b> IXCNOTERSNMODIFYINPROGRESS</p> <p><b>Meaning:</b> A REQUEST=NOTEPAD REQTYPE=MODIFY to change the #NOTES value for a note pad is rejected because a MODIFY request to change the #NOTES attribute is still in progress.</p> <p>A request is either currently in progress updating the note pad #NOTES attribute and CF structure control information or a previous MODIFY request is pending completion due to an inability to access the note pad catalog structure or note pad structure or both.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>When a connection to the note pad exists, consider using IXCNOTE REQUEST=CONNECTION REQTYPE=PAUSE to be resumed when XCF completes the MODIFY request.</li> <li>IXCNOTE REQUEST=NOTEPAD REQTYPE=QUERY can also be used to determine whether a MODIFY request is pending against the note pad. npd_Modifying#Notes is set ON if the note pad #NOTES is in the midst of being modified. When npd_Modifying#Notes is set OFF, npd_Required#Notes reflects the number of notes that the note pad is allowed to hold. npd_Required#Notes will not reflect the requested change to the #NOTES while the modify request is in progress</li> </ul>
C	xxxx0C0A	<p><b>Equate Symbol:</b> IXCNOTERSNNO#NOTESSTRRESOURCES</p> <p><b>Meaning:</b> The requested #NOTES cannot be committed to the note pad because the current number of data entries reserved for XCF and for note pads in the host note pad structure plus the requested #Notes would exceed the maximum entries supported by the structure.</p> <p><b>Action:</b> Retry the request after allowing time for the condition to clear. In some cases, manual intervention might be required to resolve the problem. For example, the installation might need to update the CFRM policy to change the size of the host note pad structure or use the IXLALTER service to expand the size of the structure. In other cases, the typical dynamics of the system could cause existing note pads to be deleted or note pad notes to be deleted which in turn would allow an increase in #NOTES for the note pad.</p> <p>If an answer area was provided, the field aa_Details provides additional information. The data is mapped by ixcyntote_tDetailsModify#Notes.</p>
C	xxxx0C0B	<p><b>Equate Symbol:</b> IXCNOTERSNNOTSUPPORTMODIFY</p> <p><b>Meaning:</b> A system that has connectors to the note pad does not support IXCNOTE REQUEST=NOTEPAD, REQTYPE=MODIFY. All systems with connectors to the note pad must support REQTYPE=MODIFY to be able to modify the #NOTES definition for a note pad.</p> <p><b>Action:</b> Connectors on a system that does not support REQTYPE=MODIFY should temporarily disconnect from the note pad until the MODIFY #Notes request completes. When using REQUEST=NOTEPAD, REQTYPE=MODIFY to modify the #Notes that a note pad is allowed to hold, IBM recommends that if connections to the note pad exist, then all connections to the note pad be from systems that support REQUEST=NOTEPAD, REQTYPE=MODIFY. To determine whether the support for REQUEST=NOTEPAD, REQTYPE=MODIFY is available on the system from which you are connecting, issue IXCQUERY REQINFO=FEATURES. QuReqRfIxcNoteResiliency indicates whether the support is available.</p>
C	xxxx0C0C	<p><b>Equate Symbol:</b> IXCNOTERSNDECREASENOTVALID</p> <p><b>Meaning:</b> The requested #NOTES cannot be less than the current number of notes in use in the note pad.</p> <p><b>Action:</b> Delete enough notes from the note pad such that the number of notes in use by the note pad is less than or equal to the number of notes being requested, then retry the MODIFY request to decrease the #Notes allowed for a note pad</p> <p>If an answer area was provided, the field aa_Details provides additional information. The data is mapped by ixcyntote_tDetailsModify#Notes.</p>

Table 14. Return and Reason Codes for the IXCNOTE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C0D	<p><b>Equate Symbol:</b> IXCNOTERSNNOTEPADNOTUSABLE</p> <p><b>Meaning:</b> The note pad is no longer usable by the local system receiving this reason code due to an enabled note pad feature(s) that is not compatible with the level of note pad support on the local system. i.e., the current instance of the note pad contains features or exploitation of function that are not understood by the local system.</p> <p><b>Action:</b> Restoring access to the note pad from the local system will require a re-IPL of the local system or deleting and re-creating the note pad.</p>
C	xxxx0C40	<p><b>Equate Symbol:</b> IXCNOTERSNNNOSECPROFILE</p> <p><b>Meaning:</b> No security decision could be made for an unauthorized caller since there is either no security product installed on the system, or there is no IXCNOTE.<i>owner.application</i> resource defined for the note pad in the FACILITY class. A request to create, delete, query, or connect to a note pad is rejected since the program is running in problem state with a PKM allowing key 8-15.</p> <p>Note that this problem can occur if the installation defines the necessary profile but then fails to refresh the security profile. For example, if RACF is being used as the security product, the installation might need to issue RACLIST(REFRESH) to get the system to recognize the update to the resource profile.</p> <p><b>Action:</b> As appropriate, ensure that proper security profiles are set up for the note pad resources.</p>
C	xxxx0CA3	<p><b>Equate Symbol:</b> IXCNOTERSNNMAXNOTEPADS</p> <p><b>Meaning:</b> Unable to create a new note pad. Maximum number of note pads already defined.</p> <p>The maximum number of note pads is determined by the size of the structure that XCF uses to manage note pads (SYSXCF_NPCATALOG).</p> <p><b>Action:</b> Instruct the system programmer to increase the size of the SYSXCF_NPCATALOG structure and try the request again.</p>
C	xxxx0CA4	<p><b>Equate Symbol:</b> IXCNOTERSNNOSYSRESOURCES</p> <p><b>Meaning:</b> Unable to obtain the system resources needed to process the request.</p> <p>If an answer area was provided, the field AA_DETAILS provides additional information. The data is mapped by IXCYNODE_TDETAILSNORESOURCES.</p> <p><b>Action:</b> Try the request again after allowing time for the condition to clear.</p>
C	xxxx0CA5	<p><b>Equate Symbol:</b> IXCNOTERSNNOSTRRESOURCES</p> <p><b>Meaning:</b> Unable to create a new note pad because there was no coupling facility structure suitable for it.</p> <p>If an answer area was provided, the field AA_DETAILS provides additional information. The data is mapped by IXCYNODE_TDETAILSNOSTRSTRUCTURES.</p> <p><b>Action:</b> Try the request again after allowing time for the condition to clear. In some cases, manual intervention might be required to resolve the problem. For example, the installation might need to update the CFRM policy to define additional structures or change the sizes of existing structures, or it might be necessary to reestablish connectivity to a coupling facility. In other cases, the typical dynamics of the system could cause existing note pads to be deleted, which in turn would allow a new note pad to be created. Review the data returned in the answer area for additional information, if an answer area was provided.</p>

Table 14. Return and Reason Codes for the IXCNOTE Macro (continued)		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0CA6	<p><b>Equate Symbol:</b> IXCNOTERSNMAXCONNECTIONS</p> <p><b>Meaning:</b> Unable to create a new connection to the requested note pad. The maximum number of connections created for the home address space have already been defined.</p> <p><b>Action:</b> Try the request again after allowing time for the condition to clear. If you still receive this code, try any of the following alternate solutions:</p> <ul style="list-style-type: none"> <li>• Submit a requirement to increase the limit of max connections, which is an internal limit imposed by XCF.</li> <li>• Rework the application to use fewer connections.</li> <li>• Obtain another address space.</li> </ul>
C	xxxx0CB0	<p><b>Equate Symbol:</b> IXCNOTERSNTIMEOUT</p> <p><b>Meaning:</b> The request timed out. No recovery action is required. The request can be retried if so desired.</p> <p><b>Note:</b> XCF will instead return status unknown (IXCNOTERSNSTATUSUNKNOWN) if you might need to perform a recovery action before the request can be retried.</p> <p><b>Action:</b> Try the request again.</p>
C	xxxx0CBD	<p><b>Equate Symbol:</b> IXCNOTERSNSTATUSUNKNOWN</p> <p><b>Meaning:</b> The request failed in a way that XCF cannot determine whether it actually worked or not. This result can occur, for example, when the local system loses connectivity to the coupling facility that contains the note pad. It can also occur if the request times out (see <a href="#">TIMEOUT</a> keyword for <a href="#">REQUEST=NOTEPAD</a> and <a href="#">TIMEOUT</a> keyword for <a href="#">REQUEST=CONNECTION</a>).</p> <p>When processing a note, the note might or might not have been successfully created, written, replaced, or deleted from the note pad.</p> <p>When deleting a note pad, the note pad might or might not have been marked for deletion.</p> <p><b>Action:</b> Take an appropriate recovery action.</p> <p>When processing a note, for example, you might issue a read request to determine whether the note exists in the note pad. As appropriate, you might then create, replace, or delete the note to achieve the desired result. In some cases you might not be able to issue such requests until the note pad is once again accessible (see <a href="#">IXCNOTE REQUEST=CONNECTION REQTYPE=PAUSE</a>).</p> <p>When deleting a note pad, for example, you might reissue the request to try deleting the note pad again. You might need to be sensitive to the particular instance of the note pad (see <a href="#">ETODCREATED</a> keyword).</p> <p>When modifying the #NOTES attribute for a note pad, processing to update the #NOTES attribute might or might not have been initiated and / or completed. The requester should take an appropriate recovery action. For example, prior to reissuing the request, the requestor might consider issuing IXCNOTE REQUEST=NOTEPAD REQTYPE=QUERY to determine whether a MODIFY request is pending against the note pad</p>
C	xxxx0CFE	<p><b>Equate Symbol:</b> IXCNOTERSNNOTCONFIGURED</p> <p><b>Meaning:</b> The system is not currently configured to support note pads. For example, the current active Coupling Facility Resource Manager (CFRM) policy does not have definitions for the necessary structures. The problem could be resolved if the installation performs the necessary actions.</p> <p>If an answer area was provided, the field AA_DETAILS provides additional information. The data is mapped by IXCYNOTE_TDETAILSNORESOURCES.</p> <p><b>Action:</b> Verify that the system has the necessary resources properly defined and configured for the note pad. Review the data returned in the answer area, if one was provided, for additional details.</p>

Table 14. Return and Reason Codes for the IXCNOTE Macro (continued)		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0CFF	<p><b>Equate Symbol:</b> IXCNOTERSNNOSERVICE</p> <p><b>Meaning:</b> IXCNOTE services are not available on this system. The situation will persist for the life of the IPL.</p> <p>For example, the hardware needed to process coupling facility requests is not present.</p> <p><b>Action:</b> If possible, reissue the request on a different system that has the IXCNOTE support installed.</p>
10	xxxxxxx	<p><b>Meaning:</b> XCF failure. Reason codes are provided for this return code, however they are not documented. In general, XCF will have produced diagnostic reports to document the failure (such as a LOGREC entry or a dump). Any such documentation should be forwarded to IBM for analysis.</p> <p>The reason code should be included in any diagnostic reports that the IXCNOTE user might choose to produce to document the impact from the exploiter perspective.</p> <p><b>Action:</b> Retry the request one or more times. If the problem persists, record the return and reason codes and supply them to the appropriate IBM support personnel.</p>

## Examples

*Operation:* Create a note pad with the name MYOWNER MYAPP1. The note pad can hold up to 10 notes and can receive updates from multiple connectors simultaneously. Register 2 points to the area where XCF will return answer area information. XCF will store the return code and reason code into the variables RETURN and REASON. The code for this operation is as follows:

```

      LA      R2,MYAA      LOAD ADDRESS OF ANSWER AREA
IXCNOTE REQUEST=NOTEPAD,NOTEPAD=MYNP,REQTYPE=CREATE,      X
        DESCRIPTION=MYDESC,#NOTES=NOTENUM,MULTIWRITE=YES  X
        ANSAREA=(R2),ANSLEN=MYAALEN,                      X
        RETCODE=RETURN,RSNCODE=REASON,MF=S

MYNP    DC    CL32'MYOWNER MYAPP1'      ' NOTE PAD NAME
MYDESC  DC    CL32'NOTE PAD USED FOR MYOWNER MYAPP1'    DESCRIPTION
NOTENUM DC    F'10'      NUMBER OF NOTES
MYAA    DS    CL1024      ANSWER AREA TO CONTAIN DATA RETURNED      X
                        BY IXCNOTE
MYAALEN DC    F'1024'    LENGTH OF THE ANSWER AREA
RETURN  DS    1F        RETURN CODE
REASON  DS    1F        REASON CODE

```



## Chapter 18. IXCQUERY – Obtain XCF Information

### Description

The IXCQUERY macro allows any authorized caller to request information about the resources the cross-system coupling facility (XCF) manages. The REQINFO parameter determines whether the information is about XCF groups, systems in the sysplex, the sysplex itself, coupling facility resources, or information related to the automatic restart manager.

- REQINFO=GROUP returns information about groups and members defined to XCF. The information can be about all groups, a specific group, or a single member of a group. If you issue IXCQUERY without any parameters, you receive general information about each group defined to XCF.
- REQINFO=SYSPLEX returns information about each system in the sysplex.
- REQINFO=CF returns information about coupling facilities defined in the CFRM active policy. The information can be about all coupling facilities or a specific coupling facility.
- REQINFO=CF\_ALLDATA returns information about all coupling facilities.
- REQINFO=STR returns information about coupling facility structures defined in the CFRM active policy. The information can be about all structures or a specific structure.
- REQINFO=STR\_ALLDATA returns information about all coupling facility structures.
- REQINFO=COUPLE returns information about the sysplex. The scope of the output depends on the other parameters specified.
- REQINFO=ARMSTATUS returns information about active elements. The scope of the output depends on the other parameters specified.
- REQINFO=ARMS\_ALLDATA returns information about all active elements.
- REQINFO=FEATURES returns information about the XCF and XES software features installed on the system from which the request is made.
- REQINFO=CDS returns general couple data set information on all functions or detailed couple data set information on a specific function type.
- REQINFO=CDS\_ALLDATA returns detailed couple data set information for all functions.

When you use REQINFO with all options except COUPLE and FEATURES, use the ANSAREA parameter to tell XCF where to return the information, and ANSLEN to tell XCF the length of the answer area. REQINFO=COUPLE and REQINFO=FEATURES generate an inline expansion.

Sections in the IXCYQUAA mapping macro provide the format for the data:

- QUACDS maps the QUAA data for couple data sets.
- QUACDSFUN maps the QUAA data for sysplex functions using couple data sets.
- QUACDSNAR maps the couple data set narrative data.
- QUACDSSU maps the QUAA data for systems using the specified couple data set function.
- QUAHDR maps the offset and length of the other record types.
- QUAGRP maps the group record.
- QUAMEM, QUAMEM1, and QUAMEM2 map the member records.
- QUASYS and QUASYS1 map system information records.
- QUACF and QUACF1 map coupling facility records.
- QUACFSC and QUACFSC1 map system connectivity to coupling facility records.
- QUACFSTR and QUACFSTR1 map information about coupling facility structures allocated in a coupling facility.
- QUASTR and QUASTR1 map the coupling facility structure record.

- QUASTRPL and QUASTRPL1 map the coupling facility structure preference list records.
- QUASTRXL and QUASTRXL1 map the coupling facility structure exclusion list records.
- QUASTRCF and QUASTRCF1 map information about the coupling facility containing a coupling facility structure.
- QUASTRUSER and QUASTRUSER1 map the connectors to coupling facility structure records.
- QUASTRSYS maps information about system participation in a system-managed process for a coupling facility structure.
- QUAARMS maps information about automatic restart manager elements.
- QUREQFEATURES maps information about software features installed on a system.

## Environment

The following are the environment requirements for the caller.

Environment	Environment requirement
Minimum authorization:	Supervisor state or PKM allowing key 0 - 7 For REQINFO=COUPLE and REQINFO=FEATURES, problem state with any key is supported.
Dispatchable unit mode:	Task The following request types allow both task and SRB mode: <ul style="list-style-type: none"> <li>• REQINFO=GROUP,REQTYPE=IMMEDIATE</li> <li>• REQINFO=SYSPLEX</li> <li>• REQINFO=COUPLE</li> <li>• REQINFO=FEATURES</li> </ul>
Cross memory mode:	Any PASN, any HASN, any SASN For REQINFO=CDS or REQINFO=CDS_ALLDATA, to obtain policy information, the primary, and home must be the same address space.
AMODE:	31-bit
ASC mode:	Primary or access register (AR) If in Access Register ASC mode, specify SYSSTATE ASCENV=AR before invoking this macro.
Interrupt status:	Enabled for I/O and external interrupts. The request types REQINFO=COUPLE and REQINFO=FEATURES support disablement.
Locks:	No locks held. The request types REQINFO=COUPLE and REQINFO=FEATURES allow any lock to be held.
Control parameters:	Control parameters must be in the primary address space or be in an address/data space that is addressable through a public entry on the caller's dispatchable unit access list (DU-AL). IXCQUERY with REQINFO=COUPLE and with REQINFO=FEATURES generates an inline expansion and does not require a control parameter list.



## Programming Requirements

---

If the program is in AR mode, issue SYSSTATE ASCENV=AR before IXCQUERY. SYSSTATE ASCENV=AR tells the system to generate code appropriate for AR mode.

If you use REQINFO= any option except COUPLE, include the IXCYQUAA macro in the code to map the data returned in ANSAREA or FEATAREA.

If you use REQINFO=CF, REQINFO=CF\_ALLDATA, STR, or STR\_ALLDATA, you need to include the IXLYNDE macro in the code to map the hardware identification information associated with a coupling facility.

## Restrictions

---

When you issue IXCQUERY REQINFO=GROUP, REQTYPE=DEFER (the default), XCF suspends the task while it serializes and accesses the most recent data.

The following request types are the only ones which allow enabled unlocked task (EUT) FRRs to be established:

- REQINFO=CF
- REQINFO=CF\_ALLDATA
- REQINFO=STR
- REQINFO=STR\_ALLDATA

**Note:** When issuing one of the above REQINFO types, XCF suspends the task while it serializes and accesses the most recent data. For all other request types, the caller cannot have any EUT FRRs established.

You cannot code REQINFO=COUPLE and REQINFO=FEATURES on the execute form of IXCQUERY.

## Input Register Information

---

Before issuing the IXCQUERY macro, the caller does not have to place any information into any register unless using it in register notation for a particular parameter, or using it as a base register.

## Output Register Information

---

When control returns to the caller of the IXCQUERY macro, the general purpose registers (GPRs) contain:

### Register

#### Contents

**0**

Reason code, if applicable, and if GPR15 return code is nonzero

**1**

Used as a work register by the system

**2-13**

Unchanged

**14**

Used as a work register by the system

**15**

Return code

When control returns to the caller of the IXCQUERY macro, the access registers (ARs) contain:

### Register

#### Contents

**0-1**

Used as work registers by the system

### 2-13

Unchanged

### 14-15

Used as work registers by the system

**For registers that the system changes**, a caller who depends on these registers that contain the same value before and after issuing the macro must save these registers before issuing the macro and then restore them after the system returns control.

IXCQUERY with REQINFO=COUPLE and with REQINFO=FEATURES generates an inline expansion; therefore, it does not use a control parameter list, does not produce a return code, and does not preserve AR/GPRs 0, 1, 14, or 15 across the expansion.

## Performance Implications

---

REQTYPE=DEFER (the default), requires that XCF suspend the task while it serializes and accesses the most recent data for the following REQINFO types:

- REQINFO=GROUP
- REQINFO=CF
- REQINFO=CF\_ALLDATA
- REQINFO=STR
- REQINFO=STR\_ALLDATA

IXCQUERY processing might involve referencing or updating the CFRM active policy to reflect the operation that has been requested. When invoking this macro, be aware of the I/O to the CFRM couple data set that might be generated.

## Understanding IXCQUERY Version Support

---

The IXCQUERY macro supports versions in the range of 0 - 2.

- Keywords not specifically noted here are supported by all versions starting with version 0 and higher of the IXCQUERY macro.
- The following keywords and functions are supported by all versions starting with version 1 and higher of the IXCQUERY macro.
  - ELEMENT
  - JOBNAME
  - LOCAL\_ONLY
  - REQINFO=ARMS\_ALLDATA
  - REQINFO=ARMSTATUS
  - RESTARTGRP
  - SYSNAME
- The following keyword is supported by all versions starting with version 2 and higher of the IXCQUERY macro.
  - QUAALEVEL

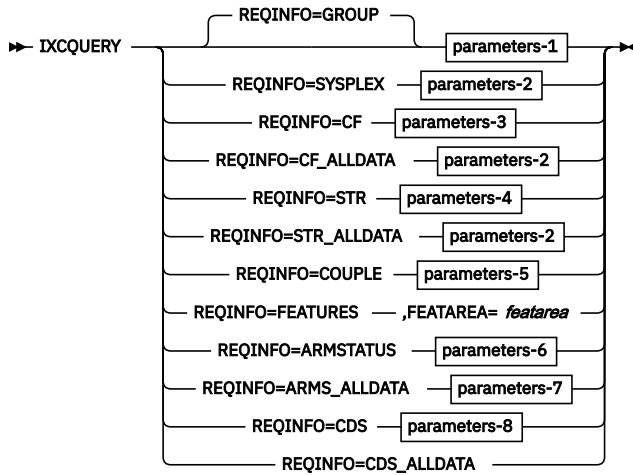
Specify the version of the parameter list that you want generated with the PLISTVER keyword. See Chapter 2, “[Specifying a Macro Version Number](#),” on page 5 for considerations when specifying the version of the parameter list with PLISTVER.

## Syntax Diagram

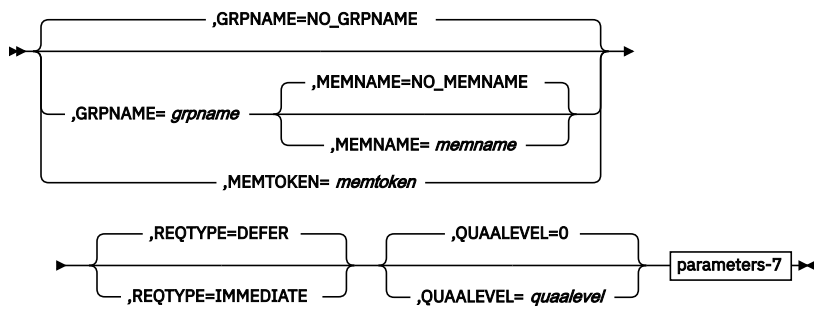
---

The syntax of the IXCQUERY macro is as follows:

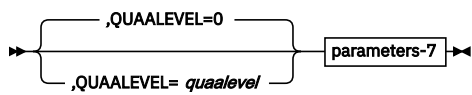
## main diagram



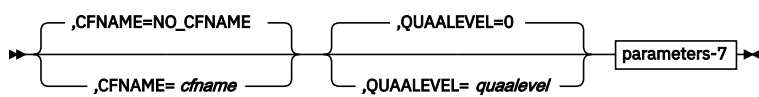
## parameters-1



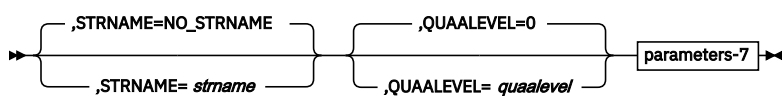
## parameters-2

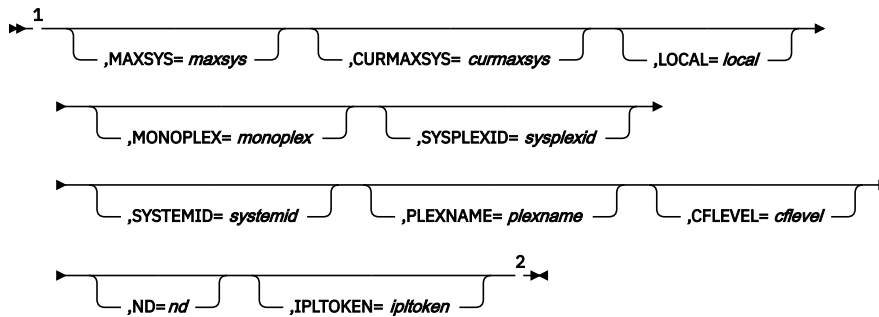


## parameters-3



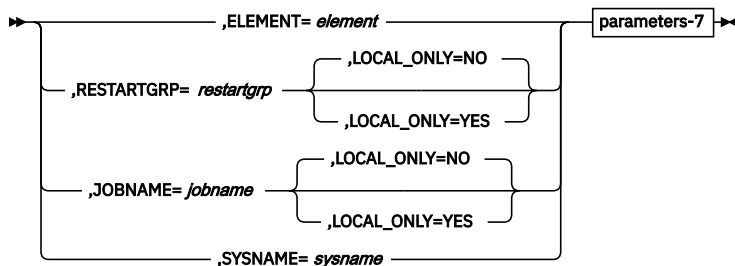
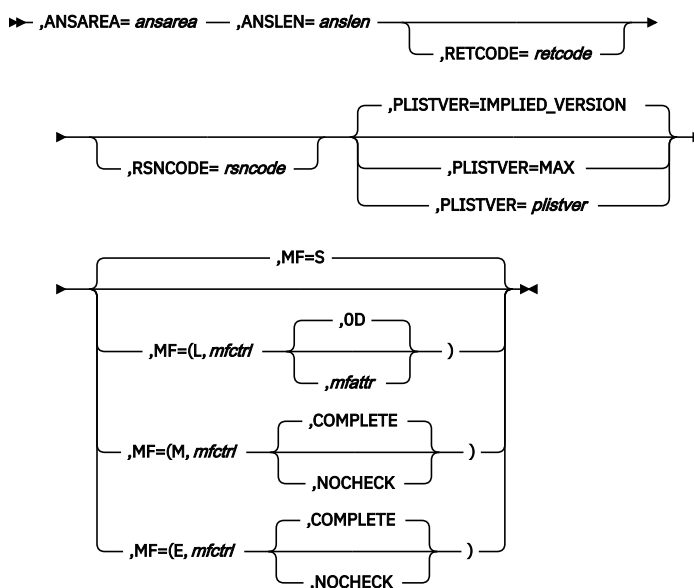
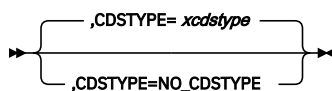
## parameters-4



**parameters-5**

Notes:

- <sup>1</sup> At least one of these keys must be specified.
- <sup>2</sup> At least one of these keys must be specified.

**parameters-6****parameters-7****parameters-8**

## Parameter descriptions

---

The parameter descriptions are listed in alphabetical order. Default values are underlined:

### **,ANSAREA=ansarea**

Use this output parameter to specify the storage area where IXCQUERY returns the requested information. The IXCYQUAA macro provides the format of the information area. The area can be in the primary address space or in an address/data space that is addressable through a public entry on the caller's DU-AL. Use the ANSLEN parameter to specify the length of the area.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the output area where IXCQUERY is to return the requested information.

### **,ANSLEN=anslen**

Use this input parameter to specify the length in bytes of the area that you provided on the ANSAREA parameter. You can estimate the length by consulting the data structures mapped by the IXCYQUAA macro. If you do not provide enough space, XCF lets you know how much space you should have provided.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword storage area which contains the length in bytes.

### **,CFLEVEL=cflevel**

Use this output parameter to specify a storage area to contain the maximum coupling facility operational level supported by the MVS operating system where the IXCQUERY macro was issued.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword output area to contain the coupling facility level.

### **,CFNAME=NO CFNAME**

### **,CFNAME=cfname**

Use this input parameter to specify the name of the coupling facility for which query data is to be returned. The name must be eight characters long, padded on the right with blanks if necessary; the valid characters are A-Z, 0-9, \$, @, #, and underscore (\_). The name must begin with an uppercase alphabetic character.

CFNAME is an optional parameter. If CFNAME is not specified, or if CFNAME=NO\_CFNAME is specified, then general information about all coupling facilities in the CFRM active policy is returned.

If a specific CFNAME is requested, then detailed information about that coupling facility is returned if the specified facility is in the active policy.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-character input area which contains the name of the coupling facility.

### **,CURRMAXSYS=currmaxsys**

Use this output parameter to specify a storage area to contain the maximum number of systems that can participate in the sysplex. When control returns from IXCQUERY processing, this area contains the current maximum number of systems that could participate in the sysplex given its current environment and restrictions. The value can range from 1 to the value returned for MAXSYS.

- If you specified either PLEXCFG=LOCAL or PLEXCFG=MONOPLEX in IEASYSxx, then the value of *currmaxsys* is 1.
- If you specified a different configuration with PLEXCFG, then this area contains the maximum number of systems that could appear within a sysplex.

**Note:** The value can change dynamically over time without a system or sysplex re-IPL.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword storage area to contain the value of the current maximum number of systems.

### **,ELEMENT=element**

Use this input parameter to specify the name of the element for which data is to be returned. The element name must be 16 characters long, padded on the right with blanks.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-byte field that contains the name of the element.

**,FEATAREA=featarea**

Use this output parameter to specify a storage area in which the system is to return information about the XCF and XES software features installed on this system.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 32-byte field to contain the XCF and XES software features as mapped by QUREQFEATURES in IXCYQUAA.

**,GRPNAME=NO\_GRPNAME**

**,GRPNAME=grpname**

Use this input parameter to specify the name of the group for which query data is to be returned. The name must be eight characters long, padded on the right with blanks if necessary; the valid characters are A-Z, 0-9, \$, @, and #. The record returned for each member of the specified group includes the state of the member, its member token, the name of the system on which the member was last active, and all user state fields.

If GRPNAME is not specified, or if GRPNAME=NO\_GRPNAME is specified, information about all groups is returned.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-character input field which contains the name of the group.

**,IPLTOKEN=ipltoken**

Use this output parameter to specify a storage area to contain the IPL token of the MVS system where the IXCQUERY macro was issued. In XCF-LOCAL mode, the returned IPLTOKEN value is 0.

**To code:** Specify the RS-type name or address (using a register from 2 to 12) of a 64-bit field to contain the IPL token.

**,JOBNAME=jobname**

Use this input parameter to specify the name of the batch job or started task for which data is to be returned.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-byte field that contains the jobname.

**,LOCAL=local**

Use this output parameter to specify a storage area that you use to test if the sysplex is in XCF-local mode. If the system is in XCF-local mode, then XCF returns the value X'01' in the storage area; otherwise XCF returns the value X'00' in the storage area.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-bit output area to contain the local mode indicator.

**,LOCAL\_ONLY=NO**

**,LOCAL\_ONLY=YES**

Use this input parameter to specify the scope of the data for which information is to be provided.

**NO**

This indicates that information is to be returned for elements running on any system in the sysplex that satisfy this request.

**YES**

This indicates that information is to be returned for all elements running on the current system.

**,MAXSYS=maxsys**

Use this output parameter to specify a storage area to contain the maximum number of systems that can participate in the sysplex, based on the MVS system level of the system on which the IXCQUERY request is issued and the sysplex configuration. When control returns from IXCQUERY:

- If you specified either PLEXCFG=LOCAL or PLEXCFG=MONOPLEX in IEASYSxx, then the value of *maxsys* is 1.
- If you specified a different configuration with PLEXCFG, then this area contains the maximum number of systems that could appear within a sysplex.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword storage area to contain the value of the maximum number of systems.

**,MEMNAME=NO\_MEMNAME**

**,MEMNAME=memname**

Use this input parameter to specify the name of the specific member of the group for which data is to be returned. The name must be 16 characters long, padded on the right with blanks if necessary; the valid characters are A-Z, 0-9, \$, @, and #.

If MEMNAME is not specified, or if MEMNAME=NO\_MEMNAME is specified, data is returned for all members of the group.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-character input field which contains the name of the member.

**,MEMTOKEN=memtoken**

Use this input parameter to specify the member token of the specific member for which data is to be returned.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-byte input variable that contains the member token.

**,MF=S**

**,MF=(L,mfctrl)**

**,MF=(L,mfctrl,mfattr)**

**,MF=(L,mfctrl,0D)**

**,MF=(M,mfctrl)**

**,MF=(M,mfctrl,COMPLETE)**

**,MF=(M,mfctrl,NOCHECK)**

**,MF=(E,mfctrl)**

**,MF=(E,mfctrl,COMPLETE)**

**,MF=(E,mfctrl,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

**,mfctrl**

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

**,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

**,COMPLETE**

**,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

**COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if `SMILE=var` were an optional parameter and the default is `SMILE=NO_SMILE` then it would not be documented. However, if the default was `SMILE=-`, then it would be documented because a value would be the default.

**NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

**,MONOPLEX=monoplex**

Use this output parameter to specify a storage area that you use to test if the sysplex is in monoplex mode. If the system is in monoplex mode, then XCF returns the value X'01' in the storage area; otherwise XCF returns the value X'00' in the storage area.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-bit output area to contain the monoplex mode indicator.

**,ND=nd**

Use this output parameter to specify a storage area to contain the node descriptor of the MVS system where the IXCQUERY macro was issued.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 32-byte field to contain the node descriptor.

**,IPLTOKEN=ipltoken**

Use this output parameter to specify a storage area to contain the IPL token of the MVS system where the IXCQUERY macro was issued. In XCF-LOCAL mode, the returned IPLTOKEN value is 0.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 64-bit field to contain the IPL token.

**,PLEXNAME=plexname**

Use this output parameter to specify a storage area to contain the name of the sysplex. When control returns from IXCQUERY processing, this area contains the name of the sysplex in which this system is participating.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-byte output area to contain the sysplex name.

**,PLISTVER=IMPLIED\_VERSION****,PLISTVER=MAX****,PLISTVER=plistver**

Use this input parameter to specify the version of the macro. See [“Understanding IXCQUERY Version Support”](#) on page 282 for a description of the options available with PLISTVER.

**,QUAALEVEL=0****,QUAALEVEL=quaalevel**

Use this input parameter to specify the level of the IXCQUAA record mappings that the system returns in the answer area. Valid values are 0, 1, 2, and 3.

- A value of 0 indicates that base level IXCQUAA records will be returned.
- A value of 1 indicates that level-1 IXCQUAA records will be returned. A level-1 IXCQUAA record is identified by the final “1” in its mapping name. For example, QUACF1, QUASTR1, and QUASTRCF1 are some of the level-1 mapping macros available. To request a level-1 IXCQUAA record, use version 2 of the IXCQUERY macro by coding `QUAALEVEL=1`.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the one-byte area containing the level of the IXCQUAA record mappings.



```
,REQINFO=GROUP
,REQINFO=SYSPLEX
,REQINFO=CF
,REQINFO=CF_ALLDATA
,REQINFO=STR
,REQINFO=STR_ALLDATA
,REQINFO=COUPLE
,REQINFO=FEATURES
,REQINFO=ARMSTATUS
,REQINFO=ARMS_ALLDATA
,REQINFO=SERVERS
,REQINFO=CDS
,REQINFO=CDS_ALLDATA
```

Use this input parameter to specify the type of information that is requested:

### GROUP

Specifies that IXCQUERY is to return data about all groups, a specific group, or a single member of a specific group.

- When requesting information about all groups (IXCQUERY REQINFO=GROUP), the data returned includes a header record, mapped by QUAHDR, followed by one record for each group, mapped by QUAGRP.
- When requesting information about all members of a specified group (IXCQUERY REQINFO=GROUP, GRPNAME=group\_name, the data returned includes a header record, mapped by QUAHDR, followed by one record for each member of the specified group, mapped by QUAMEM, QUAMEM1, or QUAMEM2.
- When requesting information about a specific member of a group (IXCQUERY REQINFO=GROUP, GRPNAME=group\_name, MEMNAME=member\_name or IXCQUERY REQINFO=GROUP, MEMTOKEN=member\_token), the data returned includes a header record, mapped by QUAHDR, followed by a record for the specified member, mapped by QUAMEM, QUAMEM1, or QUAMEM2.

The data includes a header and one record for each group requested mapped by QUAHDR, QUAGRP, and QUAMEM, QUAMEM1, or QUAMEM2.

### SYSPLEX

Specifies that IXCQUERY is to return data about all systems in the sysplex. The data includes a header record, mapped by QUAHDR, and one record for each system in the sysplex, mapped by QUASYS (QUASYS1 when QUALEVEL=1 or QUASYS2 when QUALEVEL=2). The record includes the system name, effective operator notification interval, effective failure detection interval, and system status. See *z/OS MVS Setting Up a Sysplex* for more information about effective operator notification interval and effective failure detection interval.

### CF

Specifies that IXCQUERY is to return general information about all coupling facilities or detailed information about a named coupling facility. General information includes one header record, mapped by QUAHDR, and one record for each coupling facility defined in the CFRM active policy, mapped by QUACF (QUACF1 when QUALEVEL=1). The information includes the name and ID of the coupling facility, the size of the dump space, the number of systems connected, the number of structures in the coupling facility that cannot be added to the policy, and the name and node descriptor of the coupling facility (as mapped by IXLYNDE).

Detailed information includes one header record, mapped by QUAHDR, and information mapped by QUACFSC, QUACF, QUACFSTR, (QUACFSC1, QUACF1, and QUACFSTR1 when QUALEVEL=1). The detailed information includes names of the systems connected to the coupling facility and names of structures and their allocation status.

### CF\_ALLDATA

Specifies that IXCQUERY is to return detailed information about all coupling facilities defined in the active CFRM policy. The answer area will contain the QUAHDR, the QUACF, and the QUACFSC and QUACFSTR for each coupling facility. When QUALEVEL=1, the answer area includes QUAHDR, QUACF1, QUACFSC1, and QUACFSTR1 for each coupling facility.

**STR**

Specifies that IXCQUERY is to return general information about all coupling facility structures in the CFRM active policy or detailed information about a named coupling facility structure. General information includes a header record, mapped by QUAHDR, and one record for each coupling facility structure in the CFRM active policy, mapped by QUASTR (QUASTR1 when QUAALevel=1). The information includes the name of the coupling facility structure, the size, indicators as to the state of the coupling facility structure, the number of associated preference list records, the number of associated exclusion list records, the number of allocated structures with this name, and the number of connectors to the structure.

Detailed information includes a header record, mapped by QUAHDR, and information mapped by QUASTR, QUASTRPL, QUASTRXL, QUASTRCF, QUASTRUSER, and QUASTRSYS, (QUASTR1, QUASTRPL1, QUASTRXL1, QUASTRCF1, QUASTRUSER1, and QUASTRSYS when QUAALevel=1). The information includes names of the coupling facilities in the structure's preference list, names of the structures in the structure's exclusion list, names of the coupling facilities where the structure is allocated, and detailed information about each connector to the specified structure.

**STR\_ALLDATA**

Specifies that IXCQUERY is to return detailed information about all coupling facility structures in the CFRM active policy. The answer area will contain the QUAHDR, the QUASTR, and the QUASTRPL, QUASTRXL, QUASTRCF, QUASTRUSER, and QUASTRSYS for each structure. When QUAALevel=1, the answer area includes QUAHDR, QUASTR1, QUASTRPL1, QUASTRXL1, QUASTRCF1, QUASTRUSER1, and QUASTRSYS for each structure.

**COUPLE**

Specifies that IXCQUERY is to return information about the sysplex. The scope of the information depends on the other parameters specified. IXCQUERY with REQINFO=COUPLE:

- Generates an inline expansion
- Does not require a control parameter list
- Does not produce a return code,
- Does not preserve registers 0, 1, 14, or 15 across the expansion.

**Note:** You cannot code REQINFO=COUPLE on the execute form of IXCQUERY.

**FEATURES**

Specifies that IXCQUERY is to return information about the XCF and XES software features installed on the system from which the request is made. The system returns a bit string mapped by QUREQFEATURES in IXCYQUAA.

IXCQUERY with REQINFO=FEATURES:

- Generates an inline expansion
- Does not require a control parameter list
- Does not produce a return code,
- Does not preserve registers 0, 1, 14, or 15 across the expansion.

**Note:** You cannot code REQINFO=FEATURES on the execute form of IXCQUERY.

**ARMSTATUS**

Specifies that IXCQUERY is to return information about elements that are registered with the automatic restart manager. The scope of the information depends on the other parameters specified.

The information returned is mapped by the IXCYQUAA macro in the QUAARMS section. This information includes the name of the element, the status of the element, and the number of times the element has been restarted.

**ARMS\_ALLDATA**

Specifies that IXCQUERY is to return information about all elements that are registered with the automatic restart manager.

The information returned is mapped by the IXCYQUAA macro in the QUAARMS section. This information includes the name of the elements, the status of the elements, and the number of times the elements have been restarted.

## **CDS**

**CDSTYPE=*xcdstype***

**CDSTYPE=NO CDSTYPE**

Requests either general couple data set information on all functions or detailed couple data set information for a specific function. For *xcdstype*, contains the external name of the couple data set function for which query data is to be returned.

If specified, the name must be 8 characters long, padded to the right with blanks, if necessary. If CDSTYPE is specified, the answer area contains the QUAHDR and the appropriate QUACDSFUN, QUACDS, QUACDSSU, and QUACDSNAR records if the specified couple data set function has been defined to the system. If CDSTYPE is not specified, the answer area contains the QUAHDR and the appropriate QUACDSFUN and QUACDS records for each couple data set function defined to the system.

Policy information is returned in the QUACDSFUN record only if the primary and home address spaces are the same.

## **CDS\_ALLDATA**

Requests detailed couple data set information on all functions.

The answer area will contain the QUAHDR and the appropriate QUACDSFUN, QUACDS, QUACDSSU, and QUACDSNAR records for each couple data set function defined to the system.

Policy information is returned in the QUACDSFUN record only if the primary and home address spaces are the same.

For more information about the REQINFO option, see [\*z/OS MVS Programming: Sysplex Services Guide\*](#).

**,REQTYPE=DEFER**

**,REQTYPE=IMMEDIATE**

Use this input parameter to indicate whether you want the most current group and member data.

## **DEFER**

This indicates that you are requesting the most current group and member data. The system suspends your work unit while the data requested is serialized and accessed.

## **IMMEDIATE**

This indicates that you are requesting the in-storage version of the group and member data. The system does not suspend your work unit while the data requested is accessed.

**,RESTARTGRP=*restartgrp***

Use this input parameter to specify the name of the restart group for which data is to be returned. The restart group name must be 16 characters long, padded on the right with blanks. Data on all elements in the group will be returned.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-byte field that contains the name of the restart group.

**,RETCODE=*retcode***

Use this output parameter to specify the location in which the system is to copy the return code from GPR 15.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword field to contain the return code.

**,RSNCODE=*rsncode***

Use this output parameter to specify the location in which the system is to copy the reason code from GPR 0.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword field to contain the reason code.

**,SERVERNAME=NO\_SERVERNAME****,SERVERNAME=servername**

Use this input parameter to specify the name of one or more servers for which information is to be returned. A given server name matches the server name pattern indicated by the SERVERNAME keyword if the nonzero sections of the pattern are identical to the corresponding sections of the server name. If you do not specify the SERVERNAME keyword, information about servers with any name is returned.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 32-byte area containing the server name pattern. The server name pattern indicated by the SERVERNAME keyword must be a valid server name, except that 8-byte sections 2, 3, and 4 (in any combination) can contain hexadecimal zero instead of a name. For a description of server naming conventions, see [Chapter 24, “IXCSRVR — Define a Server to XCF,”](#) on page 385.

**,STRNAME=NO\_STRNAME****,STRNAME=strname**

Use this input parameter to specify the name of the coupling facility structure for which query data is to be returned. The name must be 16 characters long, padded on the right with blanks if necessary; the valid characters are A-Z, 0-9, \$, @, #, and underscore (\_). The name must begin with an uppercase alphabetic character. IBM structure names begin with SYS or the letters A-I.

STRNAME is an optional parameter. If STRNAME is not specified, or if STRNAME=NO\_STRNAME is specified, then general information about all coupling facility structures in the CFRM active policy is returned.

If a specific STRNAME is requested, then detailed information about that coupling facility structure is returned if the specified structure is defined in the CFRM active policy.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-character input area to contain the structure name.

**,SYSNAME=sysname**

Use this input parameter to specify the name of the system for which data is to be returned. Information about elements currently running on the specified system **and** elements that originally registered on the specified system will be returned.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-byte field that contains the system name.

**,SYSNAME=sysname**

Use this input parameter to limit the returned information to servers residing on the named system. If the SYSNAME keyword is not specified, information about servers on all systems is returned.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-byte area containing the system name.

**,SYSPLEXID=sysplexid**

Use this output parameter to specify a storage area to contain the sysplex identifier. When control returns from IXCQUERY processing, this area contains the unique sysplex identifier, or token, that identifies the particular sysplex. The token is established when a sysplex is initialized and exists as long as the sysplex exists. (A sysplex is initialized when the first system IPLs into the sysplex.)

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 64-bit output area to contain the sysplex identifier.

**,SYSTEMID=systemid**

Use this output parameter to specify a storage area to contain the XCF system identifier of the system on which the IXCQUERY was invoked. When control returns from IXCQUERY processing, the high order byte contains the XCF system slot number (an index into a table of systems in the sysplex). The low order three bytes contain the XCF system sequence number used to identify a unique instantiation of the system in the sysplex.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 32-bit output area to contain the system identifier.

## Return and reason codes

When the IXCQUERY macro returns control to your program:

- GPR 15 (and *retcode*, if you coded RETCODE) contains a return code.
- When the value in GPR 15 is not zero, GPR 0 (and *rsncode*, if you coded RSNCODE) contains a reason code if applicable.

Mapping macro IXCYQUAA provides equate symbols for the reason codes. The following table identifies the hexadecimal return and reason codes and the equate symbol associated with each reason code.

Table 15. Return and Reason Codes for the IXCQUERY Macro		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0	None.	<b>Meaning:</b> IXCQUERY completed successfully and returned the requested information. <b>Action:</b> None.
4	00000004	<b>Equate Symbol:</b> QUAARSNRECORDSREMAIN <b>Meaning:</b> Program error. IXCQUERY completed successfully and provided some data; however, ANSAREA is too small to contain all the requested data. <b>Action:</b> Take one or more of the following actions: <ul style="list-style-type: none"> <li>• If the required information is contained in the ANSAREA, no further action needs to be taken.</li> <li>• If the required information is not contained in the ANSAREA, obtain a larger ANSAREA and reissue IXCQUERY. The QUAHTLEN field contains the ANSAREA size that is required to contain all of the information requested. However, because IXCQUERY returns only a snapshot of the current environment, it is possible that the QUAHTLEN may be too small on the next invocation.</li> <li>• Ensure that you specified the correct length for the answer area.</li> </ul>
8	0000002C	<b>Equate Symbol:</b> QUAARScdsnotfound <b>Meaning:</b> Program error. The couple data set type specified is not defined to the sysplex. <b>Action:</b> Take one or more of the following actions: <ul style="list-style-type: none"> <li>• Ensure that the correct external name for the couple data set function was specified.</li> <li>• Correct the group name and retry the request.</li> <li>• Any further action depends on your application. Some type of recovery action might need to be taken. However, if the couple data set is no longer defined to the sysplex, no action is required.</li> </ul>

Table 15. Return and Reason Codes for the IXCQUERY Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	00000004	<p><b>Equate Symbol:</b> QUAARSNGROUPNOTFOUND</p> <p><b>Meaning:</b> Program error. The group name specified is not defined to XCF.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• Ensure that the correct group name, group name address, and ALET (if appropriate) were specified.</li> <li>• Correct the group name and retry the request.</li> <li>• Any further action depends on your application. Some type of recovery action might need to be taken. However, if the group no longer exists, no action is required.</li> </ul>
8	00000008	<p><b>Equate Symbol:</b> QUAARSNREQINFONOTVALID</p> <p><b>Meaning:</b> Program error. The REQINFO information is not valid.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• Ensure that GROUP, SYSPLEX, COUPLE, CF, CF_ALLDATA, STR_ALLDATA, STR, ARMSTATUS, or ARMS_ALLDATA was specified for REQINFO.</li> <li>• If REQINFO=GROUP was specified, ensure that GRPNAME or MEMNAME was specified.</li> <li>• Ensure that the parameter list was not inadvertently overlaid.</li> <li>• Ensure that the correct parameter list and ALET were specified.</li> <li>• If your program is running in AR ASC mode, ensure that SYSSTATE ASCENV=AR was specified.</li> </ul>
8	0000000C	<p><b>Equate Symbol:</b> QUAARSNREQTYPEINCOR</p> <p><b>Meaning:</b> Program error. The caller specified the REQTYPE control parameter incorrectly.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• Ensure that a REQTYPE of DEFER or IMMEDIATE was specified.</li> <li>• Ensure that the parameter list was not inadvertently overlaid.</li> <li>• Ensure that the correct parameter list address and ALET were specified.</li> <li>• If your program is running in AR mode, ensure that SYSSTATE ASCENV=AR was specified.</li> </ul>

Table 15. Return and Reason Codes for the IXCQUERY Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	00000010	<p><b>Equate Symbol:</b> QUAARSNMEMBERNOTFOUND</p> <p><b>Meaning:</b> Program error. The member name specified is not defined within the specified group, or the MEMTOKEN specified was not defined to XCF.</p> <p><b>Action:</b> The action you take depends on your application. If you have reason to believe that the member is defined, ensure that:</p> <ul style="list-style-type: none"> <li>• The parameter list was not inadvertently overlaid.</li> <li>• The correct parameter list address and ALET were specified.</li> <li>• SYSSTATE ASCENV=AR was specified if your program is running in AR mode.</li> </ul>
8	00000014	<p><b>Equate Symbol:</b> QUAARSNANSAREATOOSMALL</p> <p><b>Meaning:</b> Program error. The length the caller specified on ANSLEN is too small to contain even the header.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• Provide an answer area that is large enough to contain at least the header information as mapped by the field QUAHDR of the query answer area mapping.</li> <li>• Ensure that you specified the correct length for the answer area.</li> <li>• Ensure that the parameter list was not inadvertently overlaid.</li> <li>• Ensure that you specified the correct answer area address and ALET.</li> <li>• If your program is running in AR mode, ensure that you specified SYSSTATE ASCENV=AR</li> </ul>
8	00000018	<p><b>Equate Symbol:</b> QUAARSNANSAREANOACCESS</p> <p><b>Meaning:</b> Program error. XCF cannot access the area specified by ANSAREA.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• Ensure that the correct ANSAREA address, ALET, and ANSLEN were specified.</li> <li>• Ensure that the parameter list storage area was not inadvertently freed by your program.</li> <li>• If your program is running in AR mode, ensure that you specified SYSSTATE ASCENV=AR before issuing the IXCQUERY macro.</li> </ul>

Table 15. Return and Reason Codes for the IXCQUERY Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	0000001C	<p><b>Equate Symbol:</b> QUAARSNANSALETNOTVALID</p> <p><b>Meaning:</b> Program error. The ALET that qualifies the address of the ANSAREA is neither zero nor is it associated with a valid public entry on the DU-AL.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• Your program is not intended to run in primary ASC mode.</li> <li>• If your program is running in AR mode, you specified SYSSTATE ASCENV=AR before issuing the IXCQUERY macro.</li> <li>• The ALET for the parameter list is a valid public entry on the DU-AL or is zero (primary address space ALET).</li> </ul>
8	00000020	<p><b>Equate Symbol:</b> QUAARSNCFNOTFOUND</p> <p><b>Meaning:</b> Program error. The coupling facility name specified is not defined in the CFRM active policy.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• Ensure that the correct coupling facility name was specified.</li> <li>• Correct the coupling facility name and retry the request.</li> <li>• Any further action depends on your application. Some type of recovery action might need to be taken. However, if the coupling facility is no longer defined in the CFRM active policy, no action is required.</li> </ul>
8	00000024	<p><b>Equate Symbol:</b> QUAARSNSTRNOTFOUND</p> <p><b>Meaning:</b> Program error. The structure name specified is not defined in the CFRM active policy.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• Ensure that the correct coupling facility structure name was specified.</li> <li>• Correct the coupling facility structure name and retry the request.</li> <li>• Any further action depends on your application. Some type of recovery action might need to be taken. However, if the coupling facility is no longer defined in the CFRM active policy, no action is required.</li> </ul>



Table 15. Return and Reason Codes for the IXCQUERY Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	00000028	<p><b>Equate Symbol:</b> QUAARSNARMNAMENOTFOUND</p> <p><b>Meaning:</b> Program error. The job, element, system, or restart group name specified is not known to the automatic restart manager.</p> <p><b>Action:</b> The action you take depends on your application. If you have reason to believe that the name is correct ensure that:</p> <ul style="list-style-type: none"> <li>• The parameter list was not inadvertently overlaid.</li> <li>• The correct parameter list address and ALET were specified.</li> <li>• SYSSTATE ASCENV=AR was specified if your program is running in AR mode.</li> <li>• The correct name was specified.</li> </ul>
8	00000034	<p><b>Equate Symbol:</b> QUAARSNAMODE24</p> <p><b>Meaning:</b> Program error. The IXCQUERY macro was issued in 24-bit addressing mode.</p> <p><b>Action:</b> IXCQUERY runs in 31-bit addressing mode. Correct your program so that it calls IXCQUERY while in 31-bit addressing mode.</p>
8	00000040	<p><b>Equate Symbol:</b> QUAARSNBADPLISTRSD</p> <p><b>Meaning:</b> Program error or environmental error. A reserved field in the control parameter list is not zero.</p> <p><b>Action:</b> Check to see if your program inadvertently overlaid the parameter list storage, and that it was assembled with the correct macro library for the release of MVS your program is running on.</p>
8	000000A0	<p><b>Equate Symbol:</b> IXCARMINVR0</p> <p><b>Meaning:</b> System error.</p> <p><b>Action:</b> Make sure your program was assembled with the correct macro library for the release of MVS your program is running on. If the macro version is correct, retry the request at least once.</p>

Table 15. Return and Reason Codes for the IXCQUERY Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	000000A4	<p><b>Equate Symbol:</b> IXCARMR0TYPECONFL</p> <p><b>Meaning:</b> System error.</p> <p><b>Action:</b> Verify that:</p> <ul style="list-style-type: none"> <li>• The parameter list was not inadvertently overlaid.</li> <li>• The parameter list was initialized.</li> <li>• Your program was assembled with the correct macro library for the release of MVS your program is running on.</li> <li>• If IXCQUERY was called in AR mode: <ul style="list-style-type: none"> <li>– The SYSSTATE ASCENV=AR macro was issued prior to this macro.</li> <li>– If the parameter list address was specified using explicit register notation, the corresponding access register was updated accordingly.</li> <li>– This area is either in the primary address space or in an address/data space that is addressable through a public entry on the caller's DU-AL.</li> </ul> </li> </ul> <p>If everything is verified, retry the request at least once. If the problem persists, record the return and reason code, and supply them to the appropriate IBM support personnel.</p>
8	00000100	<p><b>Equate Symbol:</b> QUAARSNPLISTALETNOTVALID</p> <p><b>Meaning:</b> Program error. The ALET that qualifies the address of the control parameter list is neither zero nor associated with a valid public entry on the caller's DU-AL.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• Your program is not intended to run in primary ASC mode.</li> <li>• You specified SYSSTATE ASCENV=AR before issuing the IXCQUERY macro.</li> <li>• The ALET for the parameter list is a valid public entry on the DU-AL or is zero (primary address space ALET).</li> </ul>
8	00000104	<p><b>Equate Symbol:</b> QUAARSNVERSIONNOTVALID</p> <p><b>Meaning:</b> Program or environmental error. The version number in the control parameter list is not valid. Your program might have inadvertently written over an area in the control parameter list.</p> <p><b>Action:</b> Check to see if your program inadvertently overlaid the parameter list storage, and that it was assembled with the correct macro library for the release of MVS your program is running on.</p>

Table 15. Return and Reason Codes for the IXCQUERY Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	00000108	<b>Equate Symbol:</b> QUAARSNFUNCCODENOTVALID <b>Meaning:</b> Program error or environmental error. The function code in the control parameter list is not valid. Your program might have inadvertently written over an area in the control parameter list. <b>Action:</b> Check to see if your program inadvertently overlaid the parameter list storage, and that it was assembled with the correct macro library for the release of MVS your program is running on.
8	0000010C	<b>Equate Symbol:</b> QUAARSNPLISTNOACCESS <b>Meaning:</b> Program error or environmental error. XCF could not access the control parameter list. <b>Action:</b> Check to see if your program inadvertently freed or overlaid the parameter list storage, and that it was assembled with the correct macro library for the release of MVS your program is running on.
8	00000118	<b>Equate Symbol:</b> QUAARSNNOTTASKMODE <b>Meaning:</b> Program error. The caller is not in task mode. <b>Action:</b> Correct your program so that it issues IXCQUERY while in task mode.
8	0000011C	<b>Equate Symbol:</b> QUAARSNNOTENABLED <b>Meaning:</b> Program error. The caller is not enabled. <b>Action:</b> Correct your program so that it issues IXCQUERY while it is enabled.
8	00000120	<b>Equate Symbol:</b> QUAARSNHASLOCK <b>Meaning:</b> Program error. The caller is holding a lock. <b>Action:</b> Correct your program so that it issues IXCQUERY when it is not holding a lock.
8	00000124	<b>Equate Symbol:</b> QUAARSNHASEUTFRR <b>Meaning:</b> Program error. The caller has an EUT FRR established. <b>Action:</b> Correct your program so that it issues IXCQUERY when it does not have an EUT FRR established.
8	00000128	<b>Equate Symbol:</b> QUAARSNQUAALEVELNOTVALID <b>Meaning:</b> Program error. The caller specified a value for QUAAREVEL that is not valid. <b>Action:</b> Correct your program so that it specifies a valid value for QUAAREVEL.

Table 15. Return and Reason Codes for the IXCQUERY Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	00000004	<p><b>Equate Symbol:</b> QUAARSNDSPSERVFAIL</p> <p><b>Meaning:</b> XCF was unable to create a data space for an IXCQUERY request. This reason code applies only to IXCQUERY requests REQINFO=CF, CF_ALLDATA, STR, or STR_ALLDATA.</p> <p><b>Action:</b> Retry the request one or more times. If the problem persists, record the return and reason codes and supply them to the appropriate IBM support personnel.</p>
C	00000008	<p><b>Equate Symbol:</b> QUAARSNALESERVFAIL</p> <p><b>Meaning:</b> XCF was unable to associate a data space to an address space on behalf of an IXCQUERY request. This reason code applies only to IXCQUERY requests REQINFO=CF, CF_ALLDATA, STR, or STR_ALLDATA.</p> <p><b>Action:</b> Retry the request one or more times. If the problem persists, record the return and reason codes and supply them to the appropriate IBM support personnel.</p>
C	00000018	<p><b>Equate Symbol:</b> QUAARSNTASKABENDED</p> <p><b>Meaning:</b> While the issuing task was suspended for XCF processing, the system abended the task (that is, another unit of work attempted to abnormally terminate this task). No data was returned in the ANSAREA. This reason code applies only to REQINFO=GROUP,REQTYPE=DEFER requests.</p> <p><b>Action:</b> Retry the request one or more times. If the problem persists, record the return and reason codes and supply them to the appropriate IBM support personnel.</p>
C	00000144	<p><b>Equate Symbol:</b> QUAARSNNOCFRMDSN</p> <p><b>Meaning:</b> Environmental error. The CFRM active policy could not be read because the couple data set supporting TYPE(CFRM) is not accessible to this system. This reason code applies only to IXCQUERY requests REQINFO=CF, CF_ALLDATA, STR, or STR_ALLDATA.</p> <p><b>Action:</b> Ensure that the request is issued from a system with access to a CFRM couple data set.</p>
C	00000154	<p><b>Equate Symbol:</b> QUAARSNNOCFRMPOL</p> <p><b>Meaning:</b> A CFRM policy has not been activated. This reason code applies only to IXCQUERY requests REQINFO=CF, CF_ALLDATA, STR, or STR_ALLDATA.</p> <p><b>Action:</b> Ensure that a CFRM policy is active.</p>

Table 15. Return and Reason Codes for the IXCQUERY Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	0000015C	<b>Equate Symbol:</b> QUAARSNFAILCFRMREAD <b>Meaning:</b> The CFRM active policy could not be read because the couple data set supporting TYPE(CFRM) is in error. This reason code applies only to IXCQUERY requests REQINFO=CF, CF_ALLDATA, STR, or STR_ALLDATA. <b>Action:</b> Retry the request one or more times. If the problem persists, record the return and reason codes and supply them to the appropriate IBM support personnel.
C	00000160	<b>Equate Symbol:</b> QUAARSNNOARMDSN <b>Meaning:</b> Environmental error. The automatic restart management couple data set could not be read because the couple data set supporting TYPE(ARM) is not accessible to this system. This reason code applies only to IXCQUERY requests REQINFO=ARMSTATUS and ARMS_ALLDATA. <b>Action:</b> If this reason code is not expected, contact the system programmer or issue a message to the operator to see if the automatic restart management couple data set can be brought online for this system. Issue this IXCQUERY request from a system that is connected to the automatic restart management couple data set.
C	00000164	<b>Equate Symbol:</b> QUAARSNFAILARMREAD <b>Meaning:</b> Environmental error. The automatic restart manager active policy could not be read because the couple data set supporting TYPE(ARM) is in error. This reason code applies only to IXCQUERY requests REQINFO=ARMSTATUS, and ARMS_ALLDATA. <b>Action:</b> Retry the request one or more times. If the problem persists, record the return and reason codes and supply them to the appropriate IBM support personnel.
10	xxxxxxx	<b>Meaning:</b> System error. XCF processing failed. <b>Action:</b> Retry the request one or more times. If the problem persists, record the return and reason codes and supply them to the appropriate IBM support personnel.

## Example

*Operation:* Request the most recent information about a specific member MEMB1 of the group GROUPA. Register 2 points to the area where XCF is to place the member information. XCF is to store the return code and reason code into the RETURN and REASON fields. The code is as follows:

LA	R2,MYAREA	OBTAIN ADDRESS OF OUTPUT AREA FOR IXCQUERY	X
IXCQUERY	REQINFO=GROUP, GRPNAME=GROUPA, ANSAREA=(R2),		X
	ANSLEN=AREALEN, REQTYPE=DEFER, RETCODE=RETURN,		X
	RSNCODE=REASON, MEMNAME=MEMB1, MF=S		
GROUPA	DC CL8 'GROUPA '	NAME OF GROUP FOR WHICH DATA	X

## IXCQUERY Macro

MEMB1	DC	CL16'MEMB1'	IS TO BE RETURNED NAME OF MEMBER FOR WHICH DATA	X
MYAREA	DS	CL156	IS TO BE RETURNED OUTPUT AREA TO CONTAIN DATA	X
RETURN	DS	1F	RETURNED BY IXCQUERY RETURN CODE	
REASON	DS	1F	REASON CODE	
AREALEN	DC	F'156'	LENGTH OF OUTPUT AREA	

## Chapter 19. IXCQUIES – Place an XCF Member in a Quiesced State

### Description

The IXCQUIES macro allows an active XCF member to place itself in a quiesced state. In a quiesced state, the member can no longer use the monitoring and signalling services of XCF. If active members have a group user-routine, XCF notifies those members that the member is quiesced. XCF delivers outstanding messages sent by the member, and discards undelivered messages that were sent to the member.

IXCQUIES requires that the member have permanent status recording established.

### Environment

The following are the environment requirements for the caller.

Environment	Environment requirement
Minimum authorization:	Supervisor state or PKM allowing key 0 - 7
Dispatchable unit mode:	Task
Cross memory mode:	PASN=HASN, any HASN, any SASN. The primary address space must be the same as the primary address space of the caller of the IXCJOIN macro that placed the calling member in the active state, or the caller must be executing in the master scheduler address space.
AMODE:	31-bit
ASC mode:	Primary or access register (AR)
Interrupt status:	Enabled for I/O and external interrupts
Locks:	No locks held
Control parameters:	Must be in the primary address space or be in an address/data space that is addressable through a public entry on the caller's dispatchable unit access list (DU-AL)

### Programming Requirements

If the program is in AR mode, issue the SYSSTATE ASCENV=AR macro before IXCQUIES. SYSSTATE ASCENV=AR tells the system to generate code appropriate for AR mode.

### Restrictions

The member must be active, with permanent status recording established.

The caller can have no enabled, unlocked task (EUT) FRRs established.

### Input Register Information

Before issuing the IXCQUIES macro, the caller does not have to place any information into any register unless using it in register notation for a particular parameter, or using it as a base register.

## Output Register Information

When control returns to the caller, the general purpose registers (GPRs) contain:

### Register Contents

**0**

If GPR 15 contains a zero or X'10', GPR 0 is used as a work register by the system; otherwise, GPR 0 contains a reason code.

**1**

Used as a work register by the system.

**2-13**

Unchanged.

**14**

Used as a work register by the system.

**15**

Return code.

When control returns to the caller, the access registers (ARs) contain:

### Register Contents

**0-1**

Used as work registers by the system

**2-13**

Unchanged

**14-15**

Used as work registers by the system

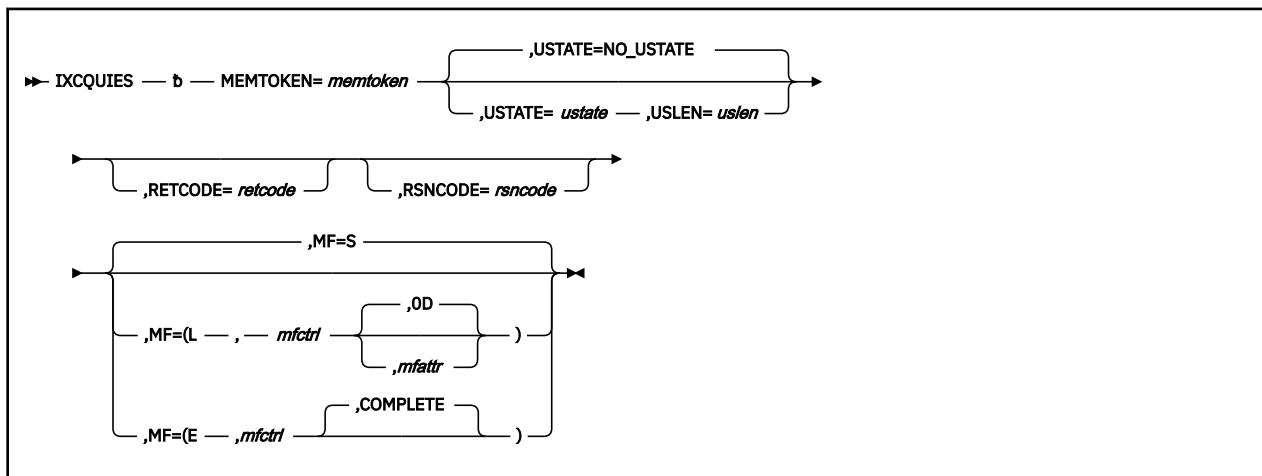
Some callers depend on register contents remaining the same before and after issuing a service. If the system changes the contents of registers on which the caller depends, the caller must save them before issuing the service, and restore them after the system returns control.

## Performance Implications

None.

## Syntax Diagram

The syntax of the IXCQUIES macro is as follows:





## Parameter Descriptions

The parameter descriptions are listed in alphabetical order. Default values are underlined:

### **MEMTOKEN=***memtoken*

Use this input parameter to specify the 64-bit token of the member. XCF provided this token when it placed the member in the active state.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-byte field that contains the member token.

### **,MF=**S

### **,MF=**(L,*mfctrl*)

### **,MF=**(L,*mfctrl*,*mfattr*)

### **,MF=**(L,*mfctrl*,0D)

### **,MF=**(M,*mfctrl*)

### **,MF=**(M,*mfctrl*,COMPLETE)

### **,MF=**(M,*mfctrl*,NOCHECK)

### **,MF=**(E,*mfctrl*)

### **,MF=**(E,*mfctrl*,COMPLETE)

### **,MF=**(E,*mfctrl*,NOCHECK)

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

### **,mfctrl**

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

### **,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

### **,COMPLETE**

### **,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

### **COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if SMILE=*var* were an optional parameter and the default is SMILE=NO\_SMILE then it would not be documented. However, if the default was SMILE=:-), then it would be documented because a value would be the default.

**NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

**,RETCODE=retcode**

Use this output parameter to specify the location in which the system is to copy the return code from GPR 15.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 4-byte field that will contain the return code when the quiesce requests completes.

**,RSNCODE=rsncode**

Use this output parameter to specify the location in which the system is to copy the reason code from GPR 0.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 4-byte field that will contain the reason code (if any) when the quiesce request completes.

**,USLEN=uslen**

Use this input parameter to specify the length in bytes of the user state data that you provide on the USTATE parameter. The length must be from 1 to 32 bytes. XCF overlays any previous value in the user state field up to the length you specify on USLEN. XCF does not pad the remainder of the user state field (up to 32 bytes) with zeros. IXCQUERY always returns the full 32 bytes, and group user-routines always receive the full 32 bytes. If you code USTATE, USLEN is required.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 4-byte field that contains the length of the user state data (USTATE).

**,USTATE=NO\_USTATE****,USTATE=ustate**

Use this input parameter to specify the area containing data that you want XCF to place in the user state field associated with the member. Use the USLEN parameter to specify the length of the user state data.

If you do not specify USTATE, or if you specify USTATE=NO\_USTATE, the user state field remains unchanged.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the area (with a length of USLEN) that contains the user state information.

## ABEND Codes

---

None.

## Return and Reason Codes

---

When the IXCQUIES macro returns control to your program:

- GPR 15 (and *retcode*, if you coded RETCODE) contains a return code.
- GPR 0 (and *rsncode*, if you coded RSNCODE) contains a reason code, if applicable.

Macro IXCYCON provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

**0**

IXCRETCODEOK

**4**

IXCRETCODEWARNING

**8**

IXCRETCODEPARMERROR

**C**

IXCRETCODEENVERROR

## 10

## IXCRETCODECOMPERROR

The following table identifies the hexadecimal return and reason codes and the equate symbol associated with each reason code.

<i>Table 16. Return and Reason Codes for the IXCQUIES Macro</i>		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
00	None.	<p><b>Meaning:</b> IXCQUIES completed successfully; XCF places the member in a quiesced state.</p> <p><b>Action:</b> None.</p>
04	04	<p><b>Equate Symbol:</b> IXCQUIESRSNEXITSNOTPURGED</p> <p><b>Meaning:</b> Environmental error. IXCQUIES completed successfully; XCF did not purge the group, status, or message user-routine SRBs successfully. For a group, status, or message exit, it is possible that the member specified on IXCJOIN is still running. Once an exit returns to XCF, it will not be scheduled again. XCF has attempted to purge the exits several times. This condition occurs only if there was an XCF error or the current task was asynchronously abended several times while XCF was in control to process the IXCQUIES request.</p> <p><b>Action:</b> Your application should be aware that one of the exits might still be running. If this return code occurs more than once for IXCQUIES, take the following actions:</p> <ul style="list-style-type: none"> <li>• Determine if any asynchronous abends are being issued against the current task. If so, you might need to reduce their frequency.</li> <li>• XCF should have taken an SDUMP to record the abend or abends. If the SDUMP indicates that the error was caused by SRB-to-task percolation or application code issuing a CALLRTM macro against your task, this is probably not an XCF error. If you believe this is an XCF error, record the return and reason code, and supply it to the appropriate IBM support personnel.</li> </ul>
08	04	<p><b>Equate Symbol:</b> IXCQUIESRSNNOTACTIVE</p> <p><b>Meaning:</b> Program error. The member token does not identify an active member.</p> <p><b>Action:</b> Ensure that the correct MEMTOKEN was specified. Further action depends on your application. If the member must be in a not-defined state, you can use the IXCQUERY service to determine what state the member is currently in. If the member is currently in a failed, quiesced, or created state, use the IXCDELET service to place the member in a not-defined state.</p>

Table 16. Return and Reason Codes for the IXCQUIES Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
08	08	<p><b>Equate Symbol:</b> IXCQUIESRSNINAPPROPRIATEPRIMARY</p> <p><b>Meaning:</b> Program error. The primary address space is neither the master scheduler address space nor the primary address space of the caller of the IXCJOIN that placed the member in the active state.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The correct MEMTOKEN was specified.</li> <li>• Your program issues IXCQUIES only from the master scheduler address space or the address space from which the member (MEMTOKEN) joined the group.</li> </ul>
08	0C	<p><b>Equate Symbol:</b> IXCQUIESRSNNOTLASTING</p> <p><b>Meaning:</b> Program error. The caller specified a member token for a member that did not specify LASTING=YES on the IXCJOIN macro. To be placed in a quiesced state, a member has to request permanent status recording.</p> <p><b>Action:</b> Ensure that the correct MEMTOKEN was specified. If your application requires the member to have permanent status recording, ensure that LASTING=YES is specified on the IXCJOIN that joins the member to the group.</p>
08	10	<p><b>Equate Symbol:</b> IXCQUIESRSNINAPPROPRIATESYSTEM</p> <p><b>Meaning:</b> Program error. The system is not the system on which the IXCJOIN for the member was issued.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The correct MEMTOKEN was specified.</li> <li>• Your program issues IXCQUIES only for members that joined on the same system as your program.</li> </ul>
08	40	<p><b>Equate Symbol:</b> IXCQUIESRSNPLISTRSDNOTVALID</p> <p><b>Meaning:</b> Program error or environmental error. A reserved field in the control parameter list is not zero.</p> <p><b>Action:</b> Check to see if your program inadvertently overlaid the parameter list storage, and that it was assembled with the correct macro library for the release of MVS your program is running on.</p>

Table 16. Return and Reason Codes for the IXCQUIES Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
08	100	<p><b>Equate Symbol:</b> IXCQUIESRSNPLISTBADALET</p> <p><b>Meaning:</b> Program error. Your program is running in AR mode, and the ALET that qualifies the address of the control parameter list is neither zero nor associated with a valid public entry on the caller's DU-AL.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• You specified SYSSTATE ASCENV=AR before issuing the IXCQUIES macro.</li> <li>• The ALET for the parameter list is on the DU-AL or is zero (primary address space ALET).</li> <li>• Your program is not running in primary ASC mode.</li> </ul>
08	104	<p><b>Equate Symbol:</b> IXCQUIESRSNPLISTVERSIONNOTVALID</p> <p><b>Meaning:</b> Program error or environmental error. The version number in the control parameter list is not valid.</p> <p><b>Action:</b> Check to see if your program inadvertently overlaid the parameter list storage, and that it was assembled with the correct macro library for the release of MVS your program is running on.</p>
08	108	<p><b>Equate Symbol:</b> IXCQUIESRSNPLISTBADFUNCTION</p> <p><b>Meaning:</b> Program error. The function code in the control parameter list is not valid.</p> <p><b>Action:</b> Check to see if your program inadvertently overlaid the parameter list storage.</p>
08	10C	<p><b>Equate Symbol:</b> IXCQUIESRSNPLISTBADSTG</p> <p><b>Meaning:</b> Program error. XCF could not access the control parameter list.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• Ensure that the correct parameter list storage area was specified.</li> <li>• If your program is running in AR mode, ensure that: <ul style="list-style-type: none"> <li>– You specified SYSSTATE ASCENV=AR before issuing the IXCQUIES macro.</li> <li>– The parameter list ALET is correct.</li> </ul> </li> <li>• Ensure that the parameter list storage area was not inadvertently freed by your program.</li> </ul>

Table 16. Return and Reason Codes for the IXCQUIES Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
08	110	<p><b>Equate Symbol:</b> IXCQUIESRSNUSTATEBADSTG</p> <p><b>Meaning:</b> Program error. XCF could not access the USTATE value.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• Ensure that the correct USTATE address was specified.</li> <li>• If your program is running in AR mode, ensure that: <ul style="list-style-type: none"> <li>– You specified SYSSTATE ASCENV=AR before issuing the IXCQUIES macro.</li> <li>– The USTATE ALET is correct.</li> </ul> </li> <li>• Ensure that the USTATE storage area was not inadvertently freed by your program.</li> </ul>
08	114	<p><b>Equate Symbol:</b> IXCQUIESRSNUSLENBADVALUE</p> <p><b>Meaning:</b> Program error. The USLEN value is less than 1 or greater than 32.</p> <p><b>Action:</b> Correct the USLEN, and retry the request.</p>
08	118	<p><b>Equate Symbol:</b> IXCQUIESRSNNOTTASKMODE</p> <p><b>Meaning:</b> Program error. The caller is not in task mode.</p> <p><b>Action:</b> Correct your program so that it issues IXCQUIES only while in task mode, and retry the request.</p>
08	11C	<p><b>Equate Symbol:</b> IXCQUIESRSNNOTENABLED</p> <p><b>Meaning:</b> Program error. The caller is not enabled.</p> <p><b>Action:</b> Correct your program so that it issues IXCQUIES only while enabled.</p>
08	120	<p><b>Equate Symbol:</b> IXCQUIESRSNPRIMARYNOTHOME</p> <p><b>Meaning:</b> Program error. The primary address space is not equal to the home address space.</p> <p><b>Action:</b> Correct your program so that it does not use IXCQUIES while in cross memory mode. You might want to pass this restriction on to your caller when you are unsure of the environment your caller might have been in.</p>
0C	18	<p><b>Equate Symbol:</b> IXCQUIESRSNTASKABENDED</p> <p><b>Meaning:</b> Environmental error. While the issuing task was suspended for XCF processing, the task was abended; that is, another unit of work attempted to abnormally terminate this task). The state of the IXCQUIES request is unpredictable.</p> <p><b>Action:</b> Find out why this task was being abended.</p>

Table 16. Return and Reason Codes for the IXCQUIES Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
10	None.	<b>Meaning:</b> System error. XCF processing failed.  <b>Action:</b> Retry the request one or more times. If the problem persists, record the return code, and supply it to the appropriate IBM support personnel.

## Example

*Operation:* Place a member in the quiesced state, and put the value X'33' in the user state field. XCF is to store the return code and reason code into the RETURN and REASON fields. The code is as follows:

```

                IXCQUIES MEMTOKEN=TOKEN2, USTATE=STATE3, USLEN=LEN,          X
                  RETCODE=RETURN, RSNCODE=REASON, MF=S
TOKEN2    DS    CL8                                TOKEN OF MEMBER TO BE PLACED  X
                                                IN QUIESCED STATE
RETURN    DS    1F                                RETURN CODE
REASON    DS    1F                                REASON CODE
STATE3    DC    X'33'                             USER STATE VALUE
LEN        DC    F'1'                             LENGTH OF USER STATE DATA

```

You can obtain the member token from the QUAMTKN field in the area returned by IXCJOIN or IXCQUERY, and mapped by the IXCYQUAA mapping macro.





## Chapter 20. IXCRECV— Receive Client/Server Information

### Description

The XCF receive service (IXCRECV macro) is one of the macros that comprise the XCF client/server interfaces. With these interfaces, a "client" can send a request to a "server" for processing and then receive the "results" provided by the server. Related macros include the XCF server interface (IXCSRVR macro) that is used to create servers, and the XCF send service (IXCSEND macro) that is used to send requests to servers and responses to clients. See Chapter 24, "IXCSRVR — Define a Server to XCF," on page 385 and Chapter 22, "IXCSEND — Send Client/Server Requests and Responses," on page 337.

The IXCRECV macro completes a parameter list with caller provided data and calls the XCF service routine. Depending on the request type specified, XCF blocks (suspends) processing until all of the expected responses arrive or returns information about the status of the request.

For guidance information and other details on using IXCRECV with the client/server interfaces, see the IXCRECV topic in the chapter "Using XCF for Client/Server Communication" of *z/OS MVS Programming: Sysplex Services Guide*.

### Environment

The following are the environment requirements for the caller.

Environment	Environment requirement
Minimum authorization:	Supervisor state or PKM allowing keys 0-7.
Dispatchable unit mode:	Task or SRB mode.
Cross memory mode:	Any PASN, any HASN, any SASN
AMODE:	31- or 64-bit. If in 64-bit mode, specify SYSSTATE AMODE64=YES before invoking this macro.
ASC mode:	Primary or access register (AR) If in Access Register ASC mode, specify SYSSTATE ASCENV=AR before invoking this macro.
Interrupt status:	Enabled for I/O and external interrupts
Locks:	No locks held
Control parameters:	The storage containing the IXCRECV parameter list must reside in the primary address space of the caller, or in a space addressable through a public entry on the dispatchable unit access list (DU-AL), or in a common area data space.

### Programming Requirements

If the program is in Access Register ASC mode, specify SYSSTATE ASCENV=AR before invoking this macro. If in 64-bit mode, specify SYSSTATE AMODE64=YES before invoking this macro. Include the mapping macro IXCYSRVR in your program to map the data returned in ANSAREA.

## Restrictions

---

IXCRECV with REQTYPE=BLOCKING cannot be used by tasks higher in the task tree than the cross memory resource owning task (the top, or first, job step task in the address space).

Callers running in SRB mode should refrain from invoking the IXCRECV service with REQTYPE=BLOCKING under the following circumstances:

- After the SRB receives a x'47B' abend
- When running in a suspend exit after invoking SUSPEND

In these cases, the IXCRECV service routine might not be able to successfully wait for the requested responses to arrive.

## Input Register Information

---

Before issuing the IXCRECVmacro, the caller does not have to place any information into any register unless using it in register notation for a particular parameter, or using it as a base register.

## Output Register Information

---

When control returns to the caller, the GPRs contain:

**0**

Reason code based on GR15 or 0

**1**

Unpredictable

**2-13**

Unchanged

**14**

Unpredictable

**15**

Return code

When control returns to the caller, the access registers (ARs) contain:

**0-1**

Unpredictable

**2-13**

Unchanged

**14-15**

Unpredictable

## Performance Implications

---

To prevent backlogs that can degrade system performance, a receiver must be able to process results of a request in a timely fashion.

## Understanding IXCRECV Version Support

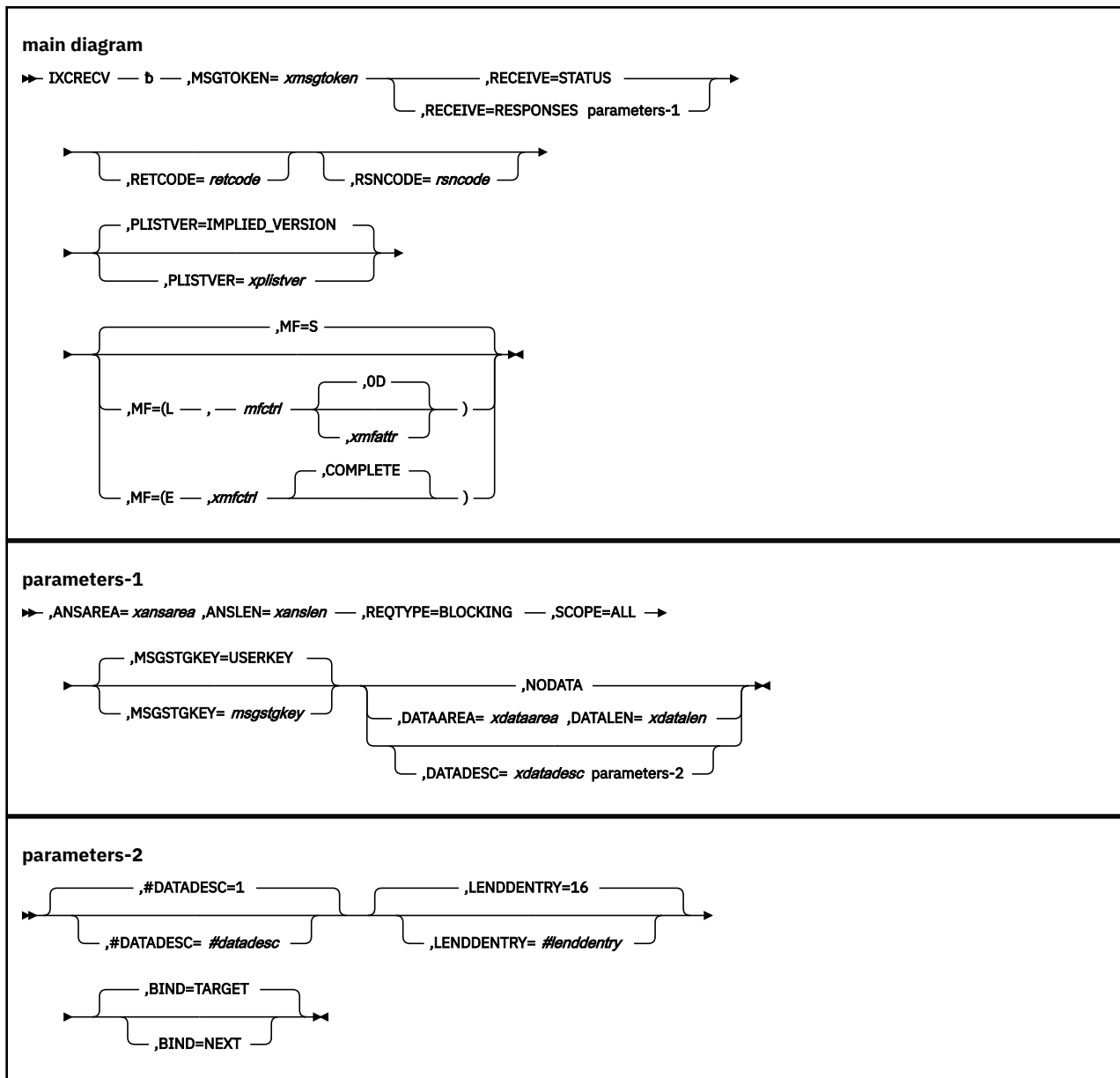
---

The IXCRECV macro supports version 0. Specify the version of the parameter list that you want generated with the PLISTVER keyword. See Chapter 2, “Specifying a Macro Version Number,” on page 5 for considerations when specifying the version of the parameter list with PLISTVER.

## Syntax

---

The IXCRECV macro is written as follows:



## Parameters

The parameters are explained as follows:

**,#DATADESC=#datadesc**

**,#DATADESC=1**

Use this optional input parameter to indicate the number of data descriptors in the data descriptor table. This parameter is valid when RECEIVE=RESPONSES and DATADESC=datadesc are specified. The default is 1.

**To code:** Specify the RS-type name or address in register (2)-(12), of a fullword field, or specify a literal decimal value that contains the number of data descriptors in the table.

**,ANSAREA=ansarea**

When RECEIVE=RESPONSES is specified, use this required input variable to specify an answer area where XCF is to store the metadata that describes the message and its responses. ANSAREA contains a header (ixcysrvr\_tAnsArea), a send descriptor (ixcysrvr\_tSendDescriptor), a target descriptor (ixcysrvr\_tTargetDescriptor) for each target, and a response descriptor (ixcysrvr\_tResponseDescriptor) for each response if EXPECTREPLY=YES was specified in the originating IXCSEND invocation. The mappings are defined in the IXCYSRVR macro.

In general, ANSAREA must be large enough to hold the header and all the descriptors. If not, none of the descriptors will be stored. The service routine returns with a return and reason code indicating that the ANSAREA needs to be bigger (ixcrecvRsnMoreAnsArea). The header indicates how much storage is needed (aa\_AnsAreaSize). The caller needs to obtain a sufficiently large answer area and reissue the request before the IXCSEND hold time (HOLDTIME) timeout value (See [Chapter 22, "IXCSEND — Send Client/Server Requests and Responses,"](#) on page 337.)

The answer area must reside in the primary address space of the caller, or in a space addressable through a public entry on the dispatchable unit access list (DU-AL), or in a common area data space.

**To code:** Specify the RS-type name or address in register (2)-(12), of a character field that contains the answer area.

**,ANSLEN=*anslen***

When RECEIVE=RESPONSES is specified, use this required input variable to specify the length in bytes of the answer area provided by the invoker.

**To code:** Specify the RS-type name or address in register (2)-(12), of a fullword field, or specify a literal decimal value to specify the length of the answer area.

**,BIND=TARGET**

**,BIND=NEXT**

Use this optional parameter to indicate how XCF is to associate entries in the data descriptor table to responses. This paramter is valid when DATADESC=*datadesc* and RECEIVE=RESPONSES are specified. The default is BIND=TARGET.

**,BIND=TARGET**

Entries in the data descriptor table are in one to one correspondence with the targets of the message. The number of entries in the data descriptor table must be greater than or equal to the number of message targets. Entry "i" in the table describes where the response from target "i" is to be stored. For example, if a request was sent to four targets, and only targets 1 and 3 replied with response data. With BIND=TARGET, the response data from target 1 will be stored in the storage area described by data descriptor table entry 1, and the response data from target 3 will be stored in the storage area described by entry 3. The storage areas described by entries 2 and 4 will not be used.

**,BIND=NEXT**

Entries in the data descriptor table are to be used successively for the next response to be delivered. For example, if a request was sent to four targets, and only targets 1 and 3 replied with response data. With BIND=NEXT, the response data from target 1 will be stored in the storage area described by data descriptor table entry 1, and the response data from target 3 will be stored in the storage area described by entry 2. The storage areas described by entries 3 and 4 will not be used.

**,DATALEN=*datalen***

When DATAAREA=*dataarea* and RECEIVE=RESPONSES are specified, use this required input parameter to contain the length in bytes of the data area provided by the invoker.

**To code:** Specify the RS-type name or address in register (2)-(12), of a fullword field, or specify a literal decimal value to specify the length of the data area.

**,LENDENTRY=*lendentry***

**,LENDENTRY=16**

Use this optional input parameter to indicate the length in bytes of each entry in the data descriptor table. This paramter is valid when DATADESC=*datadesc* and RECEIVE=RESPONSES are specified. As XCF iterates through the table, it locates the next descriptor by adding LENDENTRY to the location of the current descriptor. LENDENTRY must be greater than or equal to the length of one data descriptor, which is 16. If not specified, LENDENTRY defaults to the length of one data descriptor. The default is 16.

**To code:** Specify the RS-type name or address in register (2)-(12), of a fullword field, or specify a literal decimal value to specify the length pf each entry in the descriptor table.

**,MF=S**  
**,MF=(L,list addr)**  
**,MF=(L,list addr,attr)**  
**,MF=(L,list addr,OD)**  
**,MF=(E,list addr)**  
**,MF=(E,list addr,COMPLETE)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service. MF=S is the default.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter might be coded with the list form of the macro.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form of the macro stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

**,list addr**

The name of a storage area to contain the parameters. For MF=S and MF=E, this can be an RS-type address or an address in register (1)-(12).

**,attr**

An optional 1- to 60-character input string that you use to force boundary alignment of the parameter list. Use a value of 0F to force the parameter list to a word boundary, or 0D to force the parameter list to a doubleword boundary. If you do not code *attr*, the system provides a value of 0D.

**,COMPLETE**

Specifies that the system is to check for required parameters and supply defaults for omitted optional parameters.

**,MSGSTGKEY=msgstgkey**

**,MSGSTGKEY=USERKEY**

Use this optional input parameter to contain the storage key to be used when storing the response data into the indicated data areas (either DATAAREA or the areas described by DATADESC). This parameter is valid when RECEIVE=RESPONSES is specified. The high order nibble contains the storage key, the low order nibble is ignored. For example, set 'kkkk' in the binary bit string B'kkkkxxxx' to correspond to the desired storage key.

If MSGSTGKEY is not specified, the response data is stored using the PSW key in effect at the time the XCF receive service was called. The default is USERKEY.

**To code:** Specify the RS-type name or address in register (2)-(12), of an 8 bit field that contains the storage key.

**MSGTOKEN=msgtoken**

Use this required input parameter to contain the message token that identifies the message to be processed. The IXCSEND macro returned this token through the RETMSGTOKEN keyword when the message was sent.

If the indicated message no longer exists, the service routine sets a return and reason code to so indicate (ixcrecvRsnMsgNotFound) and returns to the caller.

**To code:** Specify the RS-type name or address in register (2)-(12), of a 32-character field that contains the message token.

**,NODATA**

**,DATAAREA=dataarea**

**,DATADESC=datadesc**

Use this required input parameter to specify how to handle the response data. This parameter is valid when RECEIVE=RESPONSES is specified.

**,NODATA**

One of set of mutually exclusive keywords indicating that no response data is to be stored.

**To code:** Specify the RS-type name or address in register (2)-(12), of a field.

**,DATAAREA=*dataarea***

One of set of mutually exclusive keywords indicating where XCF is to store the response data provided by the responder. The content, interpretation, and mapping of the response data is determined by the responder.

If multiple responses are being received, XCF concatenates the response data for each response successively in the indicated DATAAREA. The response descriptor in the ANSAREA can be used to locate the corresponding response data within DATAAREA.

Depending on the size of the response data and the number of responders, the amount of storage needed for DATAAREA could be significant. In such cases DATADESC might be more appropriate, as it might be easier to get one storage area per response rather than one storage area large enough to hold all the responses.

If DATAAREA is not large enough, none of the response data will be stored and the service routine returns with a return and reason code indicating that the DATAAREA needs to be bigger (ixcrecvRsnMoreDataArea). The header in the ANSAREA indicates how much storage is needed (aa\_DataAreaSize). The caller needs to obtain a sufficiently large data area and reissue the request before the HOLDTIME timeout value for the message expires. (See [Chapter 22, "IXCSEND — Send Client/Server Requests and Responses,"](#) on page 337.)

The data area must reside in the primary address space of the caller, or in a space addressable through a public entry on the dispatchable unit access list (DU-AL), or in a common area data space. The storage key of the data area must match the storage protect key indicated by the MSGSTGKEY keyword.

**To code:** Specify the RS-type name or address in register (2)-(12), of a character field that contains the data area.

**,DATADESC=*datadesc***

One of set of mutually exclusive keywords containing a table of one or more data descriptors. A data descriptor identifies a storage location where the response data for one response is to be stored. Data descriptors are mapped by ixcysrvr\_tDataDescriptor that is defined in the IXCYSVR macro. A data descriptor specifies the length, ALET, and address of a contiguous virtual storage area where the response data for one response is to be stored. The content, interpretation, and mapping of the response data stored in the storage area indicated by a data descriptor is determined by the responder.

The data descriptor table is an array of entries. Each entry has the same fixed size, and can contain data other than the data descriptor. The storage location named by DATADESC contains the first such data descriptor. Subsequent descriptors are iteratively located by adding the value LENDENTRY to the location of the current descriptor.

The storage area defined by the data descriptor must reside in the primary address space of the caller, or in a space addressable through a public entry on the dispatchable unit access list (DU-AL), or in a common area data space. The storage key of the data area must match the storage protect key indicated by the MSGSTGKEY keyword.

There must be a data descriptor for each response to be received. If not, the service routine returns with a return and reason code indicating that more descriptors are needed (ixcrecvRsnMoreDataDesc). Each such data descriptor must describe storage that is large enough to hold all the response data for the relevant response. If not, the service routine returns with a return and reason code indicating that more storage is required (ixcrecvRsnMoreDataArea). In the ANSAREA, the dd\_DataSize field within the data descriptor (md\_DataDesc) for the response message descriptor (rd\_MsgDesc) that is returned for each requested response indicates how much storage is needed for the associated response data. The caller needs to obtain a sufficient number of descriptors that describe large data areas and reissue the request before the IXCSEND hold time (HOLDTIME) timeout value for the message expires.

In all cases, if the data area for any one response is too small to hold the relevant response data, or if the data descriptor table does not contain enough entries for all responses, none of the responses will be stored.

If response data is stored, the response descriptor in the ANSAREA can be used to locate the corresponding message data.

The storage area containing the data descriptor table must reside in the primary address space of the caller, or in a space addressable through a public entry on the dispatchable unit access list (DU-AL), or in a common area data space.

**To code:** Specify the RS-type name or address in register (2)-(12), of a character field that contains the data descriptors.

**,PLISTVER=IMPLIED\_VERSION**

**,PLISTVER=MAX**

**,PLISTVER=0**

Use this input parameter to specify the version of the macro. See [“Understanding IXCRECV Version Support”](#) on page 314 for a description of the options available with PLISTVER.

**,RECEIVE=STATUS**

**,RECEIVE=RESPONSES**

Use this required parameter to indicate the kind of data the service routine is to gather.

**,RECEIVE=STATUS**

The service routine is to report the status of the request. The return code and reason code indicates whether the request is completed (RETCODE=0), or whether the request is still pending (RETCODE=4). If pending, the RSNCODE indicates whether there are responses available for delivery (ixcrecvRsnAvailable or ixcrecvRsnPending).

RECEIVE=STATUS is specified if the caller wants to obtain the status of the message but does not actually want to receive the responses or information about the responses. RECEIVE=STATUS can be used, for example, to poll for message completion.

**,RECEIVE=RESPONSES**

RECEIVE=RESPONSES is specified if the caller wants to receive the results associated with the indicated message.

An earlier IXCSEND SENDTO=SERVER request was used to send a message to one or more targets. If EXPECTREPLY=YES is specified, each target is expected to invoke IXCSEND SENDTO=ORIGINATOR to send its response in reply to the message. Each reply can contain "response data" (corresponding to the IXCSEND keywords MSGDATA or MSGDESC), or "response info" (keywords RESPRETCODE, RESPRSNCODE, SUPPLIEDLEVEL, SUPPORTSLEVEL), or both. The caller now wants to receive the results.

The keyword DATAAREA or the keyword DATADESC is used to indicate where the "response data" is to be stored. The metadata describing the status of the results is stored in the storage area identified by the ANSAREA keyword.

**,REQTYPE=BLOCKING**

Use this required parameter to indicate that the caller wants the service routine to wait for the results to arrive before returning. This parameter is valid when RECEIVE=RESPONSES is specified.

**,REQTYPE=BLOCKING**

If the requested results are not yet available, the IXCRECV service routine blocks (suspends) the caller. Control does not return to the caller until the requested results become available. To release a blocked receiver before all results become available, some other work unit can invoke the XCF message control service (IXCMMSGC) to discard the message (REQUEST=DISCARDMSG), complete the message (REQUEST=COMPLETION), or release the blocked receiver (REQUEST=RELEASEMSG).

**,RETCODE=retcode**

An optional output parameter into which the return code is to be copied from GPR 15. If you specify 15, GPR15, REG15, or R15 (within or without parentheses), the value will be left in GPR 15.

**To code:** Specify the RS-type name of a fullword field, or register (2)-(12) or (15), (GPR15), (REG15), or (R15).

**,RSNCODE=rsncode**

An optional output parameter into which the reason code is to be copied from GPR 0. If you specify 0, 00, GPR0, GPR00, REG0, REG00, or R0 (within or without parentheses), the value will be left in GPR 0.

**To code:** Specify the RS-type name of a fullword field, or register (0) or (2)-(12), (00), (GPR0), (GPR00), REG0), (REG00), or (R0).

**,SCOPE=ALL**

Use this required parameter to indicate that all results are to be gathered. This parameter is valid when RECEIVE=RESPONSES is specified.

**,SCOPE=ALL**

All results are to be gathered. This allows the caller to process all the results at one time.

The metadata of the results are to be stored in the answer area. If a data area (DATAAREA or DATADESC) is provided, the response data for each available response is to be stored in the data area.

## ABEND Codes

Abend codes an issuer of IXCRECV might receive are listed as follows. For detailed abend code information, see [z/OS MVS System Codes](#).

- Abend X'073' - Environment not valid. The caller held a lock.
- Abend X'C78' - XCF could not obtain storage to process the request. Try again later.

## Return and Reason Codes

When the IXCRECV macro returns control to your program:

- GPR 15 (and *retcode*, when you code RETCODE) contains a return code.
- When the value in GPR 15 is not zero, GPR 0 (and *rsncode*, when you code RSNCODE) contains a reason code.

The following table identifies the hexadecimal return and reason codes.

Table 17. Return and Reason Codes for the IXCRECV Macro		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0	None	Successful completion.  As applicable, all requested data has been stored in the ANSAREA and specified data areas.  For RECEIVE=STATUS, the message is complete.
4	4	<b>Equate Symbol:</b> IXCRECVRSNMOREANSAREA  <b>Meaning:</b> Successful completion, although with exceptional circumstances.  <b>Action:</b> ANSAREA too small. The ANSAREA is large enough to hold the header (ixcysrvr_tAnsArea) but does not have room for all the descriptors. See the value in the aa_AnsAreaSize returned in the ANSAREA; it indicates how much storage is required in order for the ANSAREA to hold all the descriptors.



Table 17. Return and Reason Codes for the IXCRECV Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	5	<p><b>Equate Symbol:</b> IXCRECVRSNMOREDATAAREA</p> <p><b>Meaning:</b> Successful completion, although with exceptional circumstances.</p> <p><b>Action:</b> Need more storage for response data. If DATAAREA was specified, DATAAREA is not large enough to receive all the requested response data. The aa_DataAreaSize returned in the ANSAREA indicates how much storage is required. If DATADESC was specified, at least one storage area described by a data descriptor was too small for the response data. In the ANSAREA, the dd_DataSize field in the response message descriptor (rd_MsgDesc) within one or more of the relevant response descriptors (ixcysrvr_tResponseDescriptor) indicates how much storage is needed for the response data.</p>
4	6	<p><b>Equate Symbol:</b> IXCRECVRSNMOREDATADESC</p> <p><b>Meaning:</b> Successful completion, although with exceptional circumstances.</p> <p><b>Action:</b> DATADESC needs more entries. There must be an entry for each response. The aa_#Desc returned in the ANSAREA indicates how many entries are needed.</p>
4	8	<p><b>Equate Symbol:</b> IXCRECVRSNPENDING</p> <p><b>Meaning:</b> The message is still pending.</p> <p><b>Action:</b> Not all the expected results have been received. No undelivered responses are available for processing.</p>
4	0C	<p><b>Equate Symbol:</b> IXCRECVRSNAVAILABLE</p> <p><b>Meaning:</b> The message is still pending.</p> <p><b>Action:</b> Not all the expected results have been received. There are undelivered responses available for processing.</p>
8	00010004	<p><b>Equate Symbol:</b> IXCRECVRSNBADSTGPLIST</p> <p><b>Meaning:</b> Parameter list is not accessible.</p> <p><b>Action:</b> Storage is not addressable. Correct the storage problem and retry.</p>
8	00010008	<p><b>Equate Symbol:</b> IXCRECVRSNBADALETPLIST</p> <p><b>Meaning:</b> Parameter list is not accessible.</p> <p><b>Action:</b> The ALET for the parameter list storage is not valid. Correct the error, and retry.</p>
8	0001000C	<p><b>Equate Symbol:</b> IXCRECVRSNBADVALANSLEN</p> <p><b>Meaning:</b> ANSAREA is too small.</p> <p><b>Action:</b> Specify an ANSAREA that is at least large enough to hold the header (ixcysrvr_tAnsArea).</p>

Table 17. Return and Reason Codes for the IXCRECV Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	00010018	<b>Equate Symbol:</b> IXCRECVRSNBADPLISTVERSION <b>Meaning:</b> Version number in parameter list is not supported. <b>Action:</b> Specify a version number that is supported, and retry.
8	00010048	<b>Equate Symbol:</b> IXCRECVRSNMSGNOTFOUND <b>Meaning:</b> The message identified by the MSGTOKEN no longer exists. <b>Action:</b> The message could have completed or been discarded.
8	000100EE	<b>Equate Symbol:</b> IXCRECVRSNBADENVNOTENABLED <b>Meaning:</b> Environmental error. Caller is not running enabled. <b>Action:</b> Ensure that the caller runs enabled.
8	00020004	<b>Equate Symbol:</b> IXCRECVRSNBADSTGANSAREA <b>Meaning:</b> Answer area is not accessible. <b>Action:</b> Storage pointed to by ANSAREA is not addressable. Correct the problem, and retry.
8	00020008	<b>Equate Symbol:</b> IXCRECVRSNBADALETANSAREA <b>Meaning:</b> Answer area is not accessible. <b>Action:</b> The ALET for the storage pointed to by ANSAREA is not valid. Correct the problem, and retry.
8	0002000C	<b>Equate Symbol:</b> IXCRECVRSNBADVALMSGTOKEN <b>Meaning:</b> MSGTOKEN value does not represent a valid message token. <b>Action:</b> Specify a valid message token, and retry.
8	00020018	<b>Equate Symbol:</b> IXCRECVRSNBADPLISTRSD <b>Meaning:</b> Invalid parameter list. A reserved field in the parameter list is not zero. <b>Action:</b> Reserved fields in the parameter list must be zero.
8	000200EE	<b>Equate Symbol:</b> IXCRECVRSNBADENVLOCKED <b>Meaning:</b> Caller is holding a lock. <b>Action:</b> Retry later.
8	00030008	<b>Equate Symbol:</b> IXCRECVRSNBADALETDATAAREA <b>Meaning:</b> Data area is not accessible. <b>Action:</b> The ALET for the storage pointed to by DATAAREA or by one of the DATADESC table entries is not valid. Correct the error, and retry.

Table 17. Return and Reason Codes for the IXCRECV Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	0003000C	<b>Equate Symbol:</b> IXCRECVRSNBADVALLENDENTRY <b>Meaning:</b> LENDDENTRY value is not valid. <b>Action:</b> LENDDENTRY must be greater than or equal to the length of one data descriptor, which is 16.
8	00030018	<b>Equate Symbol:</b> IXCRECVRSNBADPLISTRECEIVE <b>Meaning:</b> Invalid RECEIVE parameter. <b>Action:</b> RECEIVE must specify RESPONSES or STATUS.
8	00040004	<b>Equate Symbol:</b> IXCRECVRSNBADSTGDATAAREA <b>Meaning:</b> Data area storage is not accessible. The storage pointed to by DATAAREA or by one of the DATADESC table entries is not addressable. <b>Action:</b> The storage pointed to by DATAAREA or by one of the DATADESC table entries is not addressable. Correct the error, and retry.
8	00040008	<b>Equate Symbol:</b> IXCRECVRSNBADALETDATADESC <b>Meaning:</b> DATADESC table is not accessible. <b>Action:</b> The ALET for the DATADESC table storage is not valid. Correct the error, and retry.
8	00040018	<b>Equate Symbol:</b> IxcrecvRsnBadPlistDataArea <b>Meaning:</b> Invalid data area parameter. <b>Action:</b> One of the keywords DATAAREA, DATADESC, or NODATA must be specified.
8	00050004	<b>Equate Symbol:</b> IXCRECVRSNBADSTGDATADESC <b>Meaning:</b> DATADESC table is not accessible.. <b>Action:</b> Storage pointed to by DATADESC is not addressable. Correct the problem, and retry.
8	00050018	<b>Equate Symbol:</b> IXCRECVRSNBADPLISTSCOPE <b>Meaning:</b> Invalid SCOPE parameter. <b>Action:</b> SCOPE must be ALL.
8	00060004	<b>Equate Symbol:</b> IXCRECVRSNPAGEPROTECTDATAAREA <b>Meaning:</b> Data area is not accessible. <b>Action:</b> Storage pointed to by DATAAREA or by one of the DATADESC table entries is page protected.
8	00060018	<b>Equate Symbol:</b> IXCRECVRSNBADPLISTREQTYPE <b>Meaning:</b> Invalid REQTYPE parameter. <b>Action:</b> REQTYPE must specify BLOCKING.

Table 17. Return and Reason Codes for the IXCRECV Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	00070004	<b>Equate Symbol:</b> IXCRECVRSNKEYMISMATCHDATAAREA <b>Meaning:</b> Data area is not accessible. <b>Action:</b> The storage area pointed to by DATAAREA or by one of the DATADESC table entries cannot be stored into using the storage key indicated by MSGSTGKEY. Correct the error, and retry.
C	C04	<b>Equate Symbol:</b> IXCRECVRSNACTIVERECEIVER <b>Meaning:</b> Environmental error. Some other work unit is currently in the midst of receive processing for this message. <b>Action:</b> Try again later.
C	C05	<b>Equate Symbol:</b> IXCRECVRSNBLOCKINGCONFLICT <b>Meaning:</b> Environmental error. Some other work unit is currently in the midst of a blocking receive that conflicts with the receive request. For example, if a SCOPE=ALL receiver is already blocked waiting for the message to complete, no other blocking receive can be accepted. <b>Action:</b> Try again later.
C	C08	<b>Equate Symbol:</b> IXCRECVRSNNEEDRESOURCES <b>Meaning:</b> Environmental error. The request could not be processed because the system was unable to obtain the resources needed to process the request. <b>Action:</b> Try again later.
C	C10	<b>Equate Symbol:</b> IXCRECVRSNRELEASED <b>Meaning:</b> Environmental error. Blocked receiver was released, no responses received. This reason code applies only when REQTYPE=BLOCKING is specified. The XCF message control service (IXCMMSGC REQUEST=RELEASEMSG) was used to release the service routine while it was blocked and waiting for responses to arrive. No information is stored in the answer area or the data area. <b>Action:</b> None.
C	C11	<b>Equate Symbol:</b> IXCRECVRSNMSGDISCARDED <b>Meaning:</b> Environmental error. Message discarded, no responses received. The message identified by the MSGTOKEN was discarded by the IXCMMSGC DISCARDMSG service. The XCF message control service (IXCMMSGC REQUEST=DISCARDMSG) was used to discard the message identified by the MSGTOKEN while the service routine was blocked and waiting for responses to arrive. No information is stored in the answer area or the data area. <b>Action:</b> None.

Table 17. Return and Reason Codes for the IXCRECV Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	C12	<b>Equate Symbol:</b> IXCRECVRSNBADBLOCKINGENV <b>Meaning:</b> Environmental error. IXCRECV REQTYPE=BLOCKING cannot be issued from a SUSPEND exit routine or from an SRB routine that the system abended with a 47B system completion code. <b>Action:</b> None.
C	C13	<b>Equate Symbol:</b> IXCRECVRSNSYSTEMNOTREADY <b>Meaning:</b> Environmental error. The system is not ready to process the IXCRECV request. <b>Action:</b> Retry the request after allowing time for the system to complete the necessary initialization.
C	C14	<b>Equate Symbol:</b> IXCRECVRSNRECVBINDTERM <b>Meaning:</b> Environmental error. The receive bind entity identified on the RECVBIND keyword of the IXCSEND request for which this IXCRECV call was made for has ended, and the message results have been discarded by XCF. <b>Action:</b> None.
10	None	<b>Equate Symbol:</b> <b>Meaning:</b> Failure in XCF processing <b>Action:</b> None.

## Example

*Operation:* Receive the results for a message sent through the IXCSEND macro. This unit of work is to be suspended until all of the expected responses have arrived. All results for the message are to be gathered at once. MTOKEN contains the token of the message to be received. Register 2 points to the area where XCF is to store the status metadata of the results. Register 3 points to the area where XCF is to store the response data. XCF is to store the return code and reason code into variables RETURN and REASON. The code is as follows:

```

LA      R2,MYAA
LA      R3,MYDA
IXCRECV MSGTOKEN=MTOKEN,RECEIVE=RESPONSES,REQTYPE=BLOCKING, X
SCOPE=ALL,ANSAREA=(R2),ANSLEN=AALen,DATAAREA=(R3), X
DATALEN=DALEN,RETCODE=RETURN,RSNCODE=REASON,MF=S

MYAA    DS    CL1024    AREA TO PLACE THE METADATA
MYDA    DS    CL4096    AREA TO PLACE THE RESPONSE DATA
RETURN  DS    1F        RETURN CODE
REASON  DS    1F        REASON CODE
AALen   DC    F'1024'   LENGTH OF THE ANSWER AREA WHERE STATUS X
                        METADATA IS PLACED
DALEN   DC    F'4096'   LENGTH OF THE DATA AREA WHERE RESPONSE X
                        DATA IS PLACED

```



## Chapter 21. IXCREQ — Format a Request for the XCF Server

### Description

Use the IXCREQ macro to construct an XCF Server request. This macro expands inline to initialize and format an area that can then be used with the XCF Client/Server IXSEND macro to send a request to the XCF Server for processing. The client can then receive the returned response information through the XCF client/server IXCRECV macro.

For guidance information and other details on using IXCREQ with the client/server interfaces, see the IXCREQ topic in the chapter "Using XCF for Client/Server Communication" of *z/OS MVS Programming: Sysplex Services Guide*.

### Environment

The following are the environment requirements for the caller.

Environment	Environment requirement
Minimum authorization:	None, although Supervisor state or PKM allowing keys 0-7 is required to invoke the IXSEND macro service that is used to send the IXCREQ request to the XCF Server.
Dispatchable unit mode:	Task or SRB mode.
Cross memory mode:	Any PASN, any HASN, any SASN
AMODE:	31- or 64-bit. If in 64-bit mode, specify SYSSTATE AMODE64=YES before invoking this macro.
ASC mode:	Primary or access register (AR) If in Access Register ASC mode, specify SYSSTATE ASCENV=AR before invoking this macro. Any primary address space; any secondary address space; any home address space.
Interrupt status:	Enabled or disabled for I/O and external interrupts
Locks:	No locks held or locks held
Control parameters:	The storage area defined using the list form of the IXCREQ macro and the storage containing the control parameters used on the modify form of the IXCREQ macro must reside in the primary address space of the caller, or in a space addressable through a public entry on the dispatchable unit access list (DU-AL), or in a common area data space.

### Programming Requirements

None.

## Restrictions

No restrictions.

## Input Register Information

Before issuing the IXCREQ macro, the caller does not have to place any information into any register unless using it in register notation for a particular parameter, or using it as a base register.

## Output Register Information

When control returns to the caller, the GPRs contain:

**0**

Unchanged

**1**

Used as a work register

**2-15**

Unchanged

When control returns to the caller, the access registers (ARs) contain:

**0**

Unchanged

**1**

Used as a work register

**2-15**

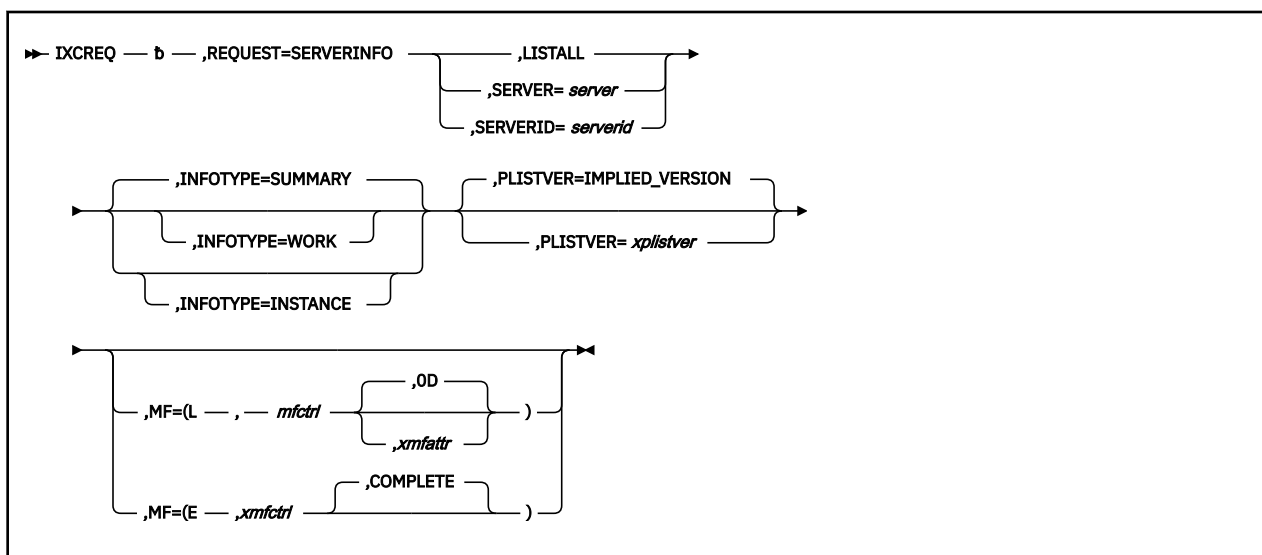
Unchanged

## Performance Implications

None.

## Syntax

The IXCREQ macro is written as follows:





## Parameters

---

The parameters are explained as follows:

### **,INFOTYPE=SUMMARY**

### **,INFOTYPE=WORK**

### **,INFOTYPE=INSTANCE**

When REQUEST=SERVERINFO is specified, use this optional parameter to indicate the type of server information to be obtained and returned in the DATAAREA provided on the IXCRECV macro service. The contents and mapping of the data area depends on the the particular request. See mapping macro IXCYSRVR for details. The default is INFOTYPE=SUMMARY.

### **,INFOTYPE=SUMMARY**

Obtain summary information that describes a server.

### **,INFOTYPE=WORK**

Obtain information that describes work that is pending for processing by the server and currently being worked on by the server.

### **,INFOTYPE=INSTANCE**

Obtain information about the instantiated server instance(s).

One or more values might be specified for the INFOTYPE parameter. If more than one value is specified, group the values within parentheses.

### **,LISTALL**

### **,SERVER=server**

### **,SERVERID=serverid**

When REQUEST=SERVERINFO is specified, use this required input parameter to obtain information about servers or server ids. You must specify one of the following mutually exclusive keywords:

### **,LISTALL**

A parameter keyword to request information about all servers defined to XCF on a system.

### **,SERVER=server**

A parameter variable containing the name of the server name or server name pattern for which information is to be returned.

Server names and server name patterns are mapped by ixcysrvr\_tName (macro IXCYSRVR).

Server names and server name patterns consist of four 8 byte sections. Each 8 byte section must be left justified, padded on the right with EBCDIC blanks as needed. Each section can contain any alphanumeric (A-Z,a-z,0-9), national (@,#,\$), or underscore (\_) character. Any section but the first can be entirely blank. Any section can contain a pattern with wild card characters. A question mark (?) within the pattern will match any one character and an asterisk (\*) within the pattern will match any sequence of zero or more characters. Server names are case sensitive.

**To code:** Specify the RS-type name or address in register (2)-(12), of a 32-character field.

### **,SERVERID=serverid**

A parameter variable that contains the server ID of the specific server instance for which information is to be returned for.

Server IDs for server instances of a server are available in the svrIir\_ServerID field of a ixcysrvr\_tSrvrInfoIR record for an IXCREQ SERVERINFO request that specified INFOTYPE=INSTANCE.

**To code:** Specify the RS-type name or address in register (2)-(12), of a 16-character field.

```
,MF=(L,list addr)
,MF=(L,list addr,attr)
,MF=(L,list addr,0D)
,MF=(E,list addr)
,MF=(E,list addr,COMPLETE)
,MF=(E,list addr,NOCHECK)
,MF=(M,list addr)
,MF=(M,list addr,COMPLETE)
,MF=(M,list addr,NOCHECK)
```

Use this required input parameter to specify the macro form.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter might be coded with the list form of the macro.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form of the macro stores the parameters into the storage area defined by the list form.

Use MF=M with the list form of the macro to provide different options according to user-provided input. Use the list form to define a storage area, and use the modify form to set the appropriate options.

IBM suggests that you use the modify and execute forms of IXCREQ in the following order:

- Use IXCREQ ...MF=(M,list-addr,COMPLETE) specifying appropriate parameters, including all required ones.
- Use IXCREQ ...MF=(M,list-addr,NOCHECK), specifying the parameters that you want to change.
- Use IXCREQ ...MF=(E,list-addr,NOCHECK), to execute the macro.

#### **,list addr**

The name of a storage area to contain the parameters. For MF=E and MF=M, this can be an RS-type name or an address in register (1)-(12).

#### **,attr**

An optional 1- to 60-character input string that you use to force boundary alignment of the parameter list. Use a value of 0F to force the parameter list to a word boundary, or 0D to force the parameter list to a doubleword boundary. If you do not code *attr*, the system provides a value of 0D.

#### **,COMPLETE**

Specifies that the system is to check for required parameters and supply defaults for omitted optional parameters.

#### **,NOCHECK**

Specifies that the system is not to check for required parameters and is not to supply defaults for omitted optional parameters.

#### **,PLISTVER=IMPLIED\_VERSION**

#### **,PLISTVER=MAX**

#### **,PLISTVER=0**

Use this optional input parameter to specify the version of the macro. PLISTVER determines which parameter list the system generates. PLISTVER is an optional input parameter on all forms of the macro, including the list form. When using PLISTVER, specify it on all macro forms used for a request and with the same value on all of the macro forms. The values are:

- **IMPLIED\_VERSION**, which is the lowest version that allows all parameters specified on the request to be processed. If you omit the PLISTVER parameter, IMPLIED\_VERSION is the default.
- **MAX**, if you want the parameter list to be the largest size currently possible. This size might grow from release to release and affect the amount of storage that your program needs.

If you can tolerate the size change, IBM recommends that you always specify `PLISTVER=MAX` on the list form of the macro. Specifying `MAX` ensures that the list-form parameter list is always long enough to hold all the parameters you might specify on the execute form, when both are assembled with the same level of the system. In this way, `MAX` ensures that the parameter list does not overwrite nearby storage.

- **0**, if you use the currently available parameters.

**To code:** Specify one of the following:

- `IMPLIED_VERSION`
- `MAX`
- A decimal value of 0

#### **,REQUEST=SERVERINFO**

Use this required parameter to indicate which request is to be formatted for one or more of the XCF servers to process.

#### **REQUEST=SERVERINFO**

Obtain server definition, server message delivery, work item and server instance information.

#### **,RETCODE=retcode**

Use this optional output parameter into which the return code is to be copied from GPR 15. If you specify 15, GPR15, REG15, or R15 (within or without parentheses), the value will be left in GPR 15.

**To code:** Specify the RS-type name of a fullword field, or register (2)-(12) or (15), (GPR15), (REG15), or (R15).

#### **,RSNCODE=rsncode**

Use this optional output parameter into which the reason code is to be copied from GPR 0. If you specify 0, 00, GPR0, GPR00, REG0, REG00, or R0 (within or without parentheses), the value will be left in GPR 0.

**To code:** Specify the RS-type name of a fullword field, or register (0) or (2)-(12), (00), (GPR0), (GPR00), REG0, (REG00), or (R0).

## ABEND Codes

None.

## Return and Reason Codes

The IXCREQ macro does not provide a return code. However, when the XCF send service (IXCSEND) is used to send the XCF Server request message created by the IXCREQ macro to an XCF Server for processing, the server that processes the request provides a return and reason code as described below. These codes are available in the answer area returned by the XCF receive service (IXCRECV) when the IXCRECV macro is used to receive the results provided by the server(s) that process the request. Fields `rd_RespRetcode` and `rd_RespRsncode` in the `ixcysrvr_tResponseDescriptor` record (see macro `IXCYSRVR`) contain the return and reason code, respectively, provided by the server.

The following table identifies the hexadecimal return and reason codes.

Table 18. Return and Reason Codes for the IXCREQ Macro		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0	None	Successful completion. The reason code is always zero (0).

Table 18. Return and Reason Codes for the IXCREQ Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	0001000C	<p><b>Equate Symbol:</b> IXCREQRSNNOREQUESTDATA</p> <p><b>Meaning:</b> No request data was received by the XCF Server exit for the IXCREQ request.</p> <p><b>Action:</b> Check the IXCSEND invocation to make sure that an XCF Server request message was specified on the MSGDATA keyword or in a MSGDESC data descriptor</p>
8	00010018	<p><b>Equate Symbol:</b> IXCREQRSNBADPLISTVERSION</p> <p><b>Meaning:</b>Version number for the XCF Server request message is not valid.</p> <p>When this reason code is set, diagnostic information mapped by ixcysrvr_tSrvrInfoDD is returned by the XCF server.</p> <p><b>Action:</b> Check the version number in the mapping:</p> <ul style="list-style-type: none"> <li>• srvrIdd_Diag1 contains the version number that the XCF server expected to receive for the IXCREQ request.</li> <li>• srvrIdd_Diag2 contains the version number that the XCF server received for the IXCREQ request</li> </ul>
8	0002000C	<p><b>Equate Symbol:</b> IXCREQRSNBADDATASIZE</p> <p><b>Meaning:</b> Unexpected amount of request data sent to the server. The value of MSGLEN on the IXCSEND request or the value in the dd_DataSize field of a MSGDESC data descriptor was not recognized by the XCF Server. The amount of data sent for an SERVERINFO request must equal the length of the storage area formatted using the List and Modify form of the IXCREQ macro.</p> <p><b>Action:</b> The amount of data sent for an SERVERINFO request must equal the length of the IXCREQ parameter list created using the List and Modify form of the IXCREQ macro.</p> <p>When this reason code is set, diagnostic information mapped by ixcysrvr_tSrvrInfoDD is returned by the XCF server. Checking the following values in the mapping:</p> <ul style="list-style-type: none"> <li>• srvrIdd_Diag1 contains the number of bytes of data that the XCF server expected to receive for the IXCREQ request.</li> <li>• srvrIdd_Diag2 contains the number of bytes of data that the XCF server received for the IXCREQ request.</li> </ul>
8	00020018	<p><b>Equate Symbol:</b> IXCREQRSNBADPLISTREQUEST</p> <p><b>Meaning:</b> REQUEST specified for the XCF Server is not valid. The IXCREQ REQUEST was not SERVERINFO.</p> <p><b>Action:</b> The IXCREQ REQUEST was not SERVERINFO. You must specify SERVERINFO on the modify form of the IXCREQ macro when formatting an XCF Server request message.</p>

Table 18. Return and Reason Codes for the IXCREQ Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	0003000C	<p><b>Equate Symbol:</b> IXCREQRSNBADVALQUERYINFO</p> <p><b>Meaning:</b> Request for SERVERINFO is not valid.</p> <p><b>Action:</b> One of the keywords LISTALL, SERVER, or SERVERID must be specified on the IXCREQ macro.</p>
8	0004000C	<p><b>Equate Symbol:</b> IXCREQRSNBADVALSERVER</p> <p><b>Meaning:</b> Name value specified on the IXCREQ SERVER keyword is not valid.</p> <p><b>Action:</b> Specify a valid server name or server name pattern. See the description of the SERVER keyword for server name syntax rules.</p>
8	00040018	<p><b>Equate Symbol:</b> IXCREQRSNBADPLISTLEN</p> <p><b>Meaning:</b> Request message length specified in the XCF Server request message is not valid.</p> <p><b>Action:</b> When this reason code is set, diagnostic information mapped by <code>ixcysrvr_tSrvrInfoDD</code> is returned by the XCF server. Check the following values in the mapping:</p> <ul style="list-style-type: none"> <li>• <code>srvrIdd_Diag1</code> contains the minimum message length that the XCF Server expected to receive for the IXCREQ request.</li> <li>• <code>srvrIdd_Diag2</code> contains the maximum message length that the XCF Server expected to receive for the IXCREQ request.</li> <li>• <code>srvrIdd_Diag3</code> contains the message length that the XCF Server received for the IXCREQ request.</li> </ul> <p>Use the modify form of the IXCREQ macro to format an XCF Server request message. Make sure that the storage for the XCF Server request message has not been corrupted or overlaid.</p>
8	0005000C	<p><b>Equate Symbol:</b> IXCREQRSNBADVALMSGCNTL</p> <p><b>Meaning:</b> MsgCntl content sent with the IXCREQ request is not valid. MsgCntl data is reserved for XCF use only.</p> <p><b>Action:</b> When this reason code is set, diagnostic information mapped by <code>ixcysrvr_tSrvrInfoDD</code> is returned by the XCF server. Check the following values in the mapping:</p> <ul style="list-style-type: none"> <li>• <code>srvrIdd_Diag1</code> contains bytes 1 - 4 of the specified MsgCntl.</li> <li>• <code>srvrIdd_Diag2</code> contains bytes 5 - 8 of the specified MsgCntl.</li> <li>• <code>srvrIdd_Diag3</code> contains bytes 9 - 12 of the specified MsgCntl.</li> <li>• <code>srvrIdd_Diag4</code> contains bytes 13 - 16 of the specified MsgCntl.</li> </ul> <p>Do not specify MSGCNTL (or specify MSGCNTL(0)) on the IXCSEND request that sends the request message to the XCF Server.</p>

Table 18. Return and Reason Codes for the IXCREQ Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	00050018	<p><b>Equate Symbol:</b> IXCREQRSNBADPLISTRSD</p> <p><b>Meaning:</b> Request message content is not valid. .</p> <p><b>Action:</b> Use the modify form of the IXCREQ macro to format an XCF Server request message. Make sure that the storage for the XCF Server request message has not been corrupted or overlaid.</p>
8	0006000C	<p><b>Equate Symbol:</b> IXCREQRSNOTHERSYSSERVERID</p> <p><b>Meaning:</b> Server ID represents a server instance that resides on some other system.</p> <p><b>Action:</b> Specify a valid server id for the instance.</p>
8	0007000C	<p><b>Equate Symbol:</b> IXCREQRSNBADSENDERFUNCTION</p> <p><b>Meaning:</b> Function specified on IXCSEND is not valid.</p> <p><b>Action:</b> When sending an IXCREQ REQUEST=SERVERINFO request to the XCF Server task, the value provided for the IXCSEND FUNCTION keyword must be "SRVRINFO". When this reason code is set, diagnostic information mapped by ixcysrvr_tSrvrInfoDD is returned by the XCF Server. See the following values in the mapping:</p> <ul style="list-style-type: none"> <li>• srvrIdd_Diag1 contains bytes 1 - 4 of the specified FUNCTION.</li> <li>• srvrIdd_Diag2 contains bytes 5 - 8 of the specified FUNCTION.</li> </ul>
8	000A000C	<p><b>Equate Symbol:</b> IXCREQRSNBADVALSERVERID</p> <p><b>Meaning:</b> Server ID value specified on the IXCREQ SERVERID keyword is not valid.</p> <p><b>Action:</b> Specify a valid server ID on the IXCREQ SERVERID keyword.</p>
C	000100CE	<p><b>Equate Symbol:</b> IXCREQRSNSYSTEMRESOURCES</p> <p><b>Meaning:</b> Failed to obtain storage to complete the request because of a system error.</p> <p><b>Action:</b> When this reason code is set, diagnostic information mapped by ixcysrvr_tSrvrInfoDD is returned by the XCF server. Check the following values in the mapping:</p> <ul style="list-style-type: none"> <li>• srvrIdd_Diag1 contains the IARV64 return code.</li> <li>• srvrIdd_Diag2 contains the IARV64 reason code.</li> </ul>

Table 18. Return and Reason Codes for the IXCREQ Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	000100EE	<p><b>Equate Symbol:</b> IXCREQRSNTOOMUCHDATA</p> <p><b>Meaning:</b> Request generates too much response data that exceeds allowable IXCSEND maximum.</p> <p><b>Action:</b> When this reason code is set, diagnostic information mapped by <code>ixcysrvr_tSrvrInfoDD</code> is returned by the XCF server. <code>srvrIdd_Diag1</code> contains the maximum amount of response data that can be sent by the XCF server on the target system. The amount of response data that can be sent is determined by the version of the IXCSEND service that the XCF server is using. Check the following value in the mapping:</p> <ul style="list-style-type: none"> <li>• <code>srvrIdd_Diag2</code> contains the amount of response data that was collected by the XCF server on the target system for the IXCREQ request.</li> </ul> <p>To reduce the amount of response data collected by the XCF Server, limit the scope of requested servers to collect data for.</p>

## Example

For an example of using IXCREQ, see the chapter "Using XCF for Client/Server Communication" of [z/OS MVS Programming: Sysplex Services Guide](#).





## Chapter 22. IXCSEND — Send Client/Server Requests and Responses

### Description

The IXCSEND macro is the interface to the XCF (cross-system coupling facility) client/server send service. The IXCSEND macro is one of the macros that comprise the XCF client/server interfaces. With these interfaces, a "client" can send a request to a "server" for processing and then receive the "results" provided by the server. Related macros include the XCF server interface (IXCSRVR) that is used to create servers, and the XCF receive service (IXCRECV) that is used to obtain the results provided by one or more of the servers. See [Chapter 24, "IXCSRVR — Define a Server to XCF,"](#) on page 385 and [Chapter 20, "IXCRECV— Receive Client/Server Information,"](#) on page 313.

A "client" uses the IXCSEND macro to send a request to one or more servers for processing. The "server" uses the IXCSEND macro to send the request results back to the client. The following summarizes the processing:

- Use the IXCSEND SENDTO=SERVER client request to send a message to one or more target systems. The XCF service routine handles the request and returns a token to represent the request. The token is used to identify the request when invoking other XCF client/server services (such as IXCRECV).
- On each target system, XCF presents a copy of the request to a suitable server instance for processing.
- When the server or server instance processes the request, it uses the IXCSEND SENDTO=ORIGINATOR request to respond and send the request results back to the client.
- If the server does not reply, XCF sends its own acknowledgment to indicate what happened to the request.

For guidance information and other details on using IXCSEND with the client/server interfaces, see the IXCSEND topic in the chapter "Using XCF for Client/Server Communication" of *z/OS MVS Programming: Sysplex Services Guide*.

### Environment

The following are the environment requirements for the caller.

Environment	Environment requirement
Minimum authorization:	Supervisor state or PKM allowing keys 0-7.
Dispatchable unit mode:	Task or SRB mode.
Cross memory mode:	Any PASN, any HASN, any SASN
AMODE:	31- or 64 bit. If in 64-bit mode, specify SYSSTATE AMODE64=YES before invoking this macro. If in 64-bit mode, specify SYSSTATE AMODE64=YES before invoking this macro.
ASC mode:	Primary or access register (AR)  If in Access Register ASC mode, specify SYSSTATE ASCENV=AR before invoking this macro.  Any primary address space; any secondary address space; any home address space.
Interrupt status:	Enabled for I/O and external interrupts
Locks:	No locks held

Environment	Environment requirement
Control parameters:	The storage containing the IXCSEND parameter list must reside in the primary address space of the caller, or in a space addressable through a public entry on the dispatchable unit access list (DU-AL), or in a common area data space.

## Programming Requirements

1. If the program is in AR mode, issue the SYSSTATE ASCENV=AR macro before issuing IXCSEND to generate code for AR mode.
2. If the parameter list specified for the IXCSEND macro service resides in 64-bit virtual storage above the 2-gigabyte bar, the caller must be executing in AMODE 64 when invoking the IXCSEND macro service.

## Restrictions

IXCSEND cannot be used by tasks higher in the task tree than the cross memory resource owning task (the top, or first, job step task in the address space).

Callers running in SRB mode should refrain from invoking the IXCSEND service under the following circumstances:

- After the SRB receives a x'47B' abend
- When running in a suspend exit after invoking SUSPEND

In these cases, the IXCSEND service routine may sometimes be unable to initiate the send of the message to one or more targets.

## Input Register Information

Before issuing the IXCSEND macro, the caller does not have to place any information into any register unless using it in register notation for a particular parameter, or using it as a base register.

## Output Register Information

When control returns to the caller, the GPRs contain:

### Register

#### Contents

**0**

Reason code, if GPR 15 is non-zero

**1**

Used as a work register by the system

**2-13**

Unchanged

**14**

Used as a work register by the system

**15**

Return code

When control returns to the caller, the access registers (ARs) contain:

### Register

#### Contents

**0-1**

Used as work registers by the system

**2-13**

Unchanged

**14-15**

Used as work registers by the system

Some callers depend on register contents remaining the same before and after issuing a service. If the system changes the contents of registers on which the caller depends, the caller must save them before issuing the service, and restore them after the system returns control.

## Performance Implications

---

To prevent backlogs that can degrade system performance, a server must be able to process server requests as fast, or faster than senders create server requests for a server name. Servers need to ensure that there are enough suitable server instances to process the volume of server requests sent from all systems in the sysplex. If requests are sent to a specific server ID, ensure that the designated server instance can process the received requests as fast, or faster than senders create the requests.

## Understanding IXCSEND Version Support

---

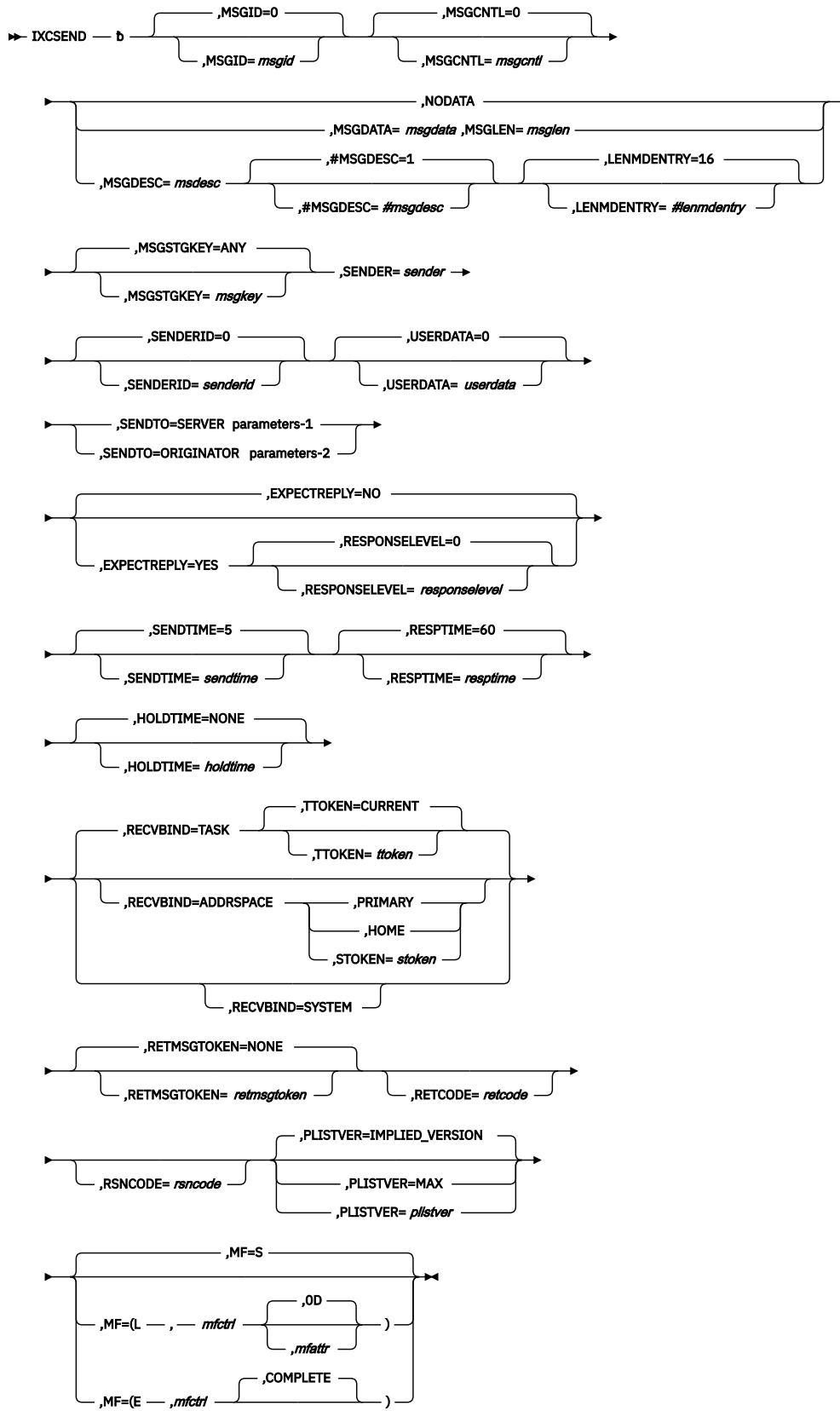
The IXCSEND macro supports version 0. See [Chapter 2, “Specifying a Macro Version Number,” on page 5](#) for considerations when specifying the version of the parameter list with PLISTVER.

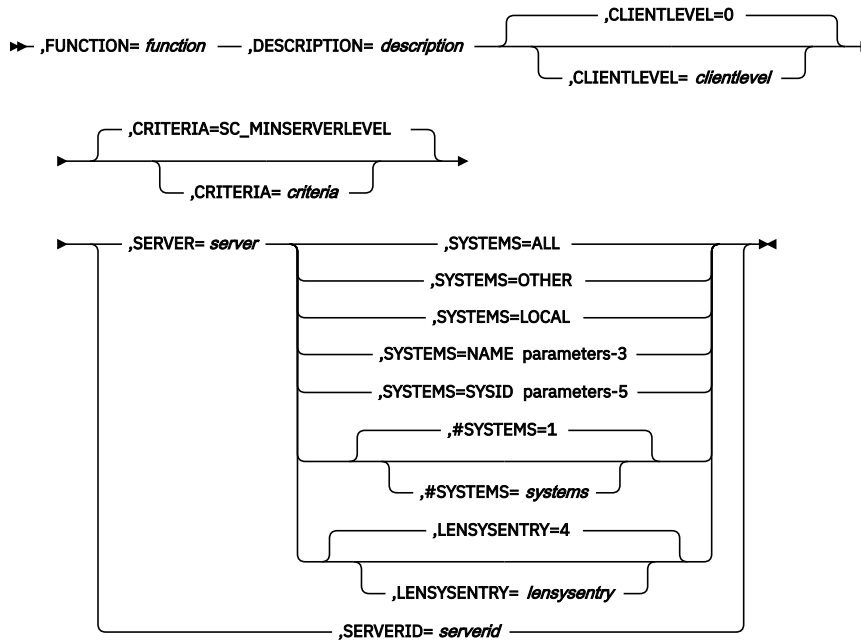
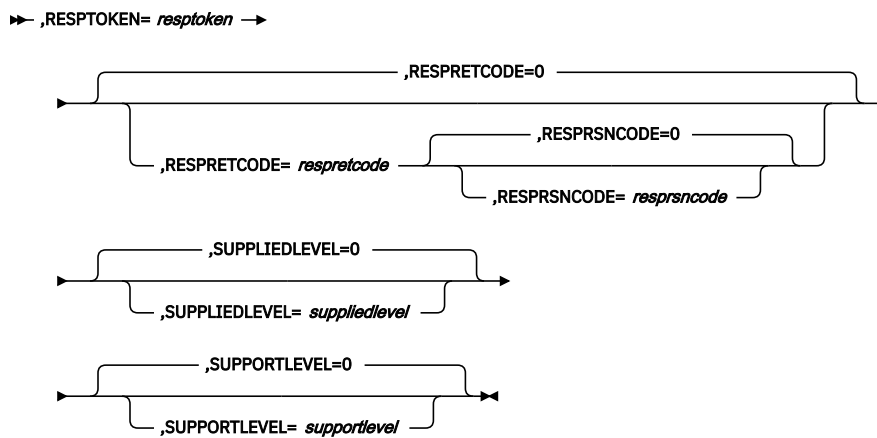
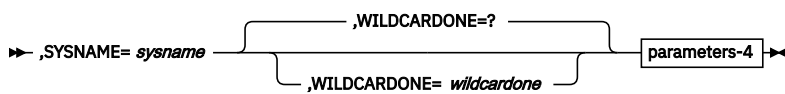
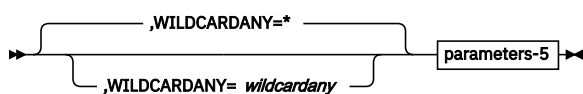
## Syntax

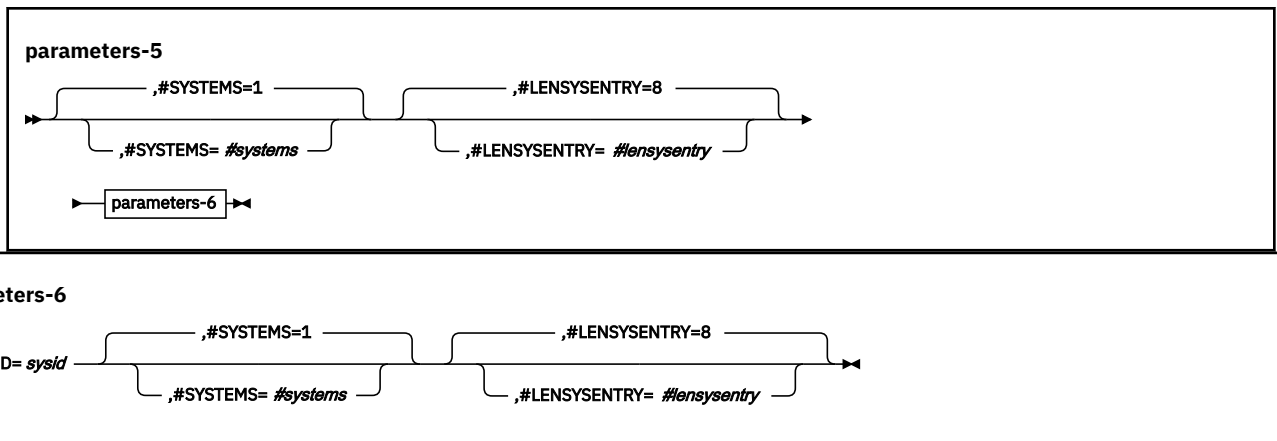
---

The IXCSEND macro is written as follows:

## main diagram



**parameters-1****parameters-2****parameters-3****parameters-4**



## Parameters

The parameters are explained as follows:

**,#MSGDESC=*#msgdesc***

**,#MSGDESC=1**

When `MSGDESC=msgdesc` is specified, use this optional input parameter to indicate the number of data descriptors in the message descriptor table. The default is 1.

**To code:** Specify the RS-type name or address in register (2)-(12), of a fullword field, or specify a literal decimal value.

**,#SYSTEMS=*#systems***

**,#SYSTEMS=1**

When `SYSTEMS=NAME`, `SERVER=server`, and `SENDTO=SERVER` are specified, use this optional input parameter to contain the number of system names specified by the `SYSNAME` keyword. The default is 1.

When `SYSTEMS=SYSID`, `SERVER=server`, and `SENDTO=SERVER` are specified, use this optional input parameter to contain the number of system IDs specified by the `SYSID` keyword. The default is 1.

**To code:** Specify the RS-type name or address in register (2)-(12), of a fullword field, or specify a literal decimal value.

**,CLIENTLEVEL=*clientlevel***

**,CLIENTLEVEL=0**

When `SENDTO=SERVER` is specified, use this optional input parameter to contain a number that identifies the level of the client. When the request arrives on one or more of the target systems, XCF presents the request only to those server instances that support the indicated client level. If no server supports the indicated level, the request is canceled because the target server does not have any instances that are suitable for processing a request from a client of `CLIENTLEVEL`.

The client and server mutually determine the content and interpretation of the client level. The intended purpose is to provide a protocol that helps clients and servers function compatibly with mixed release levels. When it invokes the `IXCSRVR` macro to define itself to XCF, a server specifies the range of client levels (`MINCLIENT` and `MAXCLIENT`) whose server level requests it is willing to accept (`MINLEVEL`, `MAXLEVEL`). An uplevel server can in effect refuse to process requests from a downlevel client.

The default is 0.

**To code:** Specify the RS-type name or address in register (2)-(12), of a fullword field, or specify a literal decimal value.

**,CRITERIA=*criteria***

**,CRITERIA=SC\_MINSERVERLEVEL,**

When `SENDTO=SERVER` is specified, use this optional input parameter mapped by `ixcysrvr_tCriteria` to define the range of server levels and set of features that the server must support and that are suitable

to the client to process the request. When a server is started, it specifies the range of server levels it supports (MINLEVEL, MAXLEVEL), the range of client levels whose requests it is willing to process (MINCLIENT, MAXCLIENT), and the features that it supports (FEATURES) when it invokes the IXCSRVR macro to define itself to XCF.

- Specifying server levels

The meaning of "server level" is defined by the server. The intended purpose is to provide a protocol that helps clients and servers function compatibly with mixed release levels. An uplevel client can specify a minimum Server Level (sc\_MinSeverLevel) to ensure that XCF does not route the request to a downlevel server. An uplevel server can in effect refuse to accept requests intended for a downlevel server instance.

The sc\_MinServerLevel field of the ixcsrvr\_tCriteria mapping contains a number that identifies the minimum server level required to process the request.

The sc\_MaxServerLevel field of the ixcsrvr\_tCriteria mapping contains a number that identifies the maximum server level that is suitable for processing the request.

If non-zero, sc\_MaxServerLevel must be greater than or equal to sc\_MinServerLevel. If sc\_MaxServerLevel is zero (0), it defaults to the value used for sc\_MinServerLevel.

The combination of minimum server level (sc\_MinServerLevel) and maximum server level (sc\_MaxServerLevel) defines the range of server levels that are acceptable to the client. When the request arrives on one or more of the target systems,, XCF presents the request only to those server instances that support a server level within the range requested by the client. If no server supports such a level, the request is cancelled with a "no receiver" response code.

- Specifying server features

The sc\_features feature string indicates the set of features that the server must support in order to process the request. XCF uses the indicated specification to determine whether a particular server instance is capable of processing the request. If no server supports the required features, the request is cancelled with a "no receiver" response code.

The feature string and how it is used by XCF to select a server instance is defined in the IXCYSRVR macro. (See the declare for ixcsrvr\_tFeatures.) However, the content and interpretation of the "feature level" and the "feature flags" within the feature string is defined by the server.

The default for SC\_MINSERVERLEVEL, sc\_MaxServerLevel and sc\_Features. are all zeroes (0).

**To code:** Specify the RS-type name or address in register (2)-(12), of a character field mapped by ixcsrvr\_tCriteria.

### **,DESCRIPTION=description**

When SENDTO=SERVER is specified, use this required input parameter to contain a description of the request. The description might indicate the function, service, purpose, or role of the client. Or it might indicate the application with which the client is associated. Or it might describe the request or the reason the request is being issued. The string can contain any alphanumeric (a-z, A-Z, 0-9), national (@, #, \$), or special (underscore or blank) character. Leading blanks and all blank descriptors are not permitted. Descriptions are case sensitive. XCF presents the client description to the server as part of the request. The description also appears in various XCF messages and diagnostic data reports. The intended purpose is to help installations and service personnel understand how the system (or sysplex) might be impacted if there are problems with this request.

**To code:** Specify the RS-type name or address in register (2)-(12), of a 32-character field.

### **,EXPECTREPLY=NO**

### **,EXPECTREPLY=YES**

Use this optional parameter to indicate whether the sender is expecting the target to send a reply in response to this message. The default is EXPECTREPLY=NO.

### **,EXPECTREPLY=NO**

The target is not expected to reply to this message.

For a SENDTO=SERVER request with a nonzero HOLDTIME, an XCF acknowledgment will be sent to confirm whether or not the message was delivered to the target. If an XCF acknowledgment is received before RESPTIME expires, the information returned by IXCRECV will provide the status of the delivery. (See field td\_RespCode in macro IXCYSRVR.) If instead HOLDTIME is zero, no XCF acknowledgments are sent.

**,EXPECTREPLY=YES**

The sender expects the target to reply to this message. EXPECTREPLY=YES can only be specified for a SENDTO=SERVER request.

Note that an XCF acknowledgment might be sent as well. Thus the status information returned by IXCRECV could indicate that the message was delivered to the target even though a response was not received from the target before the RESPTIME expired.

**,FUNCTION=function**

When SENDTO=SERVER is specified, use this required input parameter to indicate the request or function that the server is to perform. The content and interpretation of FUNCTION is defined by the server.

**To code:** Specify the RS-type name or address in register (2)-(12), of an 8-character field.

**,HOLDTIME=holdtime**

**,HOLDTIME=NONE**

Use this optional input parameter to indicate the maximum number of seconds that XCF preserves message results for retrieval using the IXCRECV service after the message completes. When the HOLDTIME expires, the message and its results are eligible for discard. Once discarded, any attempt to inspect the responses fails. If specified, the HOLDTIME value must be between 0 and 3600, inclusive. If not specified, XCF discards the message as soon as it is complete.

When a non-zero HOLDTIME value is specified, a RETMSGTOKEN is required in order to retrieve IXCSEND results using IXCRECV. You must specify RETMSGTOKEN on the IXCSEND invocation when a non-zero HOLDTIME is specified. The default is NONE.

**To code:** Specify the RS-type name or address in register (2)-(12), of a fullword containing the HOLDTIME in seconds that XCF preserves message results.

**,LENMDENTRY=lenmdentry**

**,LENMDENTRY=16**

When MSGDESC=msgdesc is specified, use this optional input parameter to indicate the length in bytes of each entry in the message descriptor table. As XCF iterates through the table, it locates the next descriptor by adding LENMDENTRY to the location of the current descriptor.

LENMDENTRY must be greater than or equal to the length of one data descriptor, which is the default value of 16. The length must be a decimal value greater than or equal to 16. If not specified, LENMDENTRY defaults to the length of one message descriptor (16). The default is 16.

**To code:** Specify the RS-type name or address in register (2)-(12), of a fullword field, or specify a literal decimal value.

**,LENSYSENTRY=lensysentry**

**,LENSYSENTRY=8|4**

When SYSTEMS=NAME or SYSTEM=SYSID are specified, use this optional input parameter to contain the length of the array entries that contain a *sysname* or system ID. XCF locates successive array entries by adding this length to the address of the current array entry.

LENSYSENTRY defaults to 8 if SYSTEMS=NAME is specified. LENSYSENTRY defaults to 4 if SYSTEMS=SYSID is specified.

**To code:** Specify the RS-type name or address in register (2)-(12), of a fullword field, or specify a literal decimal value.



**,MF=S**  
**,MF=(L,list addr)**  
**,MF=(L,list addr,attr)**  
**,MF=(L,list addr,OD)**  
**,MF=(E,list addr)**  
**,MF=(E,list addr,COMPLETE)**

Use this optional input parameter to specify the macro form.

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service. MF=S is the default.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter might be coded with the list form of the macro.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form of the macro stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

**,list addr**

The name of a storage area to contain the parameters. For MF=S and MF=E, this can be an RS-type address or an address in register (1)-(12).

**,attr**

An optional 1- to 60-character input string that you use to force boundary alignment of the parameter list. Use a value of 0F to force the parameter list to a word boundary, or 0D to force the parameter list to a doubleword boundary. If you do not code *attr*, the system provides a value of 0D.

**,COMPLETE**

Specifies that the system is to check for required parameters and supply defaults for omitted optional parameters.

**,MSGCNTL=msgcntl**

**,MSGCNTL=0**

Use this optional input parameter to contain "message control data". This information is to be presented to the receiver as part of the metadata associated with the message. The content and interpretation of the message controls is up to the sender. The intended purpose is to allow the sender to provide control data to influence the processing performed by the receiver. Depending on the protocols employed by the sender and receiver, the message control data could in fact be the message (as opposed to information about the message). The default is 0.

**To code:** Specify the RS-type name or address in register (2)-(12), of a 64-character field.

**MSGID=msgid**

**MSGID=0**

Use this optional input parameter to contain a message ID that identifies this particular message. The content and interpretation of MSGID is defined by the sender. It is intended that MSGID be a value that uniquely identifies the message from the sender perspective. The message ID is preserved in the XCF control structures for the message. It is also presented to the recipient. If problems arise, the message ID can aid diagnosis by helping service personnel correlate XCF control blocks and traces with the control blocks and traces of the sender. The default is 0.

**To code:** Specify the RS-type name or address in register (2)-(12), of a 16-character field.

**,MSGLLEN=msglen**

When MSGDATA=msgdata is specified, use this required input parameter to contain the length in bytes of the message. The length must be a decimal value in the range 0 to 104,857,600 bytes (100MB), inclusive.

**To code:** Specify the RS-type name or address in register (2)-(12), of a fullword field, or specify a literal decimal value.

**,MSGSTGKEY=msgstgkey**

**,MSGSTGKEY=ANY**

Use this optional input parameter to contain the storage key to be used when fetching the message data. The message data is fetched from either the storage area indicated by the MSGDATA keyword, or the storage area identified by a data descriptor in the message data descriptor table indicated by the MSGDESC keyword. The high order nibble contains the storage key, the low order nibble is ignored. For example, set 'kkkk' in the binary bit string 'kkkkxxxx'B to correspond to the desired storage key.

If MSGSTGKEY is not specified, the message data can reside in any storage key.

Note that key-controlled protection will permit the message data to be moved successfully if either the access key specified by MSGSTGKEY matches the storage protect key of the indicated storage areas, or if the indicated storage areas are not fetch protected.

The default is ANY - message data can reside in any storage key.

**To code:** Specify the RS-type name or address in register (2)-(12), of an 8 bit field.

**,NODATA**

**,MSGDATA=msgdata**

**,MSGDESC=msgdesc**

Use one of these mutually exclusive required input parameters to indicate information about the message data.

**,NODATA**

A parameter that indicates that the message does not have any data. Only the metadata for the message is presented to the receiver.

**,MSGDATA=msgdata**

A parameter that identifies a single contiguous buffer variable which contains the message data to be delivered to the receiver.

The message data buffer can reside in the primary address space of the caller, or in a space addressable through a public entry on the dispatchable unit access list (DU-AL), or in a common area data space. The storage key of the storage area containing the message data must match the storage protect key indicated by the MSGSTGKEY keyword.

**To code:** Specify the RS-type name or address in register (2)-(12), of the storage area containing the message data .

**,MSGDESC=msgdesc**

A parameter that contains a table of one or more data descriptors. The message descriptor table is used when the message data resides in discontinuous storage areas. A data descriptor identifies a storage location from which XCF is to obtain a portion of the message data. Data descriptors are mapped by `ixcysrvr_tDataDescriptor` which is defined in the `IXCYSRVR` macro. A data descriptor specifies the ALET, address, and length of a contiguous virtual storage area where the message data resides.

XCF concatenates each component of the message end to end and presents a copy of the result to the receiver. So for example, if the table had two entries whose descriptors identified storage containing the five byte strings 'PART1' and 'PART2' respectively, the message data presented to the receiver would be the ten byte string 'PART1PART2'.

The storage area defined by each data descriptor must reside in the primary address space of the caller, or in a space addressable through a public entry on the dispatchable unit access list (DU-AL), or in a common area data space. The storage key of the storage area containing the message data indicated by each data descriptor must match the storage protect key indicated by the MSGSTGKEY keyword.

The storage area containing the data descriptors must reside in the primary address space of the caller, or in a space addressable through a public entry on the dispatchable unit access list (DU-AL), or in a common area data space.

**To code:** Specify the RS-type name or address in register (2)-(12) of a character input variable containing a table of one or more data descriptors.

**,PLISTVER=IMPLIED\_VERSION****,PLISTVER=MAX****,PLISTVER=0**

Use this input parameter to specify the version of the macro. See [“Understanding IXCSEND Version Support”](#) on page 339 for a description of the options available with PLISTVER.

**,PRIMARY****,HOME****,STOKEN=*stoken***

When RECVBIND=ADDRSPACE is specified, use this required input parameter to indicate the responsibility of the invoker of the IXCRECV request.

**,PRIMARY**

A parameter keyword that belongs to a set of mutually exclusive keys used with RECVBIND=ADDRSPACE. PRIMARY indicates that responsibility for invoking IXCRECV is to be assigned to the primary address space of the sender.

**,HOME**

A parameter keyword that belongs to a set of mutually exclusive keys used with RECVBIND=ADDRSPACE. HOME indicates that responsibility for invoking IXCRECV is to be assigned to the home address space of the sender.

**,STOKEN=*stoken***

A parameter keyword that belongs to a set of mutually exclusive keys used with RECVBIND=ADDRSPACE. STOKEN contains the space token of the address space to which responsibility for invoking IXCRECV is to be assigned.

**To code:** Specify the RS-type name or address (using a register from 2 to 12) of the 8-byte storage area containing a space token.

**,RECVBIND=TASK****,RECVBIND=ADDRSPACE****,RECVBIND=SYSTEM**

Use this optional parameter to indicate the entity that is responsible for issuing an IXCRECV to inspect the results of the IXCSEND request.

RECVBIND is required to be specified when HOLDTIME is non-zero for SRB mode callers.

The RECVBIND specification helps XCF ensure that system resources are recovered in a timely manner if the intended invoker of IXCRECV fails before it can process the message results. The default is RECVBIND=TASK.

**,RECVBIND=TASK**

Responsibility for issuing the IXCRECV is to be assigned to the indicated task. SRB mode callers must explicitly specify a TTOKEN other than CURRENT or 0 when assigning responsibility for issuing the IXCRECV to a task (RECVBIND=TASK).

**,RECVBIND=ADDRSPACE**

Responsibility for invoking IXCRECV is to be assigned to the indicated address space.

**,RECVBIND=SYSTEM**

The IXCRECV will be issued from the local system. XCF is not to establish a receive bind to any particular address space or task. If the sender fails such that IXCRECV is never invoked, the message persists until its hold time expires (HOLDTIME).

**,RESPONSELEVEL=*responselevel*****,RESPONSELEVEL=0**

When EXPECTREPLY=YES is specified, use this optional input parameter to specify the level of the response data that the sender wants the recipient to return. The content and interpretation of the response level is defined by the receiver. The response level is made available when the message is presented to the target server. For example, when presenting a client request to a server exit, the server exit parameter list contains the level of response data that the sender wants the recipient to send.

The sender and receiver can use response level in any way they choose. The intended purpose is to provide a protocol that helps clients and servers function compatibly with mixed release levels. For example, the sender might specify the level of response data that it would like to receive, and the receiver might send a reply with data at a level no greater than what the sender requested. Thus an uplevel target would not provide results containing data that a downlevel sender is not prepared to handle. A downlevel receiver asked to provide data at a response level it does not understand might choose to reject the request of the uplevel sender outright, or it might choose to provide a response at the highest level it can. The default is 0.

**To code:** Specify the RS-type name or address in register (2)-(12), of a fullword field, or specify a literal decimal value.

**,RESPRETCODE=respcode**

**,RESPRETCODE=0**

When SENDTO=ORIGINATOR is specified, use this optional input parameter to contain the return code to be provided to the target along with the response. This return code will be placed in the rd\_RespRetCode field of the ANSAREA returned by the IXCRECV macro when the target gathers the response. The content and interpretation of the return code is defined by the sender. The default is 0.

**To code:** Specify the RS-type name or address in register (2)-(12), of a fullword field, or specify a literal decimal value.

**,RESPRSNCODE=resprsncode**

**,RESPRSNCODE=0**

When RESPRETCODE=respcode and SENDTO=ORIGINATOR are specified, use this optional input parameter to contain the reason code to be provided to the target along with the response. This reason code will be placed in the rd\_RespRsnCode field of the ANSAREA returned by the IXCRECV macro when the target gathers the response. The content and interpretation of the reason code is defined by the server. The default is 0.

**To code:** Specify the RS-type name or address in register (2)-(12), of a fullword field, or specify a literal decimal value.

**,RESPTIME=resptime**

**,RESPTIME=60**

Use this optional input parameter to indicate the amount of time the sender is willing to wait for the expected responses to arrive. For EXPECTREPLY=YES, the expected response is a reply sent by the target server. For EXPECTREPLY=NO with a nonzero HOLDTIME, the expected response is an acknowledgment sent by XCF. If the expected responses do not arrive within the specified RESPTIME, XCF deems the request to have completed and starts the HOLDTIME timer.

RESPTIME also determines the amount of time XCF attempts to send messages that are pending due to system conditions. If the RESPTIME expires before the pending messages can be sent to the targets, the pending messages are cancelled and XCF deems the request to have completed.

The timeout value must be between 1 and 3600, inclusive. The default is 60.

**To code:** Specify the RS-type name or address in register (2)-(12), of a fullword field, or specify a literal decimal value.

**,RESPTOKEN=resptoken**

When SENDTO=ORIGINATOR is specified, use this required input parameter to contain the token that identifies the originating message to which this response is being sent. The response token was made available when the originating message was presented to the receiver. For example, the SXPLRQ\_RespToken field in the server exit parameter list (macro IXCYSRVR) contains the response token to be used when sending results back to the requesting client.

**To code:** Specify the RS-type name or address in register (2)-(12), of a 64-character field.

**,RETCODE=retcode**

Use this optional output parameter into which the return code is to be copied from GPR 15. If you specify 15, GPR15, REG15, or R15 (within or without parentheses), the value will be left in GPR 15.

**To code:** Specify the RS-type name of a fullword field, or register (2)-(12) or (15), (GPR15), (REG15), or (R15).

**,RETMSGTOKEN=retmsgtoken**

**,RETMSGTOKEN=NONE**

Use this output parameter to specify a storage area to contain a token that identifies the message. For recovery purposes, the sender should initialize the storage to hexadecimal zero before invoking IXCSEND to send a message. The token is required for use with other XCF client/server interfaces and the XCF message control service (IXCMMSGC). For example it is used with the IXCRECV macro to identify the message whose results are to be received.

When a non-zero HOLDTIME is specified, a RETMSGTOKEN is required in order to retrieve IXCSEND results using IXCRECV. You must specify RETMSGTOKEN on the IXCSEND invocation when a non-zero HOLDTIME is specified.

The storage for the message token can reside in the primary address space of the caller, or in a space addressable through a public entry on the dispatchable unit access list (DU-AL), or in a common area data space.

XCF stores the message token before the IXCSEND service returns to the caller. Non-zero RetMsgToken storage indicates that the message token is available for use.

In cases where the calling work unit must be suspended, the token is stored before the work unit is actually suspended. If it should become necessary for the sender to break out of the suspend before the service routine resumes and returns of its own accord, some other work unit can use this token to cancel the request by invoking the XCF message control service (IXCMMSGC). Note that cancelling the send likely implies that the message will not be delivered to all of the intended targets. The default is NONE.

**To code:** Specify the RS-type name or address in register (2)-(12), of a 32-byte character output field to contain the message token

**,RSNCODE=rsncode**

Use this optional output parameter into which the reason code is to be copied from GPR 0. If you specify 0, 00, GPR0, GPR00, REG0, REG00, or R0 (within or without parentheses), the value will be left in GPR 0.

**To code:** Specify the RS-type name of a fullword field, or register (0) or (2)-(12), (00), (GPR0), (GPR00), REG0), (REG00), or (R0).

**,SENDER=sender**

Use this required input parameter to contain the name of the sender. When sending a request to a server for processing, the client is the sender. When sending a response containing request results back to the client, the server is the sender. XCF includes the sender name as part of the metadata presented to the receiver of the message. The sender name also appears in various XCF messages and diagnostic data reports.

Sender names consist of four 8-byte sections. Each 8 byte section must be left justified, padded on the right with EBCDIC blanks as needed. Each section string can contain any alphanumeric (a-z, A-Z, 0-9), national (@, #, \$), or underscore character (\_). Any section but the first can be entirely blank. Names are case sensitive. The IXCYSRVR macro defines a mapping for client and server names (ixcysrvr\_tName).

To avoid names used by IBM, do not begin names (section 1) with the letters A through I or the character string SYS.

**To code:** Specify the RS-type name or address in register (2)-(12), of a 32-character field.

**,SENDERID=senderid**

**,SENDERID=0**

Use this optional input parameter to contain the sender ID. The sender ID is presented to the target of the message. The intended purpose is to allow the sender to provide a token to uniquely identify itself. For example, a server might specify its server ID as the SENDERID when sending a response to a client request. The default is 0.

**To code:** Specify the RS-type name or address in register (2)-(12), of a 16-character field.

**,SENDTIME=sendtime**

**,SENDTIME=5**

Use this optional input parameter to indicate the maximum number of seconds XCF might pause the work unit to allow XCF to synchronously complete accessing callers storage areas.

If the SENDTIME expires before the processing that XCF paused for is complete, XCF will cancel incomplete send processing which might result in not all desired targets receiving the send request.

The timeout value must be between 1 and 3600, inclusive. The default is 5.

**To code:** Specify the RS-type name or address in register (2)-(12), of a fullword field, or specify a literal decimal value.

**,SENDTO=SERVER**

**,SENDTO=ORIGINATOR**

Use this required parameter to indicate the target (receiver) of the message.

**,SENDTO=SERVER**

Send a request to a server on one or more systems for processing.

**,SENDTO=ORIGINATOR**

Send a response to the indicated message.

**,SERVER=server**

**,SERVERID=serverid**

When SENDTO=SERVER is specified, use one of these mutually exclusive required input parameters to specify the name or server id of the server.

**,SERVER=server**

A parameter variable containing the name of the server that is to process the request. Server names are mapped by ixcysrvr\_tName (macro IXCYSRVR). Server names consist of four 8-byte sections. Each 8 byte section must be left justified, padded on the right with EBCDIC blanks as needed. Each section string can contain any alphanumeric (a-z, A-Z, 0-9), national (@, #, \$), or underscore character. Any section but the first can be entirely blank. Server names are case sensitive.

**To code:** Specify the RS-type name or address in register (2)-(12), of a 32-character field.

**,SERVERID=serverid**

A parameter variable that contains the server ID of the target server. The request is sent to the system on which the indicated server instance resides. When the request arrives on the target system, the request is presented to the indicated server instance for processing. The client request is presented to the specified server instance without comparing client specified server selection criteria. If the server instance no longer exists, the request is not processed and the response code indicates "no receiver."

A client might use the SERVERID if it has a set of requests that must be processed by the same server instance. SERVERID might also be used if the algorithms used by XCF to select a server are not appropriate for the client/server application.

A server instance can choose to communicate its SERVERID to a client in various ways:

- As part of the message content sent in response to a request
- As part of the MSGCNTL
- As the SENDERID when sending a response to a client request.

The IXCREQ REQUEST=SERVERINFO interface can also be used to obtain server ids for server instances that have started on the local system or on any other system in the sysplex.

**To code:** Specify the RS-type name or address in register (2)-(12), of a 16-character field.

**,SUPPLIEDLEVEL=suppliedlevel****,SUPPLIEDLEVEL=0**

When SENDTO=ORIGINATOR is specified, use this optional input parameter to indicate the "level" of response data being provided by the sender. The indicated level is placed in the rd\_SuppliedLevel field of the ANSAREA returned by the IXCRECV macro when the target gathers the response. The content and interpretation of this data is defined by the sender. The intended purpose is to provide a protocol that helps clients and servers function compatibly with mixed release levels. For example, a client might request "level 2" results (see the RESPONSELEVEL keyword), but the server might provide result data at some other level. Thus the results might include more or less data than requested. The default is 0.

**To code:** Specify the RS-type name or address in register (2)-(12), of a fullword field, or specify a literal decimal value.

**,SUPPORTSLEVEL=supportslevel****,SUPPORTSLEVEL=0**

When SENDTO=ORIGINATOR is specified, use this optional parameter to indicate the maximum level of response data that the sender could provide. The indicated level is placed in the rd\_SupportsLevel field of the ANSAREA returned by the IXCRECV macro when the target gathers the response. The content and interpretation of this data is defined by the sender. The intended purpose is to provide a protocol that helps clients and servers function compatibly with mixed release levels. For example, a client might request the lowest possible result level knowing that every server can provide that data. The server might then supply the requested "level 0" data, and indicate that it supports data as high as "level 3". The client might then modify the response level requested for future requests. The default is 0.

**To code:** Specify the RS-type name or address in register (2)-(12), of a fullword field, or specify a literal decimal value.

**,SYSID=sysid**

When SYSTEMS=SYSID, SERVER=server, and SENDTO=SERVER are specified, use this required input parameter to contain the XCF system ID of the target system.

If #SYSTEMS is greater than 1, SYSID is the first entry in an array of system IDs. XCF iterates through each array entry and includes the indicated system as a target only if the system is known to the sysplex when the request is accepted. Excluded systems will not appear in any further information that XCF provides with respect to the message. At most one instance of the message is sent to any one system.

The XCF query service (IXCQUERY REQINFO=SYSPLEX) can be used to obtain the names of systems in the sysplex. The service returns a record (QUASYS) for each system in the sysplex. Within the QUASYS record, the field QUASSID contains the system ID.

**To code:** Specify the RS-type name or address in register (2)-(12), of a fullword field, or specify a literal decimal value.

**,SYSNAME=sysname**

When SYSTEMS=NAME, SERVER=server, and SENDTO=SERVER are specified, use this required input parameter to contain a system name. The system name must be 8 characters, left justified, and padded on the right with blanks as needed.

System names must contain characters that are valid for a system name and might also contain valid wildcard characters which can be used for wildcard character matching.

Wildcard characters are defined by the WILDCARDONE and WILDCARDANY keywords.

Since system names are not case sensitive, the system names are folded to upper case for comparison.

If #SYSTEMS is greater than 1, SYSNAME is the first entry in an array of system names. XCF iterates through each 8 byte entry and includes one or more of the indicated systems as a target only if the system is known to the sysplex when the request is accepted. Excluded systems will not appear in any further information that XCF provides with respect to the message. At most one instance of the message is sent to any one system.

The XCF query service (IXCQUERY REQINFO=SYSPLEX) can be used to obtain the names of systems in the sysplex. The service returns a record (QUASYS) for each system in the sysplex. Within the QUASYS record, the field QUASNAME contains the system name.

**To code:** Specify the RS-type name or address in register (2)-(12), of an array of one or more contiguous character fields, each of which is of length LENSYSENTRY.

**,SYSTEMS=ALL**

**,SYSTEMS=OTHER**

**,SYSTEMS=LOCAL**

**,SYSTEMS=NAME**

**,SYSTEMS=SYSID**

When SERVER=*server* and SENDTO=SERVER are specified, use this required parameter to indicate which systems in the sysplex are candidates for processing the request. XCF sends a copy of the message to each of the indicated systems (provided the system supports the XCF client/server interfaces). When the message arrives, XCF presents the message to a suitable instance of the indicated server (if any).

**,SYSTEMS=ALL**

Send the request to every system in the sysplex.

**,SYSTEMS=OTHER**

Send the request to all other systems in the sysplex. Do not send the request to the local system.

**,SYSTEMS=LOCAL**

Send the request to the local system only. Do not send the request to any other system in the sysplex.

**,SYSTEMS=NAME**

Select the candidates to send the request to by system name using the provided system name array indicated by SYSNAME. XCF processes each candidate system name to determine if the system is active in the sysplex and whether the system supports the XCF client/server protocols. Duplicates of system names are ignored. Only one instance of a system name is to be used as a candidate target. System names not chosen as candidates are not provided to the IXCSEND requester. A target descriptor record for each system to which a request was sent is returned in the answer area provided when issuing the XCF receive service (IXCRECV) to obtain the results of the request.

**,SYSTEMS=SYSID**

Select the candidates to send the request to XCF by XCF system ID (SYSID).

**,TTOKEN=*ttoken***

**,TTOKEN=CURRENT**

When RECVBIND=TASK is specified, use this optional input parameter to contain the task token of the task that is responsible for invoking IXCRECV. If the TTOKEN is not specified, responsibility is assigned to the current task under which the caller is running. Note that SRB mode callers must explicitly specify a TTOKEN other than CURRENT or 0 when assigning responsibility for issuing the IXCRECV to a task. The default is CURRENT.

**To code:** Specify the RS-type name or address in register (2)-(12), of a 16-character field.

**,USERDATA=*userdata***

**,USERDATA=0**

Use this optional input parameter to contain user data that is to be associated with the message. A copy of the user data is placed in the sd\_UserData field of the send descriptor in the ANSAREA returned by the IXCRECV macro when the sender inspects the results of this message. The data is not delivered to the target. The content and interpretation of the user data is defined by the sender. For example, user data might be used to locate control information used by the sender to manage the message. The default is 0.

**To code:** Specify the RS-type name or address in register (2)-(12), of a 16-character field.



**,WILDCARDANY=wildcardany****,WILDCARDANY=\***

When SYSTEMS=NAME, SERVER=server, and SENDTO=SERVER are specified, use this optional input parameter to contain a wild card character to be used for pattern matching on the system names. WILDCARDANY indicates the character that matches zero or more characters. By default, an asterisk ('\*') within the system name will match any sequence of zero or more characters. For example the pattern 'S\*D ' would match any system name that began with 'S' and ended with 'D', including 'SD', 'SYD', and 'SYSD' but none of 'S', 'D', 'SD1', or 'SYD1'.

The WILDCARDANY character cannot be:

- Alphabetic characters (A-Z, a-z)
- Numeric characters (0-9)
- National characters (@, #, \$)
- ampersand ( & ) or underscore ( \_ )
- Blank
- Equal to WILDCARDONE

A value of decimal zero (0) will indicate that the default of asterisk ('\*') is to be used as the WILDCARDANY character. The default is \*.

**To code:** Specify the RS-type name or address in register (2)-(12), of a 1-character field.

**,WILDCARDONE=wildcardone****,WILDCARDONE=?**

When SYSTEMS=NAME, SERVER=server and SENDTO=SERVER are specified, use this optional input parameter to contain a wild card character to be used for pattern matching on the system names. WILDCARDONE indicates the character that matches any one character. By default, a question mark ('?') within the system name will match any one character. For example, the pattern 'S?D ' would match any three character name that began with 'S' and ended with 'D', such as 'SYD' but none of 'SD', 'SD1', 'SYD1', or 'SYSD'.

The WILDCARDONE character cannot be:

- Alphabetic characters (A-Z, a-z)
- Numeric characters (0-9)
- National characters (@, #, \$)
- ampersand ( & ) or underscore ( \_ )
- Blank
- Equal to WILDCARDANY

A value of decimal zero (0) will indicate that the default of question mark ('?') be used as the WILDCARDONE character. The default is ?.

**To code:** Specify the RS-type name or address in register (2)-(12), of a 1-character field.

## ABEND Codes

---

Abend codes an issuer of IXCSEND might receive are listed below. For detailed abend code information, see *z/OS MVS System Codes*.

- Abend X'073' - Environment not valid. The caller held a lock.
- Abend X'C78' - XCF could not obtain storage to process the request. Try again later.

## Return and Reason Codes

---

When the IXCSEND macro returns control to your program:

- GPR 15 (and *retcode*, when you code RETCODE) contains a return code.

- When the value in GPR 15 is not zero, GPR 0 (and *rsncode*, when you code RSNCODE) contains a reason code.

The following table identifies the hexadecimal return and reason codes.

<i>Table 19. Return and Reason Codes for the IXCSEND Macro</i>		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0	None	Successful completion. The message has been accepted for delivery to each selected target. Issue the XCF client/server receive service (IXCRECV) to obtain the results of the IXCSEND request.
8	00010004	<b>Equate Symbol:</b> IXSENDRSNBADSTGPLIST <b>Meaning:</b> Program error. XCF could not access the control parameter list. <b>Action:</b> Ensure that: <ul style="list-style-type: none"> <li>• The correct parameter list storage area was specified.</li> <li>• The parameter list storage area was not inadvertently freed by your program.</li> </ul>
8	00010008	<b>Equate Symbol:</b> IXSENDRSNBADALETPLIST <b>Meaning:</b> Program error. The ALET that qualifies the address of the parameter list is neither zero nor a valid entry on the caller's Dispatchable Unit <b>Action:</b> Take one or more of the following actions: <ul style="list-style-type: none"> <li>• If your program is running in AR mode, ensure that you specified SYSSTATE ASCENV=AR before issuing the IXCSEND macro.</li> <li>• Ensure that the ALET for the control parameter list storage is correct.</li> </ul>
8	0001000C	<b>Equate Symbol:</b> IXSENDRSNBADVALMSGLEN <b>Meaning:</b> Program error. The message length (MSGLEN) or the sum of the data size values (dd_DataSize) specified in message descriptors exceeds the allowable IXCSEND message length of 100MB. The message length is not in the decimal range of 0 to 104857600. <b>Action:</b> Action: Take one or more of the following actions: <ul style="list-style-type: none"> <li>• Ensure that the correct message length is specified.</li> <li>• Ensure that the parameter list was not inadvertently overlaid and that it was correctly specified.</li> </ul> Decrease the length of the message being sent and retry the IXCSEND request.

Table 19. Return and Reason Codes for the IXCSEND Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	00010018	<p><b>Equate Symbol:</b> IXSENDRSNBADPLISTVERSION</p> <p><b>Meaning:</b> Program error. The version number in the control parameter list was not valid.</p> <p><b>Action:</b> Check for errors such as the following:</p> <ul style="list-style-type: none"> <li>• Your program overlaid the control parameter list storage.</li> <li>• Your program was assembled with the wrong macro library for the release of z/OS on which your program is running.</li> </ul>
8	000100EE	<p><b>Equate Symbol:</b> IXSENDRSNBADENVNOTENABLE</p> <p><b>Meaning:</b> Program error. The caller is not enabled.</p> <p><b>Action:</b> Correct your program so that it does not issue IXCSEND while it is disabled.</p>
8	0002000C	<p><b>Equate Symbol:</b> IXSENDRSNBADVALSENDER</p> <p><b>Meaning:</b> Program error. Sender Name (SENDER) value is not valid.</p> <p><b>Action:</b> Consider the following for sender names:</p> <ul style="list-style-type: none"> <li>• Sender names consist of four 8-byte sections. Each 8-byte section must be left justified, padded on the right with EBCDIC blanks as needed.</li> <li>• Each section string can contain any alphanumeric (a-z, A-Z, 0-9), national (@,#,\$), or underscore characters (_).</li> <li>• Any section but the first can be entirely blank.</li> <li>• Sender name is case sensitive.</li> </ul> <p>The IXCYSRVR macro defines a mapping for client and server sender names (ixcysrvr_tName). Correct the sender name, and retry the request.</p>
8	000200EE	<p><b>Equate Symbol:</b> IXSENDRSNBADENVLOCKED</p> <p><b>Meaning:</b> Program error. The caller of IXCSEND holds a lock.</p> <p><b>Action:</b> Correct your program so that it does not issue IXCSEND while it is holding a lock.</p>
8	00030004	<p><b>Equate Symbol:</b> IXSENDRSNBADSTGSERVER</p> <p><b>Meaning:</b> The system cannot access the area that contains the server name specified on the SERVER keyword.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• Ensure that the address of the server name is correct.</li> <li>• If your program is running in AR mode, ensure that: <ul style="list-style-type: none"> <li>– The ALET for the server name is correct.</li> <li>– You specified SYSSTATE ASCENV=AR before issuing the IXCSEND macro.</li> </ul> </li> </ul>

Table 19. Return and Reason Codes for the IXCSEND Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	00030008	<p><b>Equate Symbol:</b> IXSENDRSNBADALETSERVER</p> <p><b>Meaning:</b> Program error. The ALET that qualifies the address of the target Server Name (SERVER) is not zero, not a valid entry on the caller's Dispatchable Unit Access List (DU-AL), or not a valid entry for a common area data space.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• If your program is running in AR mode, ensure that you specified SYSSTATE ASCENV=AR before issuing the IXCSEND macro.</li> <li>• Ensure that the ALET for the server name storage is correct.</li> </ul>
8	0003000C	<p><b>Equate Symbol:</b> IXSENDRSNBADVALDESCRIPTION</p> <p><b>Meaning:</b> Program error. Server description (DESCRIPTION) value is not valid.</p> <p><b>Action:</b> DESCRIPTION can contain any alphanumeric (a-z, A-Z, 0-9), national (@,#,\$), or special (underscore or blank) character. Leading blanks and all blank descriptors are not permitted. Descriptions are case sensitive.</p>
8	00030018	<p><b>Equate Symbol:</b> IXSENDRSNBADPLISTTARGET</p> <p><b>Meaning:</b> Program error. Target specified in the parameter list for the IXCSEND request is not valid. The SENDTO target was not SERVER or ORIGINATOR.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• Ensure that a correct SENDTO keyword value was specified.</li> <li>• Ensure that the parameter list was not inadvertently overlaid and that it was correctly specified.</li> </ul>
8	000300EE	<p><b>Equate Symbol:</b> IXSENDRSNNORETMSGTOKEN</p> <p><b>Meaning:</b> Program error. RETMSGTOKEN must be specified when the results of the IXCSEND are being held (HOLDTIME) by XCF.</p> <p><b>Action:</b> When a non-zero HOLDTIME is specified, a RETMSGTOKEN is required in order to retrieve IXCSEND results using IXCRECV. Re-issue the IXCSEND request specifying the RETMSGTOKEN keyword.</p>

Table 19. Return and Reason Codes for the IXCSEND Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	00040004	<p><b>Equate Symbol:</b> IXSENDRSNBADSTGSERVERID</p> <p><b>Meaning:</b> The system cannot access the area that contains the server ID specified on the SERVERID keyword.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• Ensure that the address of the server ID is correct.</li> <li>• If your program is running in AR mode, ensure that: <ul style="list-style-type: none"> <li>– The ALET for the server ID is correct.</li> <li>– You specified SYSSTATE ASCENV=AR before issuing the IXCSEND macro.</li> </ul> </li> </ul>
8	00040008	<p><b>Equate Symbol:</b> IXSENDRSNBADALETSERVERID</p> <p><b>Meaning:</b> Program error. The ALET that qualifies the address of the server ID (SERVERID) is not zero, not a valid entry on the caller's Dispatchable Unit Access List (DU-AL), or not a valid entry for a common area data space.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• If your program is running in AR mode, ensure that you specified SYSSTATE ASCENV=AR before issuing the IXCSEND macro.</li> <li>• Ensure that the ALET for the server ID storage is correct.</li> </ul>
8	0004000C	<p><b>Equate Symbol:</b> IXSENDRSNBADVALSERVER</p> <p><b>Meaning:</b> Program error. Server Name (SERVER) value is not valid.</p> <p><b>Action:</b> Consider the following for server names:</p> <ul style="list-style-type: none"> <li>• Server names consist of four 8-byte sections. Each 8-byte section must be left justified, padded on the right with EBCDIC blanks as needed.</li> <li>• Each section string can contain any alphanumeric (a-z, A-Z, 0-9), national (@, #, \$), or underscore characters (_).</li> <li>• Any section but the first can be entirely blank.</li> <li>• Sender name is case sensitive.</li> </ul>
8	00040018	<p><b>Equate Symbol:</b> IXSENDRSNBADPLISTLEN</p> <p><b>Meaning:</b> Program error. Parameter list length specified in the IXCSEND parameter list is not valid.</p> <p><b>Action:</b> Ensure that the parameter list was not inadvertently overlaid and that it was correctly specified.</p>

Table 19. Return and Reason Codes for the IXCSEND Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	00050004	<p><b>Equate Symbol:</b> IXSENDRSNBADSTGMSGDATA</p> <p><b>Meaning:</b> Program error. XCF could not access the message buffer specified on the MSGDATA keyword.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• Ensure that the correct message buffer storage area was used.</li> <li>• If your program is running in AR mode, ensure that: <ul style="list-style-type: none"> <li>– The ALET for the message buffer is correct.</li> <li>– You specified SYSSTATE ASCENV=AR before issuing the IXCSEND macro.</li> </ul> </li> <li>• Ensure that the message buffer storage area was not inadvertently freed by your program.</li> </ul>
8	00050008	<p><b>Equate Symbol:</b> IXSENDRSNBADALETMSGDATA</p> <p><b>Meaning:</b> Program error. Parameter list length specified in the IXCSEND parameter list is not valid.</p> <p><b>Action:</b> Check for errors such as the following:</p> <ul style="list-style-type: none"> <li>• You did not specify SYSSTATE ASCENV=AR before issuing the IXCSEND macro.</li> <li>• You specified the wrong ALET for the message buffer.</li> </ul>
8	0005000C	<p><b>Equate Symbol:</b> IXSENDRSNBADVALMAXLEVEL</p> <p><b>Meaning:</b> Program error. The relationship between MINSERVERLEVEL and MAXSERVERLEVEL for the server selection criteria specified on the CRITERIA keyword is not valid.</p> <p><b>Action:</b> MAXSERVERLEVEL must be greater than or equal to MINSERVERLEVEL.</p>
8	00050018	<p><b>Equate Symbol:</b> IXSENDRSNBADPLISTRSD</p> <p><b>Meaning:</b> Program error. A reserved field in the control parameter list was not zero.</p> <p><b>Action:</b> Check to see if your program inadvertently overlaid the parameter list storage.</p>

Table 19. Return and Reason Codes for the IXCSEND Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	000500EE	<p><b>Equate Symbol:</b> IXCSENDRSNNOOUTSTANDINGRESP</p> <p><b>Meaning:</b> An IXCSEND SENDTO=ORIGINATOR request was issued to send a response for a server request, but a response was not outstanding for the server request.</p> <p><b>Action:</b> Issuing a response for a server request that has already been responded to or XCF considers as no longer needing a response is not allowed by XCF. A response might have already been made in the following ways:</p> <ul style="list-style-type: none"> <li>• The server exit or a designated unit of work associated with the server instance issued an IXCSEND SENDTO=ORIGINATOR to send a response to the request originator.</li> <li>• The server has asked XCF to provide an acknowledgment on its behalf through the use of the SXPL_RefusalCode or SXPL_ResultCode interfaces in the Server Exit Parameter List (SXPL). XCF sends a response to the request originator. No other response is allowed.</li> <li>• XCF responds to an outstanding server request on behalf of a server instance when the RESPBIND entity for the server instance ends.</li> </ul> <p>XCF considers a response to no longer be needed when a server request has not been responded to before the RESPTIME specified by the client when the server request was sent has exceeded.</p>
8	00060004	<p><b>Equate Symbol:</b> IXCSENDRSNBADSTGMSGDESC</p> <p><b>Meaning:</b> Program error. A message descriptor mapped by ixcysrvr_tDataDescriptor or message data pointed to by a message descriptor dd_DataAddr field passed on the MSGDESC keyword is not accessible due to a bad address.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• Ensure that the correct storage area was specified on the MSGDESC keyword and the correct #MSGDESC value was specified.</li> <li>• Ensure that the correct buffer address was specified in the dd_DataAddr field of the message descriptor.</li> <li>• If your program is running in AR mode, ensure that: <ul style="list-style-type: none"> <li>– The ALET for the message descriptors and message data (dd_DataAlet) is correct.</li> <li>– You specified SYSSTATE ASCENV=AR before issuing the IXCSEND macro.</li> </ul> </li> <li>• Ensure that the message descriptor or message buffer storage areas were not inadvertently freed by your program.</li> </ul>

Table 19. Return and Reason Codes for the IXCSEND Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	00060008	<p><b>Equate Symbol:</b> IXSENDRSNBADALETMSGDESC</p> <p><b>Meaning:</b> Program error. The ALET that qualifies the address of a message descriptor mapped by ixcsrvr_tDataDescriptor (MSGDESC) or an ALET in a message descriptor (dd_DataAlet) that qualifies the address of message data (dd_DataAddr) pointed to by a message descriptor is not zero, not a valid entry on the caller's Dispatchable Unit Access List (DU-AL), or not a valid entry for a common area data space.</p> <p><b>Action:</b> Check for errors such as the following:</p> <ul style="list-style-type: none"> <li>• You did not specify SYSSTATE ASCENV=AR before issuing the IXCSEND macro.</li> <li>• You specified the wrong ALET for the message descriptor table (MSGDESC) or for the message data pointed to by the message descriptor (dd_DataAlet).</li> </ul>
8	0006000C	<p><b>Equate Symbol:</b> IXSENDRSNBADVALFEATURES</p> <p><b>Meaning:</b> Program error. Input feature string for the server selection criteria specified on the CRITERIA keyword is not valid.</p> <p><b>Action:</b> The feature level (sf_Level) used a value (xFF) that is reserved by XCF. Use another value.</p>
8	000600EE	<p><b>Equate Symbol:</b> IXSENDRSNBADENVPAUSESRLB</p> <p><b>Meaning:</b> Environmental error. The IXCSEND request was issued from a SUSPEND exit routine or from an SRLB routine that the system abended with a 47B system completion.</p> <p>For a SENDTO=SERVER request, this reason code is presented in a target descriptor metadata record returned on an IXCRECV service call. The target descriptor metadata records describe the results of an IXCSEND request to a target server. See the td_SendRsnocode field of the target descriptor record mapping ixcsrvr_tTargetDescriptor.</p> <p>For a SENDTO=ORIGINATOR request, this reason code is returned on the IXCSEND service call.</p> <p><b>Action:</b> IXCSEND should not be issued from a SUSPEND exit environment or from an SRLB that the system abended with a 47B system completion code. Update the calling program to not invoke IXCSEND from a SUSPEND exit or after an ABEND 47B.</p>
8	00070004	<p><b>Equate Symbol:</b> XSENDRSNBADSTGRETMSGTOKEN</p> <p><b>Meaning:</b> Program error. RETMSGTOKEN storage is not accessible.</p> <p><b>Action:</b> None. The system could not store the message token identifying the IXCSEND request into the provided storage. Correct the storage addressing problem and re-issue the request if applicable.</p>



Table 19. Return and Reason Codes for the IXCSEND Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	00070008	<p><b>Equate Symbol:</b> IXSENDRSNBADALETRETMSGTOKEN</p> <p><b>Meaning:</b> Program error. The ALET that qualifies the address of the IXCSEND message token (RETMSGTOKEN) is neither zero nor a valid entry on the caller's Dispatchable Unit Access List (DU-AL), nor a valid entry for a common area data space.</p> <p><b>Action:</b> Check for errors such as the following:</p> <ul style="list-style-type: none"> <li>• You did not specify SYSSTATE ASCENV=AR before issuing the IXCSEND macro.</li> <li>• You specified the wrong ALET for the RETMSGTOKEN storage.</li> </ul> <p>Re-issue the request with the correct ALET for the RETMSGTOKEN storage.</p>
8	0007000C	<p><b>Equate Symbol:</b> IXSENDRSNBADVALTTOKEN</p> <p><b>Meaning:</b> Program error. The task token (TTOKEN) of the task that is responsible for invoking IXCRECV is not associated with a valid address space.</p> <p><b>Action:</b> Specify a valid task token and rerun the program.</p>
8	000700EE	<p><b>Equate Symbol:</b> IXSENDRSNSYSTEMNOTACTIVE</p> <p><b>Meaning:</b> Environmental error. The target system for a SENDTO=SERVER request was not active in the sysplex or the system that a SENDTO=ORIGINATOR response request was destined for was not active in the sysplex.</p> <p>For a SENDTO=ORIGINATOR request, this reason code is returned on the IXCSEND service call.</p> <p>For a SENDTO=SERVER request, this reason code is presented in a target descriptor metadata record returned on an IXCRECV service call. The target descriptor metadata records describe the results of an IXCSEND request to a target server. See the td_SendRsnocode field of the target descriptor record mapping ixcysrvr_tTargetDescriptor.</p> <p><b>Action:</b> None.</p>
8	00080004	<p><b>Equate Symbol:</b> IXSENDRSNBADSTGDESCRIPTION</p> <p><b>Meaning:</b> Program error. The storage area specified on the DESCRIPTION keyword is not accessible. The storage is not addressable.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• Ensure that the address of the request DESCRIPTION is correct.</li> <li>• If your program is running in AR mode, ensure that: <ul style="list-style-type: none"> <li>– The ALET for the message descriptors and message data (dd_DataAlet) is correct.</li> <li>– You specified SYSSTATE ASCENV=AR before issuing the IXCSEND macro.</li> </ul> </li> </ul>

Table 19. Return and Reason Codes for the IXCSEND Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	00080008	<p><b>Equate Symbol:</b> IXSENDRSNBADALETDESCRIPTION</p> <p><b>Meaning:</b> Program error. The ALET that qualifies the address of the Description (DESCRIPTION) of the request is not zero, not a valid entry on the caller's Dispatchable Unit Access List (DU-AL), or not a valid entry for a common area data space.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• If your program is running in AR mode, ensure that you specified SYSSTATE ASCENV=AR before issuing the IXCSEND macro.</li> <li>• Ensure that the ALET for the message description information is correct.</li> </ul>
8	0008000C	<p><b>Equate Symbol:</b> IXSENDRSNBADVALSYSNAME</p> <p><b>Meaning:</b> Program error. A system name value provided through the SYSNAME keyword contains invalid characters.</p> <p><b>Action:</b> Valid SYSNAME characters are A-Z, 0-9, \$, @, # and the wild card characters specified or defaulted to. Correct the system name pattern value and re-issue the request.</p>
8	00080018	<p><b>Equate Symbol:</b> IXSENDRSNBADPLISTSYSTEMS</p> <p><b>Meaning:</b> Program error. Invalid SYSTEMS parameter specified.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• Ensure that a correct SYSTEMS keyword value was specified.</li> <li>• Ensure that the parameter list was not inadvertently overlaid and that it was correctly specified.</li> </ul>
8	000800EE	<p><b>Equate Symbol:</b> IXSENDRSNNOTTARGETSYSTEMS</p> <p><b>Meaning:</b> Program error. Invalid target selection criteria specified. There were no active systems in the sysplex that matched the criteria specified by the SYSTEMS or SERVERID keywords.</p> <p><b>Action:</b> Correct the target selection criteria and re-issue the request.</p>
8	00090004	<p><b>Equate Symbol:</b> IXSENDRSNBADSTGMSGCNTL</p> <p><b>Meaning:</b> Program error. The system cannot access the area that contains the user message control information (MSGCNTL).</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• Ensure that the address of the message control information is correct.</li> <li>• If your program is running in AR mode, ensure that <ul style="list-style-type: none"> <li>– You specified SYSSTATE ASCENV=AR before issuing the IXCSEND macro.</li> <li>– The ALET for the message control information is correct.</li> </ul> </li> </ul>

Table 19. Return and Reason Codes for the IXCSEND Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	00090008	<p><b>Equate Symbol:</b> IXCSENDRSNBADALETMSGCNTL</p> <p><b>Meaning:</b> Program error. The ALET that qualifies the address of the user message control data (MSGCNTL) is neither zero nor a valid entry on the caller's Dispatchable Unit Access List (DU-AL), nor a valid entry for a common area data space.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• If your program is running in AR mode, ensure that you specified SYSSTATE ASCENV=AR before issuing the IXCSEND macro.</li> <li>• Ensure that the ALET for the user message control information is correct.</li> </ul>
8	0009000C	<p><b>Equate Symbol:</b> IXCSENDRSNBADVALEXPECTREPLY</p> <p><b>Meaning:</b> ExpectReply=YES is not supported by the IXCSEND service for SENDTO=ORIGINATOR response requests.</p> <p><b>Action:</b> Do not code the EXPECTREPLY keyword or specify EXPECTREPLY=NO when sending a response message (SENDTO=ORIGINATOR).</p>
8	00090018	<p><b>Equate Symbol:</b> IXCSENDRSNBADPLISTRECVBIND</p> <p><b>Meaning:</b> Program error. Invalid RECVBIND specified. SRB mode callers must explicitly specify a TTOKEN on the request when establishing a recovery bind to a task for IXCRECV.</p> <p><b>Action:</b> Specify RECVBIND=TASK, TTOKEN=xttoken on the IXCSEND invocation.</p>
8	000A0004	<p><b>Equate Symbol:</b> IXCSENDRSNBADSTGRESPTOKEN</p> <p><b>Meaning:</b> Program error. The system cannot access the area that contains the response token (RESPTOKEN).</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• Ensure that the address of the response token is correct.</li> <li>• If your program is running in AR mode, ensure that: <ul style="list-style-type: none"> <li>– The ALET for the response token is correct.</li> <li>– You specified SYSSTATE ASCENV=AR before issuing the IXCSEND macro.</li> </ul> </li> </ul>

Table 19. Return and Reason Codes for the IXCSEND Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	000A0008	<p><b>Equate Symbol:</b> IXSENDRSNBADALETRESPTOKEN</p> <p><b>Meaning:</b> Program error. The ALET that qualifies the address of the response token (RESPTOKEN) is not zero, not a valid entry on the caller's Dispatchable Unit Access List (DU-AL), or not a valid entry for a common area data space.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• If your program is running in AR mode, ensure that you specified SYSSTATE ASCENV=AR before issuing the IXCSEND macro.</li> <li>• Ensure that the ALET for the response token is correct.</li> </ul>
8	000A000C	<p><b>Equate Symbol:</b> IXSENDRSNBADVALSERVERID</p> <p><b>Meaning:</b> Program error. Server ID value (SERVERID) is not valid.</p> <p><b>Action:</b> Ensure that a correct server ID value was specified.</p>
8	000B0004	<p><b>Equate Symbol:</b> IXSENDRSNBADSTGSYSNAMES</p> <p><b>Meaning:</b> Program error. The system cannot access the storage identified by the SYSNAME keyword containing a system name pattern or system name pattern array.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• Ensure that the address of the system name patterns is correct.</li> <li>• Ensure that the values for the #SYSTEMS and LENSYSENTRY are correct.</li> <li>• If your program is running in AR mode, ensure that: <ul style="list-style-type: none"> <li>– The ALET for the system name patterns is correct.</li> <li>– You specified SYSSTATE ASCENV=AR before issuing the IXCSEND macro.</li> </ul> </li> </ul>
8	000B0008	<p><b>Equate Symbol:</b> IXSENDRSNBADALETSYSNAMES</p> <p><b>Meaning:</b> Program error. The ALET that qualifies the address of system name patterns (SYSNAME) is neither zero nor a valid entry on the caller's Dispatchable Unit Access List (DU-AL), nor a valid entry for a common area data space.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• If your program is running in AR mode, ensure that you specified SYSSTATE ASCENV=AR before issuing the IXCSEND macro.</li> <li>• Ensure that the ALET for the system name patterns is correct.</li> </ul>
8	000B000C	<p><b>Equate Symbol:</b> IXSENDRSNBADVALSENDTIME</p> <p><b>Meaning:</b> The SENDTIME (SENDTIME) is not within the valid range as defined for the IXCSEND service</p> <p><b>Action:</b> Specify a valid value for SENDTIME.</p>

Table 19. Return and Reason Codes for the IXCSEND Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	000B0018	<p><b>Equate Symbol:</b> IXCSENDRSNBADPLISTTARGSERVER</p> <p><b>Meaning:</b> Program error. Target server specified in the parameter list for the IXCSEND request is not valid.</p> <p><b>Action:</b> A target server must be identified by server name (SERVER keyword) or server ID (SERVERID keyword). Ensure that the parameter list was not inadvertently overlaid and that it was correctly specified.</p>
8	000C0004	<p><b>Equate Symbol:</b> IXCSENDRSNBADSTGSYSIDS</p> <p><b>Meaning:</b> Program error. The system cannot access the storage identified by the SYSID keyword containing a system ID or system ID array.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• Ensure that the address of the system ID or system ID array is correct.</li> <li>• Ensure that the values for the #SYSTEMS and LENSYSENTRY are correct.</li> <li>• If your program is running in AR mode, ensure that <ul style="list-style-type: none"> <li>– The ALET for the system name patterns is correct.</li> <li>– you specified SYSSTATE ASCENV=AR before issuing the IXCSEND macro.</li> </ul> </li> </ul>
8	000C0008	<p><b>Equate Symbol:</b> IXCSENDRSNBADALETSYSIDS</p> <p><b>Meaning:</b> Program error. The ALET that qualifies the address of an XCF system ID of a target system or array of system IDs (SYSID) is neither zero nor a valid entry on the caller's Dispatchable Unit Access List (DU-AL), nor a valid entry for a common area data space.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• If your program is running in AR mode, ensure that you specified SYSSTATE ASCENV=AR before issuing the IXCSEND macro.</li> <li>• Ensure that the ALET for the system ID or system ID array is correct.</li> </ul>
8	000C000C	<p><b>Equate Symbol:</b> IXCSENDRSNBADVALRESPTIME</p> <p><b>Meaning:</b> Environmental error. The RespTime (RESPTIME) is not within the valid range for the IXCSEND service.</p> <p><b>Action:</b> Specify a valid response time value.</p>
8	000D000C	<p><b>Equate Symbol:</b> IXCSENDRSNBADVALHOLDTIME</p> <p><b>Meaning:</b> The HOLDTIME value is not within the valid range for the IXCSEND service</p> <p><b>Action:</b> Specify a valid value for HOLDTIME.</p>

Table 19. Return and Reason Codes for the IXCSEND Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	000D0018	<p><b>Equate Symbol:</b> IXSENDRSNBADPLISTCRITERIA</p> <p><b>Meaning:</b> Program error. CRITERIA mapping content is not valid. Reserved fields are not initialized to nulls.</p> <p><b>Action:</b> Set the reserved fields of the CRITERIA mapping to nulls and re-issue the request.</p>
8	000E000C	<p><b>Equate Symbol:</b> IXSENDRSNTTOKENTASKTERM</p> <p><b>Meaning:</b> Environmental error. The task identified by TTOKEN and responsible for invoking IXCRECV represents a task that is terminating. A receive bind cannot be made to a task that is ending.</p> <p><b>Action:</b> Specify a TTOKEN for a task that is not terminating and retry the request.</p>
8	000F0004	<p><b>Equate Symbol:</b> IXSENDRSNBADSTGCRITERIA</p> <p><b>Meaning:</b> Program error. The system cannot access the storage identified by the CRITERIA keyword containing the server selection criteria.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• Ensure that the address for the server selection criteria is correct.</li> <li>• If your program is running in AR mode, ensure that: <ul style="list-style-type: none"> <li>– The ALET for the storage containing the server selection criteria is correct</li> <li>– You specified SYSSTATE ASCENV=AR before issuing the IXCSEND macro.</li> </ul> </li> </ul>
8	000F0008	<p><b>Equate Symbol:</b> IXSENDRSNBADALETCRITERIA</p> <p><b>Meaning:</b> Program error. The ALET that qualifies the address of target server selection criteria specified by the CRITERIA keyword is neither zero nor a valid entry on the caller's Dispatchable Unit Access List (DU-AL), nor a valid entry for a common area data space.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• If your program is running in AR mode, ensure that you specified SYSSTATE ASCENV=AR before issuing the IXCSEND macro. -</li> <li>• Ensure that the ALET for the target server selection criteria is correct.</li> </ul>
8	000F000C	<p><b>Equate Symbol:</b> IXSENDRSNBADTTOKENMASTERAS</p> <p><b>Meaning:</b> Program error. A task residing in the master address space cannot be designated with a receive bind if the master address space is not the caller's HOME address space.</p> <p><b>Action:</b> Correct the error, and retry.</p>

Table 19. Return and Reason Codes for the IXCSEND Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	000F0018	<b>Equate Symbol:</b> IXSENDRSNBADCRITERIAVERSION <b>Meaning:</b> Program error. Invalid version number specified in the CRITERIA data mapping <b>Action:</b> Specify a valid version. See the ixcsrvr_tCriteria mapping in the IXCYSRVR macro.
8	0010000C	<b>Equate Symbol:</b> IXSENDRSNBADVALRESPTOKEN <b>Meaning:</b> Program error. The RESPTOKEN that identifies an originating message to which this response is being sent is invalid or corrupted. <b>Action:</b> Correct the error, and retry.
8	0011000C	<b>Equate Symbol:</b> IXSENDRSNBADVALRECVBIND <b>Meaning:</b> The RECVBIND value specified in the service parameter list is not valid and not recognized by the IXCSEND service. <b>Action:</b> Specify a valid RECVBIND value of TASK, ADDRSPACE, or SYSTEM. Ensure that the parameter list was not inadvertently overlaid and that it was correctly specified.
8	0012000C	<b>Equate Symbol:</b> IXSENDRSNBADVAL#SYSTEMS <b>Meaning:</b> Environmental error. Bad #SYSTEMS parameter specified. <b>Action:</b> #SYSTEMS must be greater than zero (0).
8	0013000C	<b>Equate Symbol:</b> IXSENDRSNBADVALWILDCARDONE <b>Meaning:</b> Program error. Bad WILDCARDONE parameter specified. The WILDCARDONE character contains a value that is not accepted by IXCSEND. <b>Action:</b> Specify a valid character for WILDCARDONE. WILDCARDONE characters cannot be: <ul style="list-style-type: none"> <li>• A-Z, a-z, 0-9</li> <li>• @, #, \$</li> <li>• &amp;</li> <li>• EBCDIC blank or the value specified for WILDCARDANY</li> </ul>
8	0014000C	<b>Equate Symbol:</b> IXSENDRSNBADVALLENMDENTRY <b>Meaning:</b> Environmental error. LENMDENTRY value is not valid. <b>Action:</b> LENMDENTRY must be greater than or equal to the length of one data descriptor (16).

Table 19. Return and Reason Codes for the IXCSEND Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	0015000C	<p><b>Equate Symbol:</b> IXSENDRSNBADVALMSGSTGKEY</p> <p><b>Meaning:</b> Environmental error. Message data described by MSGDATA or message data described by message descriptors specified on the MSGDESC keyword cannot be fetched using the storage key indicated by MSGSTGKEY.</p> <p><b>Action:</b> Specify a valid storage key.</p>
8	0016000C	<p><b>Equate Symbol:</b> IXSENDRSNBADVALWILDCARDANY</p> <p><b>Meaning:</b> Environmental error. Bad WILDCARDANY parameter specified. The WILDCARDANY character contains a value that is not accepted by IXCSEND.</p> <p><b>Action:</b> Specify a valid character for WILDCARDANY. WILDCARDANY characters cannot be:</p> <ul style="list-style-type: none"> <li>• A-Z, a-z, 0-9</li> <li>• @, #, \$</li> <li>• &amp;</li> <li>• EBCDIC blank or the value specified for WILDCARDANY</li> </ul>
8	0017000C	<p><b>Equate Symbol:</b> IXSENDRSNBADSERVERREQMSGLEN</p> <p><b>Meaning:</b> The message length of an XCF Server request message exceeds the maximum supported length for the current version of the IXCSEND macro service.</p> <p><b>Action:</b> Specify a value for the message length that is supported by the IXCSEND service for XCF Server requests.</p>
8	0018000C	<p><b>Equate Symbol:</b> IXSENDRSNBADVALWILDCARDSSAME</p> <p><b>Meaning:</b> The value for WILDCARDONE and WILDCARDANY are the same. The values must be different.</p> <p><b>Action:</b> The wildcard characters for WILDCARDANY and WILDCARDONE cannot be the same value. Change the values and retry the request.</p>
8	0019000C	<p><b>Equate Symbol:</b> IXSENDRSNBADVALSTOKEN</p> <p><b>Meaning:</b> The TOKEN for the address space responsible for invoking IXCRECV does not represent a valid address space.</p> <p><b>Action:</b> Ensure that the provided TOKEN value represents a valid address space.</p>
8	001A000C	<p><b>Equate Symbol:</b> IXSENDRSNBADVALLENSYSENTRY</p> <p><b>Meaning:</b> Environmental error. LENSYSENTRY must be at least eight (8) when SYSTEMS=NAME is specified or LENSYSENTRY must be at least four (4) when SYSTEMS=SYSID is specified.</p> <p><b>Action:</b> Specify a valid LENSYSENTRY value.</p>



Table 19. Return and Reason Codes for the IXCSEND Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	000100CE	<p><b>Equate Symbol:</b> IXSENDRSNALESERVADDFAILED</p> <p><b>Meaning:</b> Environment Error. Unexpected return code from ALESERV while trying to add the XCF address space to the DU-AL</p> <p><b>Action:</b> Retry the request after allowing some time for the condition to clear.</p>
C	000200CE	<p><b>Equate Symbol:</b> IXSENDRSNSYSTEMRESOURCES</p> <p><b>Meaning:</b> Environmental error. Environmental Error. XCF Signalling or other system resources were not readily available to satisfy the IXCSEND request.</p> <p><b>Action:</b> Retry the request after allowing some time for the system or sysplex to clear the resource constraint condition.</p>
C	000300CE	<p><b>Equate Symbol:</b> IXSENDRSNDOWNLEVELSYSTEM</p> <p><b>Meaning:</b> Environmental error. The selected target system does not support the XCF client/server protocols.</p> <p>For a SENDTO=SERVER request, this reason code is presented in a target descriptor metadata record returned on an IXCRECV service call. The target descriptor metadata records describe the results of an IXCSEND request to a target server. See the td_SendRsnocode field of the target descriptor record mapping ixcysrvr_tTargetDescriptor.</p> <p><b>Action:</b> Determine why the SENDTIME expired and increase the SENDTIME if necessary.</p>
C	000400CE	<p><b>Equate Symbol:</b> IXSENDRSNENVSENDTIMEEXP</p> <p><b>Meaning:</b> Environmental error. The amount of time that IXCSEND was allowed to pause the unit of work to synchronously complete accessing the caller's storage areas expired. XCF canceled the incomplete IXCSEND processing and the message was not sent to all the selected target systems.</p> <p>This reason code is presented in a target descriptor metadata record returned on an IXCRECV service call. The target descriptor metadata records describe the results of an IXCSEND request to a target server. See the td_SendRsnocode field of the target descriptor record mapping ixcysrvr_tTargetDescriptor.</p> <p><b>Action:</b> Determine why the SENDTIME expired and increase the SENDTIME if necessary.</p>

Table 19. Return and Reason Codes for the IXCSEND Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	000500CE	<p><b>Equate Symbol:</b> IXCSNDRSNFORCECOMPLETION</p> <p><b>Meaning:</b> An IXCMMSGC REQUEST=COMPLETION request was issued for a message while XCF was processing an IXCSEND request. The IXCSEND request was forced to complete before XCF sent messages to all intended targets.</p> <p>This reason code is presented in a target descriptor metadata record returned on an IXCRECV service call. The target descriptor metadata records describe the results of an IXCSEND request to a target server. See the td_SendRsncode field of the target descriptor record mapping ixcysrvr_tTargetDescriptor.</p> <p><b>Action:</b> Determine if the application intended to force the completion of the IXCSEND request prior to the message being sent to one or more of the intended targets. Re-issue the request if applicable.</p>
C	000600CE	<p><b>Equate Symbol:</b> IXCSNDRSNRELEASEMSG</p> <p><b>Meaning:</b> Environmental error. An IXCMMSGC REQUEST=RELEASEMSG request was issued for a message while XCF was processing an IXCSEND request. The IXCSEND request was forced to end before XCF sent messages to all intended targets.</p> <p>For a SENDTO=SERVER request, this reason code is presented in a target descriptor metadata record returned on an IXCRECV service call. The target descriptor metadata records describe the results of an IXCSEND request to a target server. See the td_SendRsncode field of the target descriptor record mapping ixcysrvr_tTargetDescriptor.</p> <p>For a SENDTO=ORIGINATOR request, this reason code is returned on the IXCSEND service call.</p> <p><b>Action:</b> None.</p>
C	000700CE	<p><b>Equate Symbol:</b> IXCSNDRSNDISCARDMSG</p> <p><b>Meaning:</b> Environmental error. An IXCMMSGC REQUEST=DISCARDMSG request was issued for a message while XCF was processing an IXCSEND request. The IXCSEND request was forced to end before XCF sent messages to all intended targets.</p> <p><b>Action:</b> None.</p>

Table 19. Return and Reason Codes for the IXCSEND Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	000800CE	<p><b>Equate Symbol:</b> IXCSENDRSNASYNABENDSENDING</p> <p><b>Meaning:</b> Environmental error. The send request was cancelled to the specific target due to an asynchronous abend that affected the suspended unit of work. XCF could not complete sending the message to the target .</p> <p>For a SENDTO=SERVER request, this reason code is presented in a target descriptor metadata record returned on an IXCRECV service call. The target descriptor metadata records describe the results of an IXCSEND request to a target server. See the td_SendRsnocode field of the target descriptor record mapping ixcysrvr_tTargetDescriptor.</p> <p>For a SENDTO=ORIGINATOR request, this reason code is returned on the IXCSEND service call.</p> <p><b>Action:</b> None.</p>
C	000900CE	<p><b>Equate Symbol:</b> IXCSENDRSNENVRESPTIMEEXP</p> <p><b>Meaning:</b> Environmental error. The amount of time (RESPTIME) that IXCSEND was allowed to let send requests to targets remain send pending expired. XCF canceled the incomplete IXCSEND processing and the message was not sent to all the selected target systems.</p> <p>This reason code is presented in a target descriptor metadata record returned on an IXCRECV service call. The target descriptor metadata records describe the results of an IXCSEND request to a target server. See the td_SendRsnocode field of the target descriptor record mapping ixcysrvr_tTargetDescriptor.</p> <p><b>Action:</b> Retry the request after allowing some time for the system or sysplex to clear the conditions causing the send to remain pending.</p>
10	None	<p><b>Equate Symbol:</b> None.</p> <p><b>Meaning:</b> Failure in XCF processing.</p> <p><b>Action:</b> None.</p>

## Examples

None.



## Chapter 23. IXCSETUS – Update the User State Field

### Description

The IXCSETUS macro updates the user state field that the cross-system coupling facility (XCF) maintains for an XCF member. The target member must be in the same XCF group as the caller and must be in one of the following states: created, active, quiesced, or failed; the caller can be the target member. The caller specifies, on the NEWUS parameter, what the new value of the target member's user state field is to be. When a change is made to the user state field, XCF broadcasts the change to those active members in the group that have group user routines.

Before it requests a change in the user state field, a caller can ask XCF to confirm the current value of the field. XCF compares the current value with a *compare value* that COMPUS identifies. This operation is similar to a *compare and swap* instruction.

- If the current value and the compare value match, XCF replaces the current value with the value that NEWUS points to.
- If the current value and the compare value do not match, XCF leaves the current value of the user state field as it is. If you specify OLDUS, XCF copies the current value to that field. The USLEN parameter determines the length of the data XCF copies to OLDUS.

If the OLDUS and COMPUS parameters both point to the same field, XCF replaces the compare value with the current value.

### Environment

The following are the environment requirements for the caller.

Environment	Environment requirement
Minimum authorization:	Supervisor state or PKM allowing key 0 - 7
Dispatchable unit mode:	Task
Cross memory mode:	Any PASN, any HASN, any SASN. The primary address space must be the same as the primary address space of the caller of the IXCJOIN that placed the calling member in the active state.
AMODE:	31-bit
ASC mode:	Primary or access register (AR)
Interrupt status:	Enabled for I/O and external interrupts
Locks:	No locks held
Control parameters:	Must be in the primary address space or be in an address/data space that is addressable through a public entry in the dispatchable unit access list (DU-AL)

### Programming Requirements

If the program is in AR mode, issue SYSSTATE ASCENV=AR before IXCSETUS. SYSSTATE ASCENV=AR tells the system to generate code appropriate for AR mode.

## Restrictions

---

- The caller must be a member in the active state, and the target member must be created, active, quiesced, or failed; both must be in the same XCF group.
- The caller can have no enabled, unlocked task (EUT) FRRs established.

## Input Register Information

---

Before issuing the IXCSETUS macro, the caller does not have to place any information into any register unless using it in register notation for a particular parameter, or using it as a base register.

## Output Register Information

---

When control returns to the caller, the general purpose registers (GPRs) contain:

### Register Contents

**0**

If GPR 15 contains a zero or X'10', GPR 0 is used as a work register by the system; otherwise, GPR 0 contains a reason code.

**1**

Used as a work register by the system.

**2-13**

Unchanged.

**14**

Used as a work register by the system.

**15**

Return code.

When control returns to the caller, the access registers (ARs) contain:

### Register Contents

**0-1**

Used as work registers by the system

**2-13**

Unchanged

**14-15**

Used as work registers by the system

Some callers depend on register contents remaining the same before and after issuing a service. If the system changes the contents of registers on which the caller depends, the caller must save them before issuing the service, and restore them after the system returns control.

## Performance Implications

---

None.

## Understanding IXCSETUS Version Support

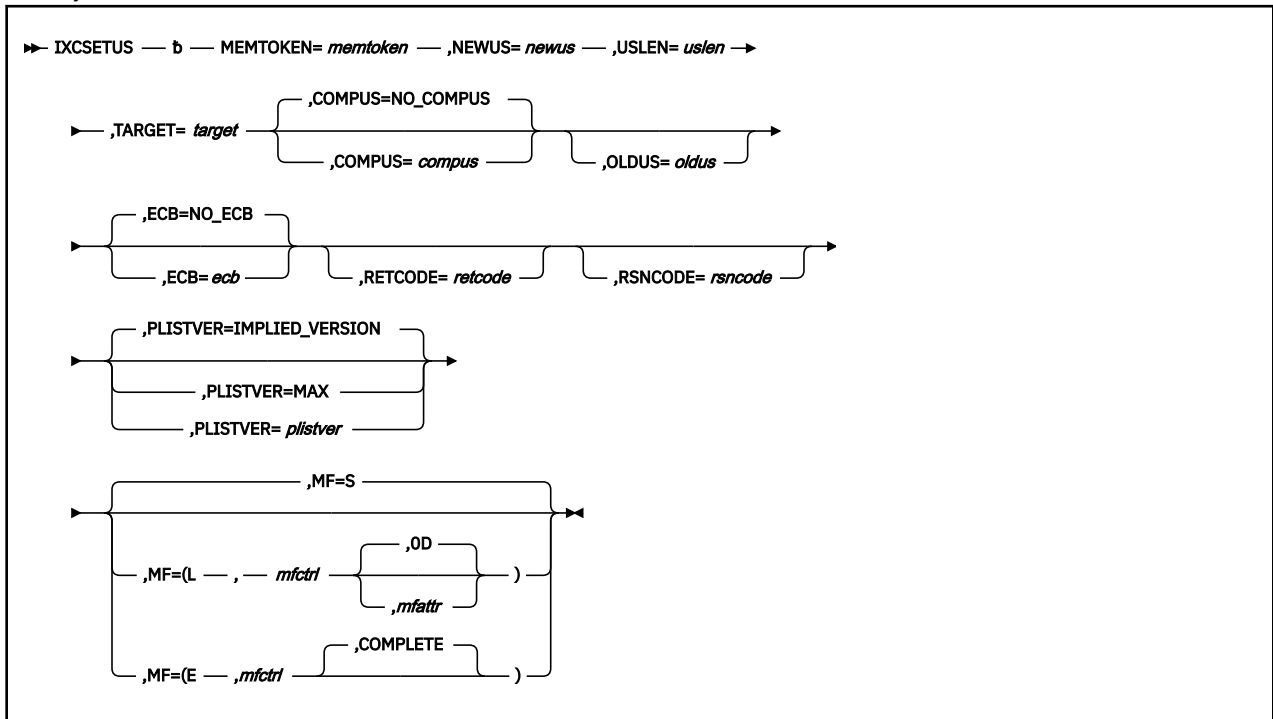
---

The IXCSETUS macro supports version 0 keywords and functions.

Specify the version of the parameter list that you want generated with the PLISTVER keyword. See Chapter 2, “[Specifying a Macro Version Number](#),” on page 5 for considerations when specifying the version of the parameter list with PLISTVER.

## Syntax Diagram

The syntax of the IXCSETUS macro is as follows:



## Parameter Descriptions

The parameters are explained as follows. Default values are underlined:

### **,COMPUS=NO\_COMPUS**

### **,COMPUS=compus**

Use this input parameter to specify the user state compare value. If the compare value matches the current value, XCF replaces the current value with the new value. If the values do not match, XCF does not change the user state field.

If you do not want to compare the user state with a value, but would like to code this parameter, specify COMPUS=NO\_COMPUS.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the area (with a length of USLEN) to be used for comparison against the current user state field for the target.

### **,ECB=NO\_ECB**

### **,ECB=ecb**

Use this input parameter to specify a 32-bit ECB that XCF is to post when IXCSETUS completes its actions. If you do not specify ECB, or if you specify ECB=NO\_ECB, IXCSETUS completes synchronously. If you do specify ECB, XCF might return control before IXCSETUS completes its actions. If the return code is 0, indicating that XCF accepted the request, then issue a WAIT on the ECB. If the return code is not 0, XCF will not post the ECB.

If you specify OLDUS with ECB, then XCF updates the OLDUS field just before it posts the ECB. In this case, the OLDUS field must be in commonly addressable storage.

When XCF posts the ECB, the ECB contains the following:

- Bits 8 - 23 contain the low-order halfword of the reason code that XCF would have returned in GPR 0 if the caller had not specified ECB.
- Bits 24 - 31 contain the low-order byte of the return code that XCF would have returned in GPR 15 if the caller had not specified ECB.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 4-byte field that contains the address of the ECB.

#### **MEMTOKEN=memtoken**

Use this input parameter to specify the 64-bit token of the calling member. IXCJOIN provided this token when it placed the member in the active state.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-byte field that contains the member token.

**,MF=S**  
**,MF=(L,mfctrl)**  
**,MF=(L,mfctrl,mfattr)**  
**,MF=(L,mfctrl,0D)**  
**,MF=(M,mfctrl)**  
**,MF=(M,mfctrl,COMPLETE)**  
**,MF=(M,mfctrl,NOCHECK)**  
**,MF=(E,mfctrl)**  
**,MF=(E,mfctrl,COMPLETE)**  
**,MF=(E,mfctrl,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

#### **,mfctrl**

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

#### **,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

#### **,COMPLETE**

#### **,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

#### **COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if SMILE=*var* were an optional parameter and the default is SMILE=NO\_SMILE then it would not be documented. However, if the default was SMILE=-), then it would be documented because a value would be the default.



**NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

**,NEWUS=newus**

Use this input parameter to specify the value that XCF is to place in the user state field. This value is called the “new value.” If you specify COMPUS and the current value and the compare value do not match, the new value will not replace the current value of the field.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the area (with a length of USLEN) that contains the new user state information.

**,OLDUS=oldus**

Use this output parameter to specify the location where XCF is to copy the current value of the user state field. If OLDUS and COMPUS both point to the same location, XCF updates the existing compare value with the current value of the user state field. You might specify this parameter to find out why the comparison failed.

If you specify ECB with OLDUS, then XCF updates the OLDUS field just before it posts the ECB. In this case, the OLDUS field must be in commonly addressable storage, and any ALET that qualifies its address must be zero.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the area (with a length of USLEN) where the current value of the user state field of the target member will be placed.

**,PLISTVER=IMPLIED\_VERSION****,PLISTVER=MAX****,PLISTVER=plistver**

Use this input parameter to specify the version of the macro. See [“Understanding IXCSETUS Version Support”](#) on page 374 for a description of the options available with PLISTVER.

**,RETCODE=retcode**

Use this output parameter to specify the location in which the system is to copy the return code from GPR 15.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 4-byte field that will contain the return code when this request completes.

**,RSNCODE=rsncode**

Use this output parameter to specify the location in which the system is to copy the reason code from GPR 0.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 4-byte field that will contain the reason code (if any) when the request completes.

**,TARGET=target**

Use this input parameter to specify the 64-bit token of the member whose user state field is to be changed or used for comparison. You can get the value of this token by using the IXCQUERY macro.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-byte field that contains the member token of the member whose user state you are processing.

**,USLEN=uslen**

Use this input parameter to specify how many bytes of the user state field XCF is to change. XCF uses the length you specify on USLEN as the length of the COMPUS, OLDUS, and NEWUS fields:

- When XCF compares the current user state value with the value in COMPUS, XCF compares only the number of bytes you specify on USLEN.
- When XCF copies the current value of the user state field to OLDUS, XCF copies only the number of bytes you specify on USLEN.
- When XCF updates the current user state value with the value in NEWUS, XCF updates only the number of bytes you specify on USLEN.

The length must be from 1 to 32 bytes.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 4-byte field that contains the number of bytes of the user state field to process.

## ABEND Codes

None.

## Return and Reason Codes

When the IXCSETUS macro returns control to your program:

- GPR 15 (and *retcode*, if you coded RETCODE) contains a return code.
- GPR 0 (and *rsncode*, if you coded RSNCODE) contains a reason code, if applicable.

Macro IXCYCON provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

**0**

IXCRETCODEOK

**4**

IXCRETCODEWARNING

**8**

IXCRETCODEPARMERROR

**C**

IXCRETCODEENVERROR

**10**

IXCRETDECOMPERROR

The following table identifies the hexadecimal return and reason codes and the equate symbol associated with each reason code.

Table 20. Return and Reason Codes for the IXCSETUS Macro		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
00	None.	<p><b>Meaning:</b> IXCSETUS completed successfully as the caller requested. If the caller specified ECB, the processing is asynchronous; if the caller omitted ECB, the processing is synchronous.</p> <p><b>Action:</b> None.</p>
04	04	<p><b>Equate Symbol:</b> IXCSETUSRSNNOCHANGEOLDEQNEW</p> <p><b>Meaning:</b> Program error. XCF did not change the user state value and did not notify members. If the caller specified OLDUS, XCF returned the current user state value. COMPUS was not specified, and the new value is the same as the current value.</p> <p><b>Action:</b> None required, because the current user state already matches the desired user state.</p>

Table 20. Return and Reason Codes for the IXCSETUS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
04	08	<p><b>Equate Symbol:</b> IXCSETUSRSNNOCHANGEOLDNECOMPUS</p> <p><b>Meaning:</b> This return code is informational. COMPUS was specified, and the compare value is not equal to the current value. XCF did not change the user state field and did not notify members. If the caller specified OLDUS, XCF returned the current user state value.</p> <p><b>Action:</b> None required. The user state should be updated only when the current value matches the COMPUS value. However, if COMPUS is being used to serialize user state updates with other user state update requests, you can retry the request with a new user state value based on the OLDUS, and use the OLDUS as the COMPUS value.</p>
08	04	<p><b>Equate Symbol:</b> IXCSETUSRSNNOTACTIVE</p> <p><b>Meaning:</b> Program error. The caller's member token does not identify an active member.</p> <p><b>Action:</b> Ensure that the correct MEMTOKEN was specified. Further action depends on your application. If the member must be placed in a not-defined state, you can use the IXCQUERY service to determine what state the member is currently in. If the member is currently in a failed, quiesced, or created state, you can use the IXCDELET service to place the member in a not-defined state.</p>
08	08	<p><b>Equate Symbol:</b> IXCSETUSRSNINAPPROPRIATEPRIMARY</p> <p><b>Meaning:</b> Program error. The primary address space is not the same as the primary address space of the caller of the IXCJOIN that placed the calling member in the active state.</p> <p><b>Action:</b> Ensure that the correct MEMTOKEN was specified. Change your program to issue IXCSETUS from the same primary address space as the primary address space of the caller of the IXCJOIN that placed the calling member in the active state.</p>
08	0C	<p><b>Equate Symbol:</b> IXCSETUSRSNTARGETDIFFERENTGROUP</p> <p><b>Meaning:</b> Program error. The calling member and the target member are not members of the same XCF group.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• Ensure that the correct TARGET and MEMTOKEN member tokens were specified.</li> <li>• Correct your program so that the member updating the user state updates only the user state of members within the same group.</li> </ul>
08	10	<p><b>Equate Symbol:</b> IXCSETUSRSNTARGETNOTVALID</p> <p><b>Meaning:</b> Program error. The target member's token is not valid.</p> <p><b>Action:</b> Correct the target member token, and retry the request.</p>

Table 20. Return and Reason Codes for the IXCSETUS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
08	14	<p><b>Equate Symbol:</b> IXCSETUSRSNOLDUSALETNOTPRIMARY</p> <p><b>Meaning:</b> Program error. The caller specified ECB and OLDUS, and is running in AR mode. However, the ALET that qualifies the address of the OLDUS field is not primary. If you specify ECB, the request completes asynchronously; therefore, the ALET for the OLDUS must be primary, and the OLDUS address must be in common storage. See the requirements for the OLDUS keyword.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• You specified SYSSTATE ASCENV=AR before issuing the IXCSETUS macro.</li> <li>• The ALET of the OLDUS field is zero (primary address space ALET).</li> </ul>
08	18	<p><b>Equate Symbol:</b> IXCSETUSRSNOLDUSBADSTGNOTCOMMON</p> <p><b>Meaning:</b> Program error. The caller specified ECB and OLDUS; however, the OLDUS field is not in common storage.</p> <p><b>Action:</b> Ensure that the correct OLDUS field address was used, and that the field is in common storage.</p>
08	28	<p><b>Equate Symbol:</b> IXCSETUSRSNOLDUSBADALET</p> <p><b>Meaning:</b> Program error. The caller specified OLDUS, but provided an inappropriate ALET. The ALET is neither zero nor associated with a valid public entry on the caller's DU-AL.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• You specified SYSSTATE ASCENV=AR before issuing the IXCSETUS macro.</li> <li>• The ALET is a public entry on the DU-AL or zero (primary address space ALET).</li> <li>• Your program is not intended to run in primary ASC mode.</li> </ul>

Table 20. Return and Reason Codes for the IXCSETUS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
08	3C	<p><b>Equate Symbol:</b> IXCSETUSRSNOLDUSINCOMPLETE</p> <p><b>Meaning:</b> Program error. The caller specified OLDUS incorrectly. The user state might or might not have been updated. Check the two high-order bytes of this reason code fullword xxyy003C for the return code xx (either 00 or 04) and reason code yy the caller would have received if the caller had coded those parameters correctly.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• Ensure that the correct OLDUS address was used.</li> <li>• If your program is running in AR mode: <ul style="list-style-type: none"> <li>– Ensure that you specified SYSSTATE ASCENV=AR before issuing the IXCSETUS macro.</li> <li>– Ensure that the OLDUS ALET is correct.</li> </ul> </li> <li>• Ensure that the OLDUS storage area was not inadvertently freed by your program.</li> <li>• You might want to abnormally end your program or take some other action that will record the problem with your OLDUS storage area.</li> </ul>
08	40	<p><b>Equate Symbol:</b> IXCSETUSRSNPLISTRSDNOTVALID</p> <p><b>Meaning:</b> Program error or environmental error. A reserved field in the control parameter list is not zero.</p> <p><b>Action:</b> Check to see if your program inadvertently overlaid the parameter list storage, and that it was assembled with the correct macro library for the release of MVS your program is running on.</p>
08	100	<p><b>Equate Symbol:</b> IXCSETUSRSNPLISTBADALET</p> <p><b>Meaning:</b> Program error. Your program is running in primary ASC mode, and the ALET that qualifies the address of the control parameter list is neither zero nor associated with a valid public entry on the caller's DU-AL.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• You specified SYSSTATE ASCENV=AR before issuing the IXCSETUS macro.</li> <li>• The ALET for the parameter list is on the DU-AL or is zero (primary address space ALET).</li> </ul>
08	104	<p><b>Equate Symbol:</b> IXCSETUSRSNPLISTVERSIONNOTVALID</p> <p><b>Meaning:</b> Program error. A version number in the control parameter list is not valid.</p> <p><b>Action:</b> Check to see if your program inadvertently overlaid the parameter list storage, and that it was assembled with the correct macro library for the release of MVS your program is running on.</p>

Table 20. Return and Reason Codes for the IXCSETUS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
08	108	<p><b>Equate Symbol:</b> IXCSETUSRSNPLISTBADFUNCTION</p> <p><b>Meaning:</b> Program error. The function code in the control parameter list is not valid.</p> <p><b>Action:</b> Check to see if your program inadvertently overlaid the parameter list storage.</p>
08	10C	<p><b>Equate Symbol:</b> IXCSETUSRSNPLISTBADSTG</p> <p><b>Meaning:</b> Program error. XCF could not access the control parameter list.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• Ensure that the correct parameter list storage area was specified.</li> <li>• If your program is running in AR mode: <ul style="list-style-type: none"> <li>– Ensure that you specified SYSSTATE ASCENV=AR before issuing the IXCSETUS macro.</li> <li>– Ensure that the parameter list ALET is correct.</li> </ul> </li> </ul>
08	110	<p><b>Equate Symbol:</b> IXCSETUSRSNNEWUSNOTACCESSIBLE</p> <p><b>Meaning:</b> Program error. XCF could not access the NEWUS value.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• Ensure that the correct NEWUS address was used.</li> <li>• If your program is running in AR mode: <ul style="list-style-type: none"> <li>– Ensure that you specified SYSSTATE ASCENV=AR before issuing the IXCSETUS macro.</li> <li>– Ensure that the NEWUS ALET is correct.</li> </ul> </li> </ul>
08	114	<p><b>Equate Symbol:</b> IXCSETUSRSNUSLENBADVALUE</p> <p><b>Meaning:</b> Program error. The length value in USLEN is less than 1 or greater than 32.</p> <p><b>Action:</b> Correct the USLEN, and retry the request.</p>
08	118	<p><b>Equate Symbol:</b> IXCSETUSRSNNOTTASKMODE</p> <p><b>Meaning:</b> Program error. The caller is not in task mode.</p> <p><b>Action:</b> Correct your program so that it issues IXCSETUS only while in task mode.</p>
08	11C	<p><b>Equate Symbol:</b> IXCSETUSRSNNOTENABLED</p> <p><b>Meaning:</b> Program error. The caller is not enabled.</p> <p><b>Action:</b> Correct your program so that it issues IXCSETUS only while enabled.</p>

Table 20. Return and Reason Codes for the IXCSETUS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
08	124	<b>Equate Symbol:</b> IXCSETUSRSNCOMPUSNOTACCESSIBLE <b>Meaning:</b> Program error. XCF could not access the COMPUS value. <b>Action:</b> Take one or more of the following actions: <ul style="list-style-type: none"> <li>• Ensure that the correct COMPUS address was used.</li> <li>• If your program is running in AR mode: <ul style="list-style-type: none"> <li>– Ensure that you specified SYSSTATE ASCENV=AR before issuing the IXCSETUS macro.</li> <li>– Ensure that the COMPUS ALET is correct.</li> </ul> </li> </ul>
0C	18	<b>Equate Symbol:</b> IXCSETUSRSNTASKABENDED <b>Meaning:</b> Environmental error. While the issuing task was suspended for XCF processing, the task was abended; that is, another unit of work attempted to abnormally terminate this task. The state of the IXCSETUS request is unpredictable. <b>Action:</b> Determine why this task was abended.
10	None.	<b>Meaning:</b> System error. XCF processing failed. <b>Action:</b> Retry the request at least once. If the problem persists, record the return code, and supply it to the appropriate IBM support personnel.

## Example

*Operation:* If the user state current value for the target member is X'22', replace that value with the value X'44', and save the current value. In the IXCSETUS macro,

- COMPUS points to the value 22.
- NEWUS points to the value 44.
- OLDUS points to the area where XCF will save the current value.

For OLDUS, the storage must be in common storage. In this example, the STORAGE OBTAIN macro returns the address of the area in register 1. The code moves the address to register 3. XCF is to store the return code and reason code into the fields RETURN and REASON. The code is as follows:

```

* ZERO ECB1
  STORAGE OBTAIN,LENGTH=LEN,SP=228  OBTAIN STORAGE FOR OLD USER    X
                                     STATE VALUE
  LR   R3,R1                        SAVE ADDRESS FOR IXCSETUS      X
                                     INVOCATION

  IXCSETUS MEMTOKEN=TOKEN1,NEWUS=STATE4,USLEN=LEN,                   X
          TARGET=TOKEN2,COMPUS=STATE2,OLDUS=(R3),ECB=ECB1,          X
          RETCODE=RETURN,RSNCODE=REASON,MF=S

* TEST RETURN CODE
  WAIT ECB=ECB1                                WAIT FOR IXCSETUS TO COMPLETE

TOKEN1 DS CL8      MEMBER TOKEN OF CALLER
TOKEN2 DS CL8      TOKEN OF MEMBER WHOSE USER    X
                      STATE IS TO BE CHANGED
RETURN DS 1F       RETURN CODE
REASON DS 1F       REASON CODE

```

**IXCSETUS Macro**

ECB1	DS	1F	ECB TO BE POSTED BY XCF
STATE2	DC	X'22'	USER STATE FOR COMPARISON
STATE4	DC	X'44'	NEW USER STATE VALUE
LEN	DC	F'1'	LENGTH OF ALL USER STATES

You can obtain the member tokens from the QUAMTKN field in the area returned by IXCCREAT, IXCJOIN, or IXCQUERY, and mapped by the IXCYQUAA mapping macro.



## Chapter 24. IXCSRVR — Define a Server to XCF

### Description

The XCF server interface (IXCSRVR) is one of the macros that comprise the XCF Client/Server interfaces. With these interfaces, a "client" can send a request to a "server" for processing and then receive the "results" provided by the server. Related macros include the XCF send service (IXCSEND), which is used to send requests to servers and responses to clients, and the XCF receive service (IXCRECV), which is used to obtain results provided by the servers. See [Chapter 22, "IXCSEND — Send Client/Server Requests and Responses,"](#) on page 337 and [Chapter 20, "IXCRECV— Receive Client/Server Information,"](#) on page 313.

With IXCSRVR you can start and stop server instances. A task becomes a server by invoking IXCSRVR with REQTYPE=START. If the start request is successful, XCF does not return to the caller until the server is stopped. When starting a server, you must provide the address of a server exit routine. Running under the caller's task, XCF repeatedly calls the indicated server exit routine to process client requests. If there are no requests to process, XCF suspends the task until there is a request for it to process.

You need to provide a server exit routine that the XCF Server can use to build the server exit parameter list (SXPL) for the request and a work area for additional data. You can perform the following actions:

- Start or stop a server or server instance
- Define feature strings and client/server compatibility data for mixed release levels of clients and servers
- Define response binds to allow for some other work unit to handle sending responses if needed.

For guidance information and other details on using IXCSRVR with the client/server interfaces, see the IXCSRVR topic in the chapter "Using XCF for Client/Server Communication" of [z/OS MVS Programming: Sysplex Services Guide](#).

### Environment

The following are the environment requirements for the caller.

Environment	Environment Requirement
Minimum authorization:	Supervisor state or PKM allowing keys 0-7.
Dispatchable unit mode:	For REQTYPE=START: Task mode For REQTYPE=STOP: Task or SRB mode.
Cross memory mode:	For REQTYPE=START, any primary address space. Primary, secondary, and home must be the same space.  For REQTYPE=STOP, any primary address space; any secondary address space; any home address space.
AMODE:	31- or 64-bit If in 64-bit mode, specify SYSSTATE AMODE64=YES before invoking this macro.

Environment	Environment Requirement
ASC mode:	<p>Primary or access register (AR)</p> <p>If in Access Register ASC mode, specify SYSSTATE ASCENV=AR before invoking this macro.</p> <p>For REQTYPE=START, any primary address space. Primary, secondary, and home must be the same space.</p> <p>For REQTYPE=STOP, any primary address space; any secondary address space; any home address space.</p>
Interrupt status:	Enabled for I/O and external interrupts
Locks:	No locks held
Control parameters:	<p>The storage containing the IXCSRVR parameter list, as well as the storage areas referenced by certain keywords (SERVER, SERVERID, INFO, DESCRIPTION), must reside in the primary address space of the caller, or in a space addressable through a public entry on the dispatchable unit access list (DU-AL), or in a common area data space. XCF uses the key of the caller when accessing these storage areas.</p> <p>If the parameter list, or the storage areas referenced by certain keywords (SERVER, SERVERID, INFO, DESCRIPTION) do not reside in the primary address space of the caller, the caller must be running in Access Register ASC mode.</p>

## Programming Requirements

---

None.

## Restrictions

---

REQTYPE=START has the following restrictions:

- No EUT FRR set.
- IXCSRVR cannot be used by tasks higher in the task tree than the cross memory resource owning task (the top, or first, job step task in the address space).
- Cannot be invoked by an address space resource manager.
- Cannot be invoked by a task that is being terminated.

## Input Register Information

---

Input registers:

**0-1**

reserved

**2-12**

undefined

**13**

When starting a server, value to be set in GR13 when XCF calls the server exit routine. Otherwise undefined.

**14-15**

reserved

**AR0-AR1**

reserved

**AR2-AR12**

undefined

**AR13**

When starting a server in AR mode, value to be set in AR13 when XCF calls the server exit routine.  
 When starting a server in primary mode, the AR13 is zero when XCF calls the server exit routine.  
 Otherwise undefined.

**AR14-AR15**

reserved

## Output Register Information

---

When control returns to the caller, the GPRs contain:

**0**

Reason code based on GR15 or 0

**1**

Unpredictable

**2-13**

Unchanged

**14**

Unpredictable

**15**

Return code

When control returns to the caller, the access registers (ARs) contain:

**0-1**

Unpredictable

**2-13**

Unchanged

**14-15**

Unpredictable

## Performance Implications

---

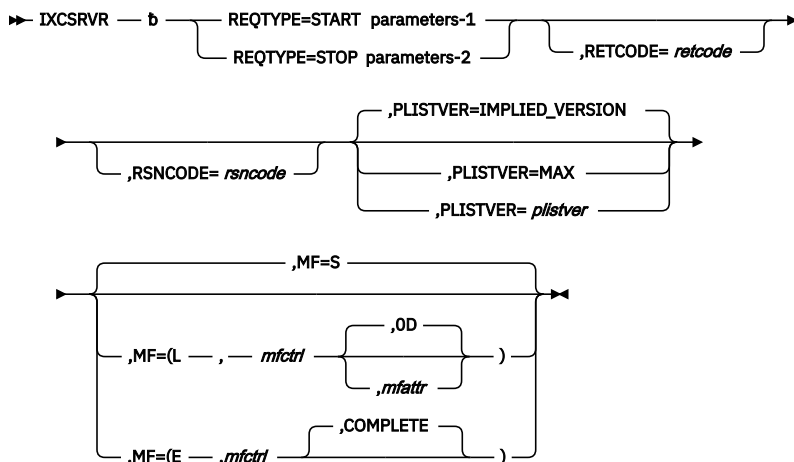
None.

## Syntax

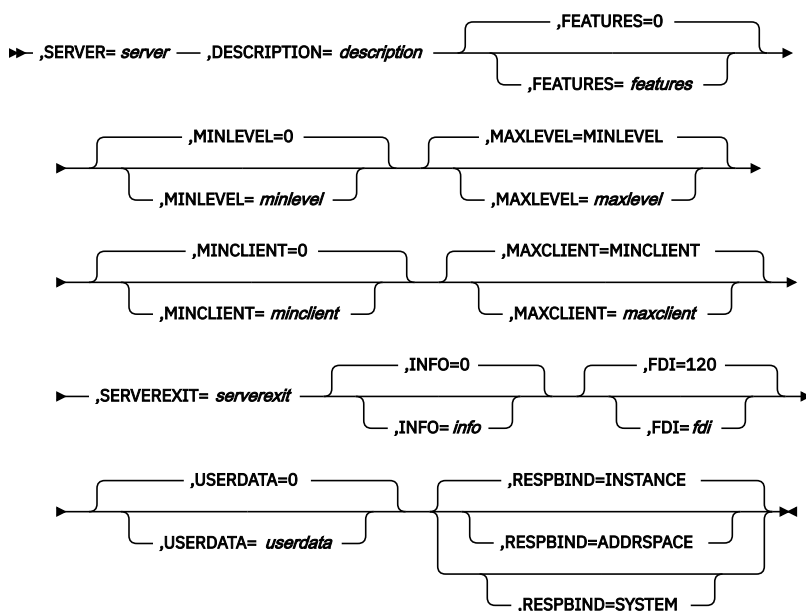
---

The IXCSRVR macro is written as follows:

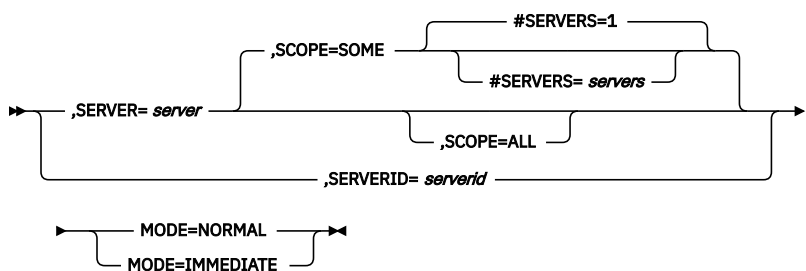
## main diagram



## parameters-1



## parameters-2



## Parameters

The parameters are explained as follows:

**,#SERVERS=#servers**

**,#SERVERS=1**

When SCOPE=SOME, SERVER=server and REQTYPE=STOP are specified, use this optional input parameter to contain the number of server instances on the local system that are to be stopped. If #SERVERS exceeds the number of instances, all instances of the indicated server on the local system are to be stopped. The default is 1.

**To code:** Specify the RS-type name, or address in register (2)-(12), of a fullword field, or specify a literal decimal value.

**,DESCRIPTION=description**

When REQTYPE=START is specified, use this required input parameter to contain a description of the server. The string can contain any alphanumeric (A-Z,a-z,0-9), national (@,#,\$), or special (underscore or blank) character. Leading blanks and all blank descriptors are not permitted. Descriptions are case sensitive. The description will appear in various XCF messages and diagnostic data reports. The description is intended to help installations and service personnel understand the function, purpose, or role of the server.

**To code:** Specify the RS-type name, or address in register (2)-(12), of a 32-character field that contains the description.

**,FDI=fdi**

**,FDI=120**

When REQTYPE=START is specified, use this optional input parameter to indicate the server failure detection interval, which is the number of seconds that the server can appear to be unresponsive before the system considers it to have failed. If the server is deemed to have failed, the server task is subject to being terminated by XCF (abend 00C reason x164).

A server is deemed responsive if it returns to the XCF server exit stub routine, and unresponsive otherwise. If the server is processing a request for which there is an associated timeout value, the XCF monitor expects the server exit to return to XCF within FDI seconds after the timeout expires. If there is no associated timeout, the server exit is expected to return to XCF within FDI seconds.

For example, suppose the server exit was called to process a client request that had a timeout value of 100 seconds (IXCSEND MSG=REQUEST RESPTIME=100). XCF would allow the server to process the request for as long as 100 seconds. After 100 seconds have elapsed, XCF allows up to FDI more seconds for the server exit routine to return to XCF. If it fails to do so, XCF deems the server to be unresponsive.

If not specified, the server failure detection interval is set to two minutes (120). The FDI value must be between 1 and 3600, inclusive.

The default is 120.

**To code:** Specify the RS-type name, or address in register (2)-(12), of a halfword field, or specify a literal decimal value for the interval.

**,FEATURES=features**

**,FEATURES=0**

When REQTYPE=START is specified, use this optional input parameter to contain a "feature string". The feature string indicates the features supported by the server and is mapped by ixcsrvr\_tFeatures in macro IXCSRVR. The interpretation of the feature level and feature flags within the feature string is determined by the server.

When using IXCSEND SENDTO=SERVER to send a request to a server, clients can specify the features that a server must support in order to process a request. As explained in the IXCSRVR macro, XCF compares the features requested by the client to the features supported by the server to determine whether the server is capable of processing a client request. If no server supports the required features, the request is cancelled and acknowledged with a response code indicating "no receiver". The default is 0.

**To code:** Specify the RS-type name, or address in register (2)-(12), of an 8-character field that contains the feature string.

**,INFO=info**

**,INFO=0**

When REQTYPE=START is specified, use this optional input parameter to contain information about the server. The content and interpretation of this information is determined by the server. XCF saves a copy of the indicated data. The saved copy is passed to the server exit routine whenever it is called. A copy of the data is also visible to other processes in the sysplex through queries that return information about the server. Thus related componentry, such as a sender, can obtain whatever information the server cares to define and share about itself. The default is 0.

**To code:** Specify the RS-type name, or address in register (2)-(12), of a 64-character field that contains the information.

**,MAXCLIENT=maxclient**

**,MAXCLIENT=MINCLIENT**

When REQTYPE=START is specified, use this optional input parameter to contain a number that identifies the largest client level that the server will accept. MAXCLIENT must be greater than or equal to MINCLIENT. The default is MINCLIENT.

**To code:** Specify the RS-type name, or address in register (2)-(12), of a fullword field, or specify a literal decimal value for the client level.

**,MAXLEVEL=maxlevel**

**,MAXLEVEL=MINLEVEL**

When REQTYPE=START is specified, use this optional input parameter to contain a number that identifies the maximum server level supported by this server. MAXLEVEL must be greater than or equal to MINLEVEL.

The default is MINLEVEL.

**To code:** Specify the RS-type name, or address in register (2)-(12), of a fullword field, or specify a literal decimal value for the server level.

**,MF=S**

**,MF=(L,list addr)**

**,MF=(L,list addr,attr)**

**,MF=(L,list addr,0D)**

**,MF=(E,list addr)**

**,MF=(E,list addr,COMPLETE)**

Use this optional input parameter to specify the macro form.

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service. MF=S is the default.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter might be coded with the list form of the macro.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form of the macro stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

**,list addr**

The name of a storage area to contain the parameters. For MF=S and MF=E, this can be an RS-type address or an address in register (1)-(12).

**,attr**

An optional 1- to 60-character input string that you use to force boundary alignment of the parameter list. Use a value of 0F to force the parameter list to a word boundary, or 0D to force the parameter list to a doubleword boundary. If you do not code *attr*, the system provides a value of 0D.

**,COMPLETE**

Specifies that the system is to check for required parameters and supply defaults for omitted optional parameters.

**,MINCLIENT=*minclient*****,MINCLIENT=0**

When REQTYPE=START is specified, use this optional input parameter to contain a number that identifies the smallest client level that is acceptable to the server. In combination with MAXCLIENT, MINCLIENT defines the range of client levels whose requests the server will accept. The interpretation of level is determined by the client.

When a client sends a request, it specifies its level. When the request arrives on the server system, XCF compares the client level to the range of client levels acceptable to the server. If the client level does not fall within the acceptable range, the server is not eligible to process the request.

The default is 0.

**To code:** Specify the RS-type name, or address in register (2)-(12), of a fullword field, or specify a literal decimal value for the client level.

**,MINLEVEL=*minlevel*****,MINLEVEL=0**

When REQTYPE=START is specified, use this optional input parameter to contain a number that identifies the smallest server level supported by this server. In combination with MAXLEVEL, MINLEVEL defines the range of levels supported by the server. The interpretation of level is determined by the server.

When a client sends a request, it specifies the range of server levels that are acceptable for processing the request. When the request arrives on the server system identified by server name, XCF compares the range of server levels requested by the client to the range of levels supported by the server. If the requested range does not intersect the supported range, the server is not eligible to process the request.

The default is 0.

**To code:** Specify the RS-type name, or address in register (2)-(12), of a fullword field, or specify a literal decimal value of the server level.

**,MODE=NORMAL****,MODE=IMMEDIATE**

When REQTYPE=STOP is specified, use this required parameter to indicate whether the servers being stopped are to finish pending work

**,MODE=NORMAL**

The server is to be stopped as soon as it finishes processing both its current request as well as any pending work that might have been queued prior to the stop request being recognized.

Requests that arrive after the stop request is recognized will be cancelled if there is no other suitable server instance to process the request.

**,MODE=IMMEDIATE**

The server is to be stopped as soon as it finishes processing its current request (if any). The server will not process any pending work that might have been queued. If stopping the server instance leaves the pending work with no suitable server to process it, the pending work is cancelled.

For example, pending client requests would be discarded and acknowledged with a "no receiver" response code.

**,PLISTVER=IMPLIED\_VERSION****,PLISTVER=MAX****,PLISTVER=0**

Use this optional input parameter to specify the version of the macro. PLISTVER determines which parameter list the system generates. PLISTVER is an optional input parameter on all forms of the macro, including the list form. When using PLISTVER, specify it on all macro forms used for a request and with the same value on all of the macro forms. The values are:

- **IMPLIED\_VERSION**, which is the lowest version that allows all parameters specified on the request to be processed. If you omit the PLISTVER parameter, IMPLIED\_VERSION is the default.
- **MAX**, if you want the parameter list to be the largest size currently possible. This size might grow from release to release and affect the amount of storage that your program needs.

If you can tolerate the size change, IBM recommends that you always specify PLISTVER=MAX on the list form of the macro. Specifying MAX ensures that the list-form parameter list is always long enough to hold all the parameters you might specify on the execute form, when both are assembled with the same level of the system. In this way, MAX ensures that the parameter list does not overwrite nearby storage.

- **0**, if you use the currently available parameters.

**To code:** Specify one of the following:

- IMPLIED\_VERSION
- MAX
- A decimal value of 0

#### **,REQTYPE=START**

#### **,REQTYPE=STOP**

Use this required parameter to indicate whether a server is to be started or stopped.

#### **REQTYPE=START**

Start an instance of the named server.

If the request is valid, the XCF service routine will not return to the caller until the server is stopped. The XCF service routine will repeatedly call the indicated server exit routine (SERVEREXIT) to process client messages (requests). If there are no messages to process, the work unit is suspended until one arrives.

#### **REQTYPE=STOP**

Stop one or more servers on the local system.

XCF will initiate stop processing for the indicated instances of the server on the local system, and then return to the caller. The actual stopping of the servers completes asynchronously to the stop request. Thus on return from the IXCSRVR invocation, the indicated server instances might or might not have been stopped.

#### **,RESPBIND=INSTANCE**

#### **,RESPBIND=ADDRSPACE**

#### **,RESPBIND=SYSTEM**

When REQTYPE=START is specified, use this optional parameter to indicate the default recovery bind XCF should establish for the server with respect to response processing. This specification determines the circumstances for which XCF will cancel responses that this server instance is expected to send. Canceling a response implies that its recipient does not need to wait for its timeout value to expire to discover that no response was sent.

The server exit routine can update the SXPL\_RespBind field to change the default response bind on a request by request basis. The default is RESPBIND=INSTANCE.

#### **,RESPBIND=INSTANCE**

XCF is to associate responsibility for sending responses to this instance of the server. If the server is stopped, XCF is to cancel any outstanding responses that the server was expected to provide.

This option is intended for servers whose server exit is responsible for sending responses. In such cases, termination of the server instance implies no further responses will be sent. Termination of the server instance includes task termination, address space termination, system termination, or simply stopping the server.

This option can be used even if there are other work units responsible for sending the responses. However, there will in effect be a race condition between such responses and the XCF cancel processing. The responses might or might not be presented to the intended recipient. The IXCSEND service might or might not accept the response for delivery.



**,RESPBIND=ADDRSPACE**

XCF is to associate responsibility for sending responses to the server address space. When the address space terminates, XCF cancels any outstanding responses that the server instance was expected to provide.

This option is intended for servers whose server exit routine does not send responses. Instead, the server exit routine arranges for other work units in the server address space to send responses. Since the server exit is not responsible for sending responses, XCF should not cancel the expected responses if the server instance terminates. Not until the server address space terminates should XCF cancel the responses.

This option can be used even if there are work units in other address spaces responsible for sending the responses. However, there will in effect be a race condition between such responses and the XCF cancel processing. The responses might or might not be presented to the intended recipient. The IXCSEND service might or might not accept the response for delivery.

**,RESPBIND=SYSTEM**

XCF is to associate responsibility for sending responses to the local system. When the system terminates, XCF cancels any outstanding responses that the server instance was expected to provide.

This option is intended for servers whose server exit routine does not send responses. Instead, the server exit routine arranges for work units in an address space other than the server address space to send responses. Since neither the server exit nor the server address space is responsible for sending responses, XCF should not cancel the expected responses when the server instance or the server address space terminates. Not until the system terminates should XCF cancel the responses.

If the server arranges for some other system in the sysplex to send responses, there will be a race condition between such responses and the XCF cancel processing. The responses sent by the third party systems might or might not be presented to the intended recipient.

**,RETCODE=retcode**

Use this optional output parameter into which the return code is to be copied from GPR 15. If you specify 15, GPR15, REG15, or R15 (within or without parentheses), the value will be left in GPR 15.

**To code:** Specify the RS-type name of a fullword field, or register (2)-(12) or (15), (GPR15), (REG15), or (R15).

**,RSNCODE=rsncode**

Use this optional output parameter into which the reason code is to be copied from GPR 0. If you specify 0, 00, GPR0, GPR00, REG0, REG00, or R0 (within or without parentheses), the value will be left in GPR 0.

**To code:** Specify the RS-type name of a fullword field, or register (0) or (2)-(12), (00), (GPR0), (GPR00), REG0, (REG00), or (R0).

**,SCOPE=SOME****,SCOPE=ALL**

When SERVER=server and REQTYPE=STOP are specified, use this required parameter to indicate how many instances of the indicated server are to be stopped.

**,SCOPE=SOME**

The indicated number of server instances on the local system are to be stopped.

**,SCOPE=ALL**

All instances of the indicated server on the local system are to be stopped.

**,SERVER=server**

When REQTYPE=START or REQTYPE=STOP is specified, use this required input parameter to contain the name of the server.

Server names are mapped by `ixcysvr_tName` (macro `IXCYSRVR`). Server names consist of four 8 byte sections. Each 8 byte section must be left justified, padded on the right with EBCDIC blanks as needed. Each section can contain any alphanumeric (A-Z,a-z,0-9), national (@,#,\$), or underscore

character. Any section but the first can be entirely blank. Server names are case sensitive. The name will appear in various XCF messages and diagnostic data reports. Clients using the IXCSEND service to send requests to a server can identify the target server by its name.

To avoid names used by IBM, do not begin server names (section 1) with the letters A through I or the character string SYS.

**To code:** Specify the RS-type name, or address in register (2)-(12), of a 32-character field that contains the server name.

**,SERVEREXIT=*serverexit***

When REQTYPE=START is specified, use this required input parameter to specify the routine that XCF is to call to process the indicated request type. The exit receives control under the current thread in the same addressing mode and ASC mode as the caller. On entry, R1 contains the address of the server exit parameter list (SXPL) that is mapped by `ixcysrvr_tSXPL` in `IXCYSRVR`. The exit is responsible for establishing its own recovery environment.

**To code:** Specify the RS-type name, or address in register (2)-(12), of a field that contains the routine.

**,SERVERID=*serverid***

When REQTYPE=STOP is specified, use this required input parameter to specify the name of the server id of the server instance that is to be stopped. If a specified server ID does not identify an active server instance on the local system, the request is rejected.

The storage containing the indicated SERVERID can reside in the primary address space of the caller, or in a space addressable through a public entry on the dispatchable unit access list (DU-AL) or in a common area data space.

**To code:** Specify the RS-type name, or address in register (2)-(12), of a 16-character field with the server id.

**,USERDATA=*userdata***

**,USERDATA=0**

When REQTYPE=START is specified, use this optional input parameter to contain user data that is to be associated with the server. A copy of the user data is presented to the server exit routine. The default is 0.

**To code:** Specify the RS-type name, or address in register (2)-(12), of a 16-character field that contains the user data.

## ABEND Codes

---

None.

## Return and Reason Codes

---

When the IXCSRVR macro returns control to your program:

- GPR 15 (and *retcode*, when you code RETCODE) contains a return code.
- When the value in GPR 15 is not zero, GPR 0 (and *rsncode*, when you code RSNCODE) contains a reason code.

The following table identifies the hexadecimal return and reason codes and the equate symbol associated with each reason code.

Table 21. Return and Reason Codes for the IXCSRVR Macro

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0	None	Successful completion.  For a START request, the server exit routine set the "stop code" to indicate that it had finished processing (ixcysrvr_kStopCodeFinished). For a STOP request, stop processing for the indicated server was initiated.
4	01	<b>Equate Symbol:</b> IXCSRVRRSNSTOPPED  <b>Meaning:</b> The XCF Server Stub loop terminated because the server was stopped through an IXCSRVR REQTYPE=STOP request. The server exit routine was not involved in the stop and so did not have an opportunity to set the stop code.  This reason applies only to REQTYPE=START. <b>Action:</b> None.
4	02	<b>Equate Symbol:</b> IXCSRVRRSNEXITFAILURE  <b>Meaning:</b> Environmental error The server exit routine set the "stop code" to indicate that it was terminating due to a failure (ixcysrvr_kStopCodeFailure).  This reason applies only to REQTYPE=START. <b>Action:</b> None.
4	04	<b>Equate Symbol:</b> IXCSRVRRSNNOSERVER  <b>Meaning:</b> Environmental error. The server does not have any instances defined on the local system. Nothing to stop. <b>Action:</b> None.
8	01	<b>Equate Symbol:</b> IXCSRVRRSNPLISTBADSTG  <b>Meaning:</b> Environmental error. Program error. The storage containing the parameter list is not accessible. <b>Action:</b> Correct the storage error, and retry.
8	02	<b>Equate Symbol:</b> IXCSRVRRSNPLISTBADALET  <b>Meaning:</b> Environmental error. Program error. The parameter list ALET is not valid.  <b>Action:</b> The storage containing the IXCSRVR parameter list must reside in the primary address space of the caller, or in a space addressable through a public entry on the dispatchable unit access list (DU-AL), or in a common area data.
8	03	<b>Equate Symbol:</b> IXCSRVRRSNPLISTBADRSVD  <b>Meaning:</b> Environmental error. Reserved field in the parameter list is not zero.  <b>Action:</b> Either the parameter list was corrupted, or the program is running on a system that does not have the necessary support.

Table 21. Return and Reason Codes for the IXCSRVR Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	04	<b>Equate Symbol:</b> IXCSRVRRSNPLISTBADVERSION <b>Meaning:</b> Environmental error. Program error. Parameter list version is not valid. <b>Action:</b> Either the parameter list was corrupted or the program is running on a system that does not have the necessary support.
8	05	<b>Equate Symbol:</b> IXCSRVRRSNPLISTBADREQTYPE <b>Meaning:</b> Environmental error. Program error. <b>Action:</b> Parameter list request type is not valid. Either the parameter list was corrupted or the program is running on a system that does not have the necessary support.
8	06	<b>Equate Symbol:</b> IXCSRVRRSNEXITFAILED <b>Meaning:</b> Environmental error. Program error. The server exit routine failed. XCF stops the offending server instance. <b>Action:</b> Note that this particular reason code will not normally be seen. If the server exit routine fails, XCF recovery processing will stop the server instance, cleanup XCF resources, and then percolate to the recovery routine established by the work unit that started the server.
8	07	<b>Equate Symbol:</b> IXCSRVRRSNSERVERBADSTG <b>Meaning:</b> Environmental error. The storage containing the server name is not accessible. <b>Action:</b> None.
8	08	<b>Equate Symbol:</b> IXCSRVRRSNSERVERBADALET <b>Meaning:</b> Invalid parameters. The ALET for the server name is not valid. The storage containing the server name must reside in the primary address space of the caller, or in a space addressable through a public entry on the dispatchable unit access list (DU-AL), or in a common area data space. <b>Action:</b> Ensure that the ALET for the server name is valid.
8	09	<b>Equate Symbol:</b> IXCSRVRRSNSERVERBADNAME <b>Meaning:</b> Environmental error. Program error. The server name is not valid. <b>Action:</b> Specify a valid server name.
8	0A	<b>Equate Symbol:</b> IXCSRVRRSNDESCBADSTG <b>Meaning:</b> Environmental error. The storage containing the server description is not accessible. <b>Action:</b> None.

Table 21. Return and Reason Codes for the IXCSRVR Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	0B	<b>Equate Symbol:</b> IXCSRVRRSNDESCBADALET <b>Meaning:</b> Environmental error. The ALET for the server description is not valid. The storage containing the server description must reside in the primary address space of the caller, or in a space addressable through a public entry on the dispatchable unit access list (DU-AL), or in a common area data space. <b>Action:</b> Ensure that the ALET for the server description is valid.
8	0C	<b>Equate Symbol:</b> IXCSRVRRSNDESCBADDESC <b>Meaning:</b> Invalid parameters. The server description is not valid. <b>Action:</b> Specify a valid description for the server.
8	0D	<b>Equate Symbol:</b> IXCSRVRRSNINFOBADSTG <b>Meaning:</b> Environmental error. The storage containing the server info is not accessible. <b>Action:</b> None.
8	0E	<b>Equate Symbol:</b> IXCSRVRRSNINFOBADALET <b>Meaning:</b> Environmental error. Program error. The ALET for the server info is not valid. The storage containing the server description must reside in the primary address space of the caller, or in a space addressable through a public entry on the dispatchable unit access list (DU-AL), or in a common area data space. <b>Action:</b> Specify a valid ALET for the server description information.
8	0F	<b>Equate Symbol:</b> IXCSRVRRSNFEATURESBADLEVEL <b>Meaning:</b> Environmental error. Program error. The content of FEATURES is not valid because the sf_level field specified a value that is reserved by XCF. <b>Action:</b> None.
8	10	<b>Equate Symbol:</b> IXCSRVRRSNLEVELBADMAX <b>Meaning:</b> Invalid parameters. Program error. The value of MAXLEVEL must be greater than or equal to the value of MINLEVEL. <b>Action:</b> Specify a correct value for MAXLEVEL.
8	11	<b>Equate Symbol:</b> IXCSRVRRSNCLIENTBADMAX <b>Meaning:</b> Environmental error. Program error. The value of MAXCLIENT must be greater than or equal to the value of MINCLIENT. <b>Action:</b> Specify a correct value for MAXCLIENT.

Table 21. Return and Reason Codes for the IXCSRVR Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	12	<b>Equate Symbol:</b> IXCSRVRRSNFDIBADVALUE <b>Meaning:</b> Environmental error. Program error. The parameter list FDI value is not valid. <b>Action:</b> FDI must be an integer value in the range 1 and 3600.
8	13	<b>Equate Symbol:</b> IXCSRVRRSNRESPBINDBADVALUE <b>Meaning:</b> Environmental error. Program error. The parameter list RESPBIND value is not valid <b>Action:</b> Specify a valid RESPBIND option.
8	14	<b>Equate Symbol:</b> IXCSRVRRSNSERVERIDBADVALUE <b>Meaning:</b> Environmental error. Program error. The SERVERID is not valid. <b>Action:</b> specify a valid server id.
8	15	<b>Equate Symbol:</b> IXCSRVRRSNSERVERIDBADSYSTEM <b>Meaning:</b> Environmental error. Program error. The SERVERID is not valid for use on the local system. For example, the server designated by the SERVERID appears to reside on some other system in the sysplex. <b>Action:</b> Correct the server id for the local system.
8	16	<b>Equate Symbol:</b> IXCSRVRRSNDDTBADSTG <b>Meaning:</b> Environmental error. Program error. The data descriptor table returned to XCF through the work area descriptor in the Server Exit Parameter List (SXPL_WAD.WAD_DataDesc.dd_DataAddr) was not accessible. As a consequence, XCF stops running the server. <b>Action:</b> None.
8	17	<b>Equate Symbol:</b> IXCSRVRRSNDDTBADALET <b>Meaning:</b> Environmental error. Program error. The ALET of a data descriptor table returned to XCF through the work area descriptor in the Server Exit Parameter List (SXPL_WAD.WAD_DataDesc.dd_DataAlet) was not valid. As a consequence, XCF stops running the server. The storage containing the data descriptor table must be in the primary address space of the server, or in a space addressable through a public entry on the dispatchable unit access list (DU-AL), or in a common area data space. <b>Action:</b> Correct the ALET for the data descriptor table.

Table 21. Return and Reason Codes for the IXCSRVR Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	1A	<b>Equate Symbol:</b> IXCSRVRRSNWORKAREABADSTG <b>Meaning:</b> Environmental error. Program error. The work area provided by the server exit routine is not accessible. As a consequence, XCF stops running the server. <b>Action:</b> None.
8	1B	<b>Equate Symbol:</b> IXCSRVRRSNWORKAREABADALET <b>Meaning:</b> Environmental error. Program error. The ALET for the work area provided by the server exit routine is not valid. As a consequence, XCF stops running the server. The storage containing the work area must be in the primary address space of the server, or in a space addressable through a public entry on the dispatchable unit access list (DU-AL), or in a common area data space. <b>Action:</b> Correct the ALET for the work area.
8	1C	<b>Equate Symbol:</b> IXCSRVRRSNMODEBADVALUE <b>Meaning:</b> Environmental error. Program error. The parameter list MODE specification is not valid. <b>Action:</b> Correct the MODE specification for the parameter list.
8	1D	<b>Equate Symbol:</b> IXCSRVRRSNSERVERIDBADSTG <b>Meaning:</b> Environmental error. Program error. The storage containing the SERVERID is not accessible. <b>Action:</b> None.
8	1E	<b>Equate Symbol:</b> IXCSRVRRSNSERVERIDBADALET <b>Meaning:</b> Environmental error. Program error. The ALET for the SERVERID is not valid. The storage containing the SERVERID must reside in the primary address space of the caller, or in a space addressable through a public entry on the dispatchable unit access list (DU-AL), or in a common area data space. <b>Action:</b> Correct the ALET for the server id.
8	1F	<b>Equate Symbol:</b> IXCSRVRRSNSCOPEBADVALUE <b>Meaning:</b> Environmental error. Program error. The parameter list SCOPE specification is not valid. <b>Action:</b> Correct the SCOPE specification for the parameter list.
8	20	<b>Equate Symbol:</b> IXCSRVRRSN#SERVERSBADVALUE <b>Meaning:</b> Environmental error. Program error. The number of servers is not valid. <b>Action:</b> The value must be non-zero

Table 21. Return and Reason Codes for the IXCSRVR Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	21	<b>Equate Symbol:</b> IXCSRVRRSNXCFSERVER <b>Meaning:</b> Environmental error. Program error. Only XCF itself is allowed to start and stop an XCF Server. <b>Action:</b> None.
8	22	<b>Equate Symbol:</b> IXCSRVRRSNSXPLRSVD <b>Meaning:</b> Environmental error. Program error. The server exit routine overlaid reserved fields in the Server Exit Parameter List (SXPL). XCF stopped the offending server instance <b>Action:</b> None.
8	23	<b>Equate Symbol:</b> IXCSRVRRSNSXPLWADRSVD <b>Meaning:</b> Environmental error. Program error. The server exit routine overlaid reserved fields in the Work Area Descriptor (SXPL_WAD) within the Server Exit Parameter List (SXPL). XCF stopped the offending server instance. <b>Action:</b> None.
8	24	<b>Equate Symbol:</b> IXCSRVRRSNSXPLRESPBIND <b>Meaning:</b> Environmental error. Program error. The server exit routine stored an invalid response bind value in the SXPL_RespBind field within the Server Exit Parameter List (SXPL). XCF stopped the offending server instance. <b>Action:</b> None.
8	25	<b>Equate Symbol:</b> IXCSRVRRSNSXPLREFUSALCODE <b>Meaning:</b> Environmental error. Program error. The server exit routine stored a nonzero refusal code in the SXPL_RefusalCode field within the Server Exit Parameter List (SXPL). The refusal code is not valid for use when processing an initialization request (SXPL_ServerCode = ixcsrvr_kSC_InitServer). XCF stopped the offending server instance. <b>Action:</b> None.
8	26	<b>Equate Symbol:</b> IXCSRVRRSNSXPLSTOPCODE <b>Meaning:</b> Environmental error. Program error. The server exit routine stored an invalid stop code in the SXPL_StopCode field within the Server Exit Parameter List (SXPL). XCF stopped the offending server instance. <b>Action:</b> None.



Table 21. Return and Reason Codes for the IXCSRVR Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	27	<p><b>Equate Symbol:</b> IXCSRVRSNSXPLRESULTCODE</p> <p><b>Meaning:</b> Environmental error. Program error. The server exit routine stored a nonzero result code in the SXPL_ResultCode field within the Server Exit Parameter List (SXPL). The result code is not valid for use when processing an initialization request (SXPL_ServerCode = ixcysrvr_kSC_InitServer) nor when processing a get work area request (SXPL_ServerCode = ixcysrvr_kSC_GetWorkArea). XCF stopped the offending server instance.</p> <p><b>Action:</b> None.</p>
8	28	<p><b>Equate Symbol:</b> IXCSRVRSNSXPLMIXEDRESULT</p> <p><b>Meaning:</b> Environmental error. Program error. The server exit routine stored a nonzero value in both the SXPL_ResultCode field and the SXPL_RefusalCode field within the Server Exit Parameter List (SXPL). Thus XCF cannot determine whether the request was accepted or whether it was refused. The relevant request is acknowledged with a response code indicating that the server exit failed while processing the request. XCF stopped the offending server instance.</p> <p><b>Action:</b> None.</p>
8	57	<p><b>Equate Symbol:</b> IXCSRVRNSHASFRR</p> <p><b>Meaning:</b> Environmental error. Program error. Caller is running with an FRR established.</p> <p><b>Action:</b> Ensure that macro is invoked while program is running in the required environment. This reason applies only to REQTYPE=START requests.</p>
8	76	<p><b>Equate Symbol:</b> IXCSRVRNSNLOCKED</p> <p><b>Meaning:</b> Environmental error. Program error. Caller holds locks. Ensure that macro is invoked while program is running in the required environment.</p> <p><b>Action:</b> Retry.</p>
8	8A	<p><b>Equate Symbol:</b> IXCSRVRNSNBADASCMode</p> <p><b>Meaning:</b> Environmental error. Program error. Caller is running neither in primary ASC mode nor in AR ASC mode.</p> <p><b>Action:</b> Ensure that macro is invoked while program is running in the required environment.</p>

Table 21. Return and Reason Codes for the IXCSRVR Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	A4	<b>Equate Symbol:</b> IXCSRVRRSNREQTYPECONFLICT <b>Meaning:</b> Environmental error. Program error. The request type in the parameter list is not consistent with the request type in register 0 on entry to the service routine. The parameter list or the register contents were corrupted before the service routine received control. <b>Action:</b> None.
8	118	<b>Equate Symbol:</b> IXCSRVRRSNNOTTASKMODE <b>Meaning:</b> Environmental error. Program error. Caller is not running in task mode. <b>Action:</b> Ensure that macro is invoked while program is running in the required environment. This reason applies only to REQTYPE=START requests.
8	11C	<b>Equate Symbol:</b> IXCSRVRRSNNOTENABLED <b>Meaning:</b> Environmental error. Program error. Caller is not running enabled. <b>Action:</b> Ensure that macro is invoked while program is running in the required environment.
8	120	<b>Equate Symbol:</b> IXCSRVRRSNXMEM <b>Meaning:</b> Caller is running cross-memory mode. <b>Action:</b> Ensure that macro is invoked while program is running in the required environment. This reason applies only to REQTYPE=START requests.
8	127	<b>Equate Symbol:</b> IXCSRVRRSNONLYONE <b>Meaning:</b> Environmental error. Program error. Current task is already instantiated as a server. <b>Action:</b> At most one server can be instantiated under any given task.
8	128	<b>Equate Symbol:</b> IXCSRVRRSNTASKTERM <b>Meaning:</b> Environmental error. Program error. Current task is ending. <b>Action:</b> A server cannot be started once task termination begins.
8	129	<b>Equate Symbol:</b> IXCSRVRRSNRESMGR <b>Meaning:</b> Environmental error. Program error. The caller is a resource manager. <b>Action:</b> A server cannot be started by an (address space) resource manager.

Table 21. Return and Reason Codes for the IXCSRVR Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	OCA1	<b>Equate Symbol:</b> IXCSRVRRSNNOUSERSTORAGE <b>Meaning:</b> Environmental error. Unable to obtain user storage needed to process the request. <b>Action:</b> None.
C	OCA2	<b>Equate Symbol:</b> IXCSRVRRSNNOXCFSTORAGE <b>Meaning:</b> Environmental error. Unable to obtain XCF storage needed to process the request. <b>Action:</b> None.
C	OCA3	<b>Equate Symbol:</b> IXCSRVRRSNMAXSERVERS <b>Meaning:</b> Environmental error. Unable to define new server. Maximum number of servers already defined. <b>Action:</b> None.
C	OCA4	<b>Equate Symbol:</b> IXCSRVRRSNNOSYSRESOURCES <b>Meaning:</b> Environmental error. Unable to obtain the system resources needed to process the request. <b>Action:</b> None.
10	None	<b>Equate Symbol:</b> None. <b>Meaning:</b> Failure in XCF processing. <b>Action:</b>

## Example

For an example of starting a client/server application using IXCSRVR, see "Using XCF for Client/Server Communication" in *z/OS MVS Programming: Sysplex Services Guide*.



## Chapter 25. IXCSYSCL – Notify the System that Cleanup Has Completed

### Description

When a system leaves the sysplex, automatic restart management will wait for all the members that specified SYSCLEANUPMEM=YES on the IXCJOIN macro to issue the IXCSYSCL macro before it will perform restarts for elements that were on the failed system.

When a system leaves the sysplex, members of cross-system coupling facility (XCF) groups will be notified through their group user routine. If SYSCLEANUPMEM=YES was specified by a member of the group, IXCSYSCL must be issued to indicate when cleanup has been completed by an XCF member (specified in the MEMTOKEN parameter), for the system that left the sysplex (specified in the FAILEDSYS parameter), or that no cleanup is necessary.

**Note:** Automatic restart management will not wait forever for cleanup processing to be completed. After a certain amount of time, automatic restart management will proceed with restarts regardless of whether there are any outstanding IXCSYSCL macros to be issued.

### Environment

The following are the environment requirements for the caller.

Environment	Environment requirement
Minimum authorization:	Supervisor state or PKM allowing key 0 - 7
Dispatchable unit mode:	Task or SRB
Cross memory mode:	PASN=HASN, any SASN
AMODE:	31-bit
ASC mode:	Primary or access register (AR)
Interrupt status:	Enabled for I/O and external interrupts
Locks:	No locks held
Control parameters:	Must be in the primary address space or be in an address/data space that is addressable through a public entry in the caller's dispatchable unit access list (DU-AL)

### Programming Requirements

If the program is in AR mode, issue the SYSSTATE ASCENV=AR macro before IXCSYSCL. SYSSTATE ASCENV=AR tells the system to generate code appropriate for AR mode.

### Restrictions

This macro must be issued from the same address space from which the IXCJOIN was issued.

### Input Register Information

Before issuing the IXCSYSCL macro, the caller does not have to place any information into any register unless using it in register notation for a particular parameter, or using it as a base register.

## Output Register Information

---

When control returns to the caller, the general purpose registers (GPRs) contain:

### Register Contents

**0**

If GPR 15 contains a zero, GPR 0 is used as a work register by the system; otherwise, GPR 0 contains a reason code.

**1**

Used as a work register by the system.

**2-13**

Unchanged.

**14**

Used as a work register by the system.

**15**

Return code.

When control returns to the caller, the access registers (ARs) contain:

### Register Contents

**0-1**

Used as work registers by the system

**2-13**

Unchanged

**14-15**

Used as a work register by the system

Some callers depend on register contents remaining the same before and after issuing a service. If the system changes the contents of registers on which the caller depends, the caller must save them before issuing the service, and restore them after the system returns control.

## Performance Implications

---

None.

## Understanding IXCSYSCL Version Support

---

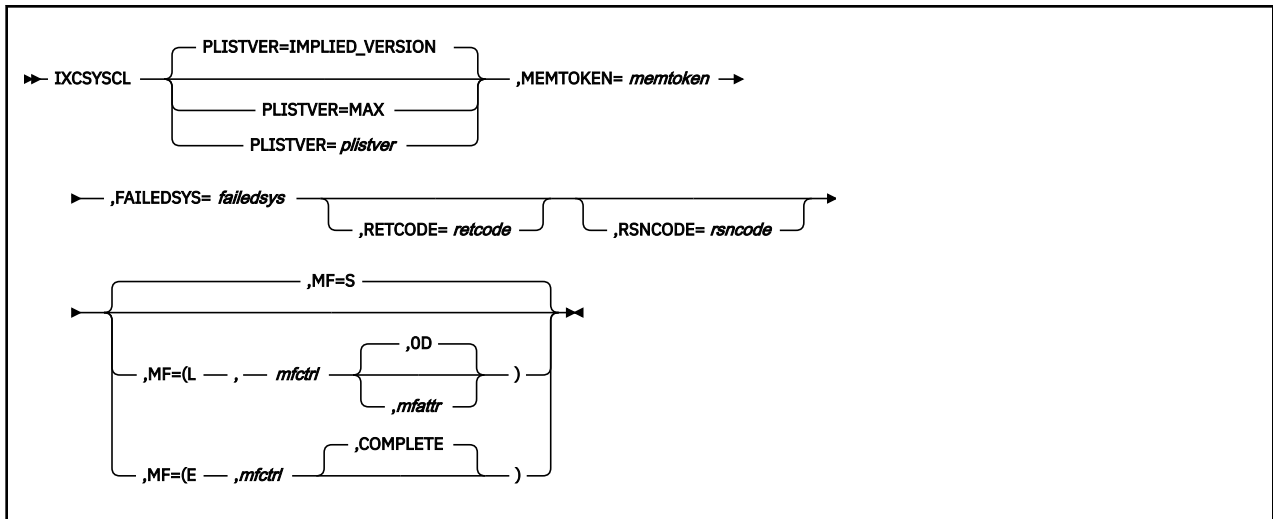
The IXCSYSCL macro supports version 0.

Specify the version of the parameter list that you want generated with the PLISTVER keyword. See Chapter 2, “Specifying a Macro Version Number,” on page 5 for considerations when specifying the version of the parameter list with PLISTVER.

## Syntax Diagram

---

The syntax of the IXCSYSCL macro is as follows:



## Parameter Descriptions

The parameter descriptions are listed in alphabetical order. Default values are underlined:

### **,FAILEDYS=failedsys**

Use this input parameter to specify the name of the field that contains the system token of the system for which the cleanup has been completed. This token is from the group exit parameter list (mapped by IXCYGEPL) in the field GEPLSID. The group exit was given control when a system was removed from the sysplex.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 4-byte field that contains the token for the system that left the sysplex.

### **,MEMTOKEN=memtoken**

Use this input parameter to specify the name of the field that contains the member token for the member issuing this macro. This member must be active on the current system, and must have issued the IXCJOIN macro with SYSCLEANUPMEM=YES specified.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-byte field that contains the member token.

### **,MF=S**

#### **,MF=(L,mfctrl)**

#### **,MF=(L,mfctrl,mfattr)**

#### **,MF=(L,mfctrl,OD)**

#### **,MF=(M,mfctrl)**

#### **,MF=(M,mfctrl,COMPLETE)**

#### **,MF=(M,mfctrl,NOCHECK)**

#### **,MF=(E,mfctrl)**

#### **,MF=(E,mfctrl,COMPLETE)**

#### **,MF=(E,mfctrl,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into

the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

**,mfctrl**

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

**,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

**,COMPLETE**

**,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

**COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if *SMILE=var* were an optional parameter and the default is *SMILE=NO\_SMILE* then it would not be documented. However, if the default was *SMILE=-:-*, then it would be documented because a value would be the default.

**NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

**,PLISTVER=IMPLIED\_VERSION**

**,PLISTVER=MAX**

**,PLISTVER=*plistver***

Use this input parameter to specify the version of the macro. See “[Understanding IXCSYSCL Version Support](#)” on page 406 for a description of the options available with *PLISTVER*.

**,RETCODE=*retcode***

Use this output parameter to specify the location in which the system is to copy the return code from GPR 15.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 4-byte field that will contain the return code when the request is complete.

**,RSNCODE=*rsncode***

Use this output parameter to specify the location in which the system is to copy the reason code from GPR 0.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 4-byte field that will contain the reason code (if any) when the request completes.

## ABEND Codes

---

None.

## Return and Reason Codes

---

When the IXCSYSCL macro returns control to your program:

- GPR 15 (and *retcode*, if you coded *RETCODE*) contains a return code.
- GPR 0 (and *rsncode*, if you coded *RSNCODE*) contains a reason code, if applicable.



Macro IXCYCON provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

<b>0</b>	IXCRETCODEOK
<b>4</b>	IXCRETCODEWARNING
<b>8</b>	IXCRETCODEPARMERROR
<b>C</b>	IXCRETCODEENVERROR
<b>10</b>	IXCRETCODECOMPERROR

The following table identifies the hexadecimal return and reason codes and the equate symbol associated with each reason code.

<i>Table 22. Return and Reason Codes for the IXCSYSCL Macro</i>		
<b>Hexadecimal Return Code</b>	<b>Hexadecimal Reason Code</b>	<b>Equate Symbol Meaning and Action</b>
00	None.	<b>Meaning:</b> The request was accepted for processing. <b>Action:</b> None.
08	04	<b>Equate Symbol:</b> IXCSYSCLRSNNOTACTIVE <b>Meaning:</b> Program error. The member token specified is not for an active member in the primary address space. <b>Action:</b> The member token was returned by IXCJOIN when this member joined the XCF group. Try one or more of the following: <ul style="list-style-type: none"> <li>• Verify the token passed.</li> <li>• Ensure that this request is running in the correct address space.</li> <li>• Ensure that this request is for an active member.</li> <li>• Verify that if IXCSYSCL was called in AR mode, the SYSSTATE ASCENV=AR macro was issued prior to this macro.</li> <li>• Verify that if member token was specified using explicit register notation, the corresponding access register was updated accordingly.</li> </ul>
08	08	<b>Equate Symbol:</b> IXCSYSCLRSNINAPPROPRIATEPRIMARY <b>Meaning:</b> Program error. The primary address space is neither the master scheduler address space nor the primary address space of the caller of the IXCJOIN request. <b>Action:</b> This macro must be issued from the same primary address space as when the IXCJOIN was issued. Ensure that: <ul style="list-style-type: none"> <li>• The parameter list has not been overlaid.</li> <li>• This request is running in the correct address space.</li> <li>• If IXCSYSCL was called in AR mode, the SYSSTATE ASCENV=AR macro was issued prior to this macro.</li> </ul>

Table 22. Return and Reason Codes for the IXCSYSCL Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
08	0C	<p><b>Equate Symbol:</b> IXCSYSCLRSNSYSCLEANUPMEMNO</p> <p><b>Meaning:</b> Program error. The member token specified in MEMTOKEN is not for a member that specified SYSCLEANUPMEM=YES on the IXCJOIN macro.</p> <p><b>Action:</b> The IXCSYSCL macro should be issued only by a member that joined the XCF group with SYSCLEANUPMEM=YES specified. If this return code is unexpected, then ensure that:</p> <ul style="list-style-type: none"> <li>• The member token has not been overlaid.</li> <li>• The correct member token is being used.</li> <li>• If IXCSYSCL was called in AR mode, the SYSSTATE ASCENV=AR macro was issued prior to this macro.</li> <li>• If the member token was specified using explicit register notation, the corresponding access register was updated accordingly.</li> </ul>
08	10	<p><b>Equate Symbol:</b> IXCSYSCLRSNFAILEDYSNOTVALID</p> <p><b>Meaning:</b> Program error. The system token specified for FAILEDYS is not valid.</p> <p><b>Action:</b> The system token is from the group exit parameter list (IXCYGEPL). Ensure that:</p> <ul style="list-style-type: none"> <li>• The system token has not been overlaid.</li> <li>• If IXCSYSCL was called in AR mode, the SYSSTATE ASCENV=AR macro was issued prior to this macro.</li> <li>• If the address space token was specified using explicit register notation, the corresponding access register was updated accordingly.</li> </ul>
08	40	<p><b>Equate Symbol:</b> IXCSYSCLRSNPLISTRSDNOTVALID</p> <p><b>Meaning:</b> Program error. A reserved field in the parameter list is not zero.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The parameter list was not inadvertently overlaid.</li> <li>• The parameter list was initialized before it was used.</li> <li>• If IXCSYSCL was called in AR mode, the SYSSTATE ASCENV=AR macro was issued prior to this macro.</li> <li>• Your program was assembled with the correct macro library for the release of MVS your program is running on.</li> <li>• The correct parameter list version was specified.</li> </ul>

Table 22. Return and Reason Codes for the IXCSYSCL Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
08	100	<p><b>Equate Symbol:</b> IXCSYSCLRSNPLISTBADALET</p> <p><b>Meaning:</b> Program error. The ALET that qualifies the address of the parameter list is not valid.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The address of the parameter list has not been overlaid.</li> <li>• If IXCSYSCL was called in AR mode, the SYSSTATE ASCENV=AR macro was issued prior to this macro.</li> <li>• If the parameter list address was specified using explicit register notation, the corresponding access register was updated accordingly.</li> </ul>
08	104	<p><b>Equate Symbol:</b> IXCSYSCLRSNPLISTVERSIONNOTVALID</p> <p><b>Meaning:</b> Program error. The version number specified in the IXCSYSCL parameter list is not valid.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• Your program did not overlay the parameter list storage.</li> <li>• Your program was assembled with the correct macro library for the release of MVS your program is running on.</li> <li>• The correct parameter list version was specified.</li> </ul>
08	108	<p><b>Equate Symbol:</b> IXCSYSCLRSNPLISTBADFUNCTION</p> <p><b>Meaning:</b> Program error. The parameter list is not valid.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• Your program did not overlay the parameter list storage.</li> <li>• Your program was assembled with the correct macro library for the release of MVS your program is running on.</li> <li>• If IXCSYSCL was called in AR mode, the SYSSTATE ASCENV=AR macro was issued prior to this macro.</li> <li>• If the parameter list address was specified using explicit register notation, the corresponding access register was updated accordingly.</li> </ul>

Table 22. Return and Reason Codes for the IXCSYSCL Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
08	10C	<p><b>Equate Symbol:</b> IXCSYSCLRSNPLISTBADSTG</p> <p><b>Meaning:</b> Program error. An error occurred when the system tried to access the parameter list.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• Ensure that the parameter list address has not been overlaid.</li> <li>• Ensure that the correct parameter list storage area was specified.</li> <li>• If your program is running in AR ASC mode ensure that: <ul style="list-style-type: none"> <li>– You specified SYSSTATE ASCENV=AR before issuing the IXCSYSCL macro.</li> <li>– If the parameter list address was specified using explicit register notation, the corresponding access register was updated accordingly.</li> <li>– The parameter list is either in the primary address space or in an address/data space that is addressable through a public entry on the caller's DU-AL.</li> </ul> </li> <li>• Ensure that the parameter list storage area was not inadvertently freed by your program.</li> <li>• Ensure that your program was assembled with the correct macro library for the release of MVS your program is running on.</li> </ul>
08	11C	<p><b>Equate Symbol:</b> IXCSYSCLRSNNOTENABLED</p> <p><b>Meaning:</b> The caller is not enabled.</p> <p><b>Action:</b> Correct your program so that it does not issue IXCSYSCL while it is disabled.</p>
08	12C	<p><b>Equate Symbol:</b> IXCSYSCLRSNLOCKHELD</p> <p><b>Meaning:</b> Program error. The caller of IXCSYSCL holds a lock.</p> <p><b>Action:</b> Correct your program so that it does not issue IXCSYSCL while it is holding a lock.</p>
10	None.	<p><b>Meaning:</b> System error. The system experienced an unexpected error while processing this request.</p> <p><b>Action:</b> Retry the request at least once. If the problem persists, record the return and reason code, and supply them to the appropriate IBM support personnel.</p>

## Example

```

      TITLE  'XCNSGY20- Sample IXCSYSCL macro usage'
XCNSGY20 CSECT
XCNSGY20 AMODE 31
XCNSGY20 RMODE ANY
*/ * START OF SPECIFICATIONS *****
*
*
```

```

*01* MODULE-NAME = XCNSGY20
*
*02*   DESCRIPTIVE-NAME = Sample IXCSYSL macro usage
*
*   STATUS = HBB5520
*
*01* FUNCTION =
*   Sample program to illustrate use of IXCSYSL macro by a
*   multi-system application that becomes a XCF group member and
*   needs to perform system-wide cleanup after a system leaves
*   the sysplex.
*
*02*   OPERATION =
*
*       (1) Get into Supervisor state Key 0
*
*       (2) Load group exit routine
*
*       (3) Join a XCF group and tell XCF that this group member
*           performs system-wide cleanup after a system leaves the
*           sysplex. This is done by invoking IXCJOIN macro with
*           SYSCLEANUPMEM=YES and specifying a group exit such that
*           XCF can notify us that a system has left the sysplex.
*
*       (4) Wait for our group exit to tell us that it is time to
*           do system-wide cleanup for a system that has left the
*           sysplex. This will be done by waiting on the ECB passed
*           to the group exit.
*
*       (5) Do the necessary system-wide cleanup. This example will
*           just issue a WTO saying it is doing system_wide cleanup.
*
*       (6) Tell XCF this XCF group member has completed system-wide
*           cleanup. This is accomplished by invoking the IXCSYSL
*           macro service and telling XCF which system the member has
*           completed system-wide cleanup for.
*
*
*       (7) Leave the XCF group.
*
*       (8) Delete group exit routine
*
*       (9) Return to problem state key 8
*
*****
*02*   RECOVERY-OPERATION = This program functions without recovery.
*
*****
*01* NOTES =
*
*   (1) Sample install Linkedit JCL:
*
*       //LINK      EXEC PGM=IEWL,
*       //   PARM='RENT,REFR,XREF,LET,LIST,NCAL,SIZE=(750K,200K)'
*       //SYSUT1    DD UNIT=SYSDA,SPACE=(1024,(200,20))
*       //OBJLIB    DD DSN=userid.my.obj,DISP=SHR
*       //SYSLMOD   DD DSN=lnklst.lib,DISP=OLD
*       //SYSPRINT  DD SYSOUT=*
*       //SYSLIN   DD *
*                   INCLUDE OBJLIB(XCNSGY20)
*                   ENTRY XCNSGY20
*                   NAME XCNSGY20(R)
*
*   (2) Sample execution JCL to run the example out of a authorized
*       library in the linklist concatenation:
*
*       //RUNIT     EXEC PGM=XCNSGY20
*       //SYSABEND  DD SYSOUT=*
*
*02*   DEPENDENCIES = None
*
*02*   RESTRICTIONS =
*       None
*
*02*   REGISTER-CONVENTIONS =
*
*03*   REGISTER-USAGE = See register declarations in code

```

```

*
*02*  PATCH-LABEL = None
*

*01*  MODULE-TYPE = CSECT
*
*02*  PROCESSOR = Assembler-H
*
*02*  MODULE-SIZE = See assembler External Symbol Dictionary
*
*02*  ATTRIBUTES =
*
*03*    LOCATION = User private
*
*03*    RMODE = Any
*
*03*    TYPE = Reentrant
*
*****
*
*01*  ENTRY-POINT = XCNSGY20
*
*02*  PURPOSE = See FUNCTION section for this module.
*
*03*    OPERATION = See OPERATION section for this module.
*
*02*  LINKAGE = BALR
*
*03*    CALLERS = Any
*
*02*  ATTRIBUTES =
*
*03*    ENTRY
*
*04*      ENABLED
*04*      STATE = Problem program
*04*      KEY = User key
*04*      AMODE = 31
*04*      LOCKS HELD = None
*04*      ASC MODE = Primary
*04*      MEMORY MODE = Primary equal to Secondary equal to Home
*04*      DISPATCH MODE = Task
*
*03*    EXECUTION
*
*04*      ENABLED
*04*      STATE = Supervisor State
*04*      KEY = Key zero
*04*      AMODE = 31
*04*      LOCKS OBTAINED = None
*04*      ASC MODE = Primary
*04*      MEMORY MODE = Primary equal to Secondary equal to Home
*02*    SERIALIZATION = None
*
*02*  INPUT = None
*

*03*  ENTRY-REGISTERS =
*
*    R0      = Irrelevant
*    R1      = Irrelevant
*    R2 - R12 = Irrelevant
*    R13     = Address of a standard save area
*    R14     = Return address
*    R15     = Entry point address
*
*02*  OUTPUT = WTOs
*
*02*  EXIT-NORMAL = Return to caller
*
*03*  CONDITIONS = The multisystem application has performed
*                  system-wide cleanup for a system removed from the
*                  sysplex.
*
*03*  EXIT-REGISTERS = N/A
*
*03*  RETURN-CODES =
*    R15 - 00 if expected return codes from IXCJOIN, IXCSYSL
*             and IXCLEAVE macro services

```

```

*
*           - 08  if unexpected return codes from IXCJOIN,
*                IXCSYSL, and IXCLEAVE macro services
*
*02*  EXIT-ERROR =  None
*
*****
*
*01*  EXTERNAL-REFERENCES  =
*
*
*02*  ROUTINES = None
*
*02*  DATA-AREAS = None
*
*02*  CONTROL-BLOCKS =
*
*   Common  Mapping
*   Name    Macro      Usage      Full Name
*   -----
*   QUA    IXCYQUAA  Read      IXCYQUAA Macro Service ANSAREA
*                               mappings
*
*
*

```

```

*01*  MACROS-EXECUTABLE =
*
*           IXCJOIN
*           IXCLEAVE
*           IXCSYSL
*           STORAGE
*           WTO
*
*01*  SERIALIZATION = None.
*
*01*  MESSAGES =
*
*           The following WTOs may be issued to job log:
*           XCNSGY20 IXCJOIN  RETCODE=rrrrrrrr RSNCODE:ssssssss
*           XCNSGY20 NOW IN GROUP=ggggggggg MEMBER=mmmmmmmmmmmmmmmmmmmm
*           XCNSGY20 DOING SYSTEM-WIDE CLEANUP FOR cccccccc
*           XCNSGY20 IXCSYSL  RETCODE=rrrrrrrr RSNCODE:ssssssss
*           XCNSGY20 IXCLEAVE RETCODE=rrrrrrrr RSNCODE:ssssssss
*
*01*  ABEND-CODES =  None
*
*01*  WAIT-STATE-CODES = None
*
*01*  CHANGE-ACTIVITY = None
*
****  END OF SPECIFICATIONS *****/
      EJECT
*****
*
*   Standard entry linkage
*
*****
      STM      R14,R12,12(R13)
      BALR     BASEREG1,0           Establish addressability
      USING   *,BASEREG1
      MODID    BR=YES
      LR       R3,R13              Save callers savearea address
      STORAGE  OBTAIN,ADDR=(DATAREG1),SP=0,LENGTH=DYNASIZE
      LA       DATAREG2,4095(,DATAREG1) Set Second Data Register
      USING    DYNA,DATAREG1        First Data Register
      USING    DYNA+4095,DATAREG2   Second Data Register
      ST       R3,SAVEAREA+4        Save @ of callers savearea
      ST       DATAREG1,8(,R3)      Chain our savearea to callers
      EJECT
*****
*
*   Initialize variables
*
*****
      MVC      EXITRC,=AL4(GOODRETC) * Initialize return code
      MVC      WTOEXEC(LENWTOS),WTOS * Copy static parm list to dynamic
      MVC      WTOTXTD1(L'WTOTXTD1),WTOTXTS1 * Prime WTO text length

```

```

*-----
* Set up MEMDATA for the XCF group member
*-----

```

```

        XC      DATAGE(16),DATAGE      Clear area used by group exit
        LA      R3,MYECB              Get address of ECB
        ST      R3,MYECB_PTR          Save address for group exit
        LA      R3,DATAGE             Get address of data area to
*                                     be used by group exit to pass
*                                     information back to this task
        ST      R3,DATAGE_PTR          Save address in first word of
*                                     data area to be used by group
*                                     exit to pass info back to this
*                                     task
        EJECT
*****
* (1) Get into Key 0 , Supervisor State
*
*****
        MODESET KEY=ZERO,MODE=SUP
        EJECT
*****
* (2) Load Group exit routine.
*
*****
        LOAD EP=XCNSGY21
        LR      R2,R0                  Save address of group exit
        EJECT
*****
* (3) Join a XCF group and tell XCF that this group member
*      performs system-wide cleanup after a system leaves the
*      sysplex. This is done by invoking IXCJOIN macro with
*      SYSCLEANUPMEM=YES and specifying a group exit such that
*      XCF can notify us that a system has left the sysplex.
*
*****
        IXCJOIN GRPNAME=MYGRPNAM,      Name of XCF group to be used
        MEMNAME=MYMEMNAM,              Member name to be used
        GRPEXIT=(R2),                  Group exit routine
        ANSLN=QUAMLENG,                Answer area length : One QUAMEM
        ANSAREA=QUAMEM,                Answer area (returned QUAMEM)
        MEMDATA=MYMDATA,               Member data
        SYSCLEANUPMEM=YES,             Member does System-wide cleanup
        RETCODE=SAVERET,               Return code
        RSNCODE=SAVERSN,               Reason code
        MF=(E,JOINPL)
*
* Issue a WTO that shows the IXCJOIN RETCODE/RSNCODE
        MVC     WTOTXTD2(L'WTOTXTD2),WTOSERV * Set message text
        MVC     MAPSERV(8),=CL8'IXCJOIN ' * Service invoked
* Convert hex return code to printable hex
        MVC     PHEXIN(4),SAVERET      Hex return code to convert
        UNPK    PHEXOUT,PHEXIN         Unpack the data
        MVC     MAPRETC(8),PHEXOUT+1   Store unpacked data into target
        TR      MAPRETC(8),TRTBL-240   Translate to printable hex
* Convert hex reason code to printable hex
        MVC     PHEXIN(4),SAVERSN      Hex reason code to convert
        UNPK    PHEXOUT,PHEXIN         Unpack the data
        MVC     MAPRSNC(8),PHEXOUT+1   Store unpacked data into target
        TR      MAPRSNC(8),TRTBL-240   Translate to printable hex
        BAL     R14,ISSUEWTO           Tell user IXCJOIN RETCODE/RSNCODE
*****
* Check for successfully joining XCF group.
*
*****
        L       R3,SAVERET             Get return code
        C       R3,=AL4(4)             Member is now in active state
        BH      BADJOIN                Bad return code from IXCJOIN
*****
* Joined XCF group successfully.
*
*****
        MVC     MYMEMTOK(8),QUAMTOKN   Save my XCF member token
*
* Issue a WTO that shows the XCF group and member name join was for
        MVC     WTOTXTD2(L'WTOTXTD2),WTOJOIN * Set message text
        MVC     MAPGRPNM(8),MYGRPNAM   * Set group name
        MVC     MAPMEMNM(16),MYMEMNAM   * Set member name
        BAL     R14,ISSUEWTO           Tell user XCF group & member name
        EJECT
*****
* (4) Wait for our group exit to tell us that it is time to
*      do system-wide cleanup for a system that has left the
*      sysplex. This will be done by waiting on the ECB passed

```



```

*      to the group exit.
*
*****
      LA      R3,MYECB          Point to ECB to wait on
      WAIT   ECB=(R3)          Wait for group exit to tell us it
*                               is time to do system-wide cleanup
*                               for some other system
      SPACE 2
*****
* (5) Do the necessary system-wide cleanup. This example will
*      just issue a WTO saying it is doing system-wide cleanup and
*      identify name of system that group exit told us we are cleaning
*      up for. Group exit sets the value of SYSNAME.
*
*****
* Issue a WTO that shows that member is doing system-wide cleanup
      MVC     WTOTXTD2(L'WTOTXTD2),WTOCLEAN * Set message text
      MVC     MAPSYSNM(8),SYSNAME Copy Sysname to WTO text area
      BAL     R14,ISSUEWTO      Say doing system-wide cleanup
      EJECT
*****
* (6) Tell XCF this XCF group member has completed system-wide
*      cleanup. This is accomplished by invoking the IXCSYSCL
*      macro service and telling XCF which system the member has
*      completed system-wide cleanup for.
*
*****
      IXCSYSCL MEMTOKEN=MYMEMTOK, XCF group member token
      FAILEDYS=MYSYSTOK, Failed system's system token
      RETCODE=SAVERET, Return code from IXCSYSCL
      RSNCODE=SAVERSN, Reason code from IXCSYSCL
      MF=(E,SYSCLPL)

*
* Issue a WTO that shows the IXCSYSCL RETCODE/RSNCODE
      MVC     WTOTXTD2(L'WTOTXTD2),WTOSERV * Set message text
      MVC     MAPSERV(8),=CL8'IXCSYSCL' * Service invoked
* Convert hex return code to printable hex
      MVC     PHEXIN(4),SAVERET Hex return code to convert
      UNPK    PHEXOUT,PHEXIN Unpack the data
      MVC     MAPRETC(8),PHEXOUT+1 Store unpacked data into target
      TR      MAPRETC(8),TRTBL-240 Translate to printable hex
* Convert hex reason code to printable hex
      MVC     PHEXIN(4),SAVERSN Hex reason code to convert
      UNPK    PHEXOUT,PHEXIN Unpack the data
      MVC     MAPRSNC(8),PHEXOUT+1 Store unpacked data into target
      TR      MAPRSNC(8),TRTBL-240 Translate to printable hex
      BAL     R14,ISSUEWTO Tell user IXCSYSCL RETCODE/RSNCODE
      EJECT
*****
* Check for successfully telling XCF that this member has
*      completed system-wide cleanup for the failed system.
*
*****
      L       R3,SAVERET Get return code
      C       R3,=AL4(0) Request was accepted by XCF
      BE      LEAVEGRP Good return code from IXCSYSCL
*****
* XCF did not accept the IXCSYSCL macro response.
*****
      MVC     EXITRC,=AL4(BADRETC) Set bad return code
      EJECT
*****
* (7) Leave the XCF group.
*
*****
LEAVEGRP EQU * Label to leave XCF group
      IXCLEAVE MEMTOKEN=MYMEMTOK, XCF member token
      RETCODE=SAVERET, Return code from IXCLEAVE
      RSNCODE=SAVERSN Reason code from IXCLEAVE
      MF=(E,LEAVEPL)

*
* Issue a WTO that shows the IXCLEAVE RETCODE/RSNCODE
      MVC     WTOTXTD2(L'WTOTXTD2),WTOSERV * Set message text
      MVC     MAPSERV(8),=CL8'IXCLEAVE' * Service invoked
* Convert hex return code to printable hex
      MVC     PHEXIN(4),SAVERET Hex return code to convert
      UNPK    PHEXOUT,PHEXIN Unpack the data
      MVC     MAPRETC(8),PHEXOUT+1 Store unpacked data into target
      TR      MAPRETC(8),TRTBL-240 Translate to printable hex
* Convert hex reason code to printable hex

```

```

MVC      PHEXIN(4),SAVERSN      Hex reason code to convert
UNPK     PHEXOUT,PHEXIN         Unpack the data
MVC      MAPRSNC(8),PHEXOUT+1    Store unpacked data into target
TR       MAPRSNC(8),TRTBL-240    Translate to printable hex
BAL      R14,ISSUEWTO           Tell user IXCLEAVE RETCODE/RSNCODE
B        DELGEXIT               Delete the group exit
BADJOIN  EQU *                   IXCJOIN got a bad return code
MVC      EXITRC,=AL4(BADRETC)    Set bad return code
EJECT

```

```

*****
* (8) Delete Group exit routine                                     *
*                                                                 *
*****
DELGEXIT EQU *
        DELETE EP=XCNSGY21
        EJECT
*****
* (9) Return to Key 8, Problem State                               *
*                                                                 *
*****
COMPLETE EQU *
        MODESET KEY=NZERO,MODE=PROB
        EJECT
*****
* Exit linkage                                                     *
*                                                                 *
*****
L        R2,SAVEAREA+4      Save caller's save area address
L        R3,EXITRC          Save testcase return code
STORAGE RELEASE,ADDR=((DATAREG1)),LENGTH=DYNASIZE
LR       R13,R2             Restore caller's save area address
L        R14,12(R13)        Restore Return address
LR       R15,R3             Set testcase return code
LM       R0,R12,20(R13)     Restore Registers R0-R12
BR       R14                Return to caller
*****
*                                                                 *
* Subroutine: ISSUEWTO                                           *
*                                                                 *
* Function : This routine is called to issue a WTO.             *
*                                                                 *
* Input    : WT0TXTD1 contains text for WTO message to be issued *
*                                                                 *
*****
ISSUEWTO EQU *
        STM      R14,R12,SAVE1      Save callers regs
        LA       R5,WT0TXTD1        Address WTO parmlist
        WTO TEXT=(R5),ROUTCDE=(11),MF=(E,WTOEXEC) * Issue WTO
        LM       R14,R12,SAVE1      Restore callers regs
        BR       R14                Return to caller
        EJECT

```

```

*****
* Register declares                                               *
*                                                                 *
*****
R0       EQU      0
R1       EQU      1
R2       EQU      2
R3       EQU      3
R4       EQU      4
R5       EQU      5
R6       EQU      6
R7       EQU      7
R8       EQU      8
R9       EQU      9
R10      EQU      10
*
* DATAREG2 EQU      11      Reserved for future expansion
* BASEREG1 EQU      12      of the code or the dynamic area
* R12       EQU      12      Second data register
* DATAREG1 EQU      13      Code register
* R13       EQU      13
* R14       EQU      14      First data register
* R15       EQU      15
* EJECT
*****

```

```

*
*   Static data
*
*****
DS      0F
TRTBL   DC      CL16'0123456789ABCDEF' * Translate table
ZERO    DC      F'0'                  * Constant zero for comparisons
SPACE   1
*****
*
*   Static WTO data
*
*****
WTOS     WTO TEXT=,ROUTCDE=(11),MF=L * Static form of WTO
LENWTOS  EQU *-WTOS                  * Length of WTO parm list
WTOTXTS1 DC      AL2(L'WTOTXTD2)      * WTO text length
WTOSERV  DC CL65'XCNSGY20 mmmmmmm RETCODE=rrrrrrr RSNCODE=sssssss'
MAPSERV  EQU  WTOTXTD2+9,8,C'C'        * Map service WTO insert
MAPRETC  EQU  WTOTXTD2+26,8,C'C'       * Map service RETCODE insert
MAPRSNC  EQU  WTOTXTD2+43,8,C'C'       * Map service RETCODE insert
WTOCLEAN DC CL65'XCNSGY20 DOING SYSTEM-WIDE CLEANUP FOR ccccccc '
MAPSYSNM EQU  WTOTXTD2+39,8,C'C'       * Map system name insert
WTOJOIN  DC CL65'XCNSGY20 NOW IN GROUP=ggggggg MEMBER=mmmmmmmmmmmmmm'
MAPGRPNM EQU  WTOTXTD2+22,8,C'C'       * Map group name insert
MAPMEMNM EQU  WTOTXTD2+38,8,C'C'       * Map member name insert
EJECT

*****
*
*   Constants used to identify the application to XCF.
*
*****
MYGRPNAM DC      CL8'XCNSGY20'         Application group name to join
*                                         XCF group with
MYMEMNAM DC      CL16'M1               ' Application member name to join
*                                         XCF group with
*                                         WTOTXTD2 fields)
LTORG
EJECT
*****
*
*   Dynamic data
*
*****
DYNA      DSECT
SAVEAREA DS      18F                  Standard savearea (first field)
SPACE     1
*****
*   Member data passed to group exit via MEMDATA keyword
*
*****
DS      0D                            Ensure boundary alignment
MYMDATA  DS      CL8                  Member data provided to group
*                                         when member needs to do
*                                         system-wide cleanup
DATAGE_PTR EQU MYMDATA+0,4,C'A'        Address of data area to be used
*                                         by group exit to pass info back
*                                         to this task
SPACE     1
*****
*   Data area to be used by group exit to pass info back to this task
*
*****
DS      0D                            Ensure boundary alignment
DATAGE  DS      CL16                  Data area to be used by
*                                         group exit to pass info back to
*                                         this task
SYSNAME  EQU DATAGE+0,8,C'C'          System name being cleanup up for
MYSYSTOK EQU DATAGE+8,4,C'F'          System token to cleanup for
MYECB_PTR EQU DATAGE+12,4,C'A'        ECB that group exit should post
SPACE     1
DS      0D                            Ensure boundary alignment
MYGE_ADDR DS      A                  Address of group exit
MYMEMTOK DS      XL8                Member token of XCF group member
PHEXIN   DS      CL5                Work area for printable hex conv
PHEXOUT  DS      CL10               Work area for printable hex conv
MYECB    DS      F                  ECB for group exit to post
SAVERET  DS      F                  Save macro service return code
SAVERSN  DS      F                  Save macro service reason code
SAVE1    DS      15F                First level subroutine savearea
SAVE2    DS      15F                Second level subroutine savearea
EXITRC   DS      F                  Module Return code
BADRETC  EQU      8                  Bad return code from testcase

```

```
GOODRETC EQU      0                      Good return code from testcase
EJECT
```

```
*****
*
*   Dynamic WTO storage
*
*****
WTOEXEC  WTO TEXT=,ROUTCDE=(11),MF=L * List form of WTO parmlist
WTOTXTD1 DC      AL2(L'WTOTXTD1)      * WTO text length
WTOTXTD2 DC      CL65' '              * WTO text
      SPACE 2
*****
*   Dynamic storage for parmlists
*
*****
      SPACE 2
      IXCJOIN MF=(L,JOINPL)
      IXCSYSCL MF=(L,SYSCPL)
      IXCLEAVE MF=(L,LEAVEPL)
*****
*   Dynamic storage for a answer area to hold a QUAMEM record
*   recorded by IXCJOIN.
*
*****
      IXCYQUAA DSECT=NO,HEADER=NO,MEMBER=YES,SYSTEM=NO,
      GROUP=NO,CF=NO,CFSC=NO,CFSTR=NO,STR=NO,
      STRPL=NO,STRXL=NO,STRCF=NO,STRUSER=NO,ARMS=NO
      SPACE 2
*****
*
*   End of dynamic storage
*
*****
DYNASIZE EQU      *-DYNA                Total size of dynamic storage
EJECT
END
```

## Chapter 26. IXCTERM – Terminate a Member of an XCF Group

### Description

The IXCTERM macro terminates an active member (called the target member) of a cross-system coupling facility (XCF) group.

The target member is terminated in accordance with the TERMLEVEL value specified when that member invoked IXCJOIN to become an active member of the group.

- For TERMLEVEL=MEMASSOC, the system abnormally ends the associated task, which is either the task that issued the IXCJOIN causing the target member to become active, or the jobstep task that was current at the time the IXCJOIN was issued. Which type of task the member is associated with is specified by the MEMASSOC parameter on the IXCJOIN.

**Note:** The IXCTERM request is rejected with return code X'8', reason code X'120' if the target member is associated with an address space (MEMASSOC=ADDRSPACE).

- For TERMLEVEL=ADDRSPACE, the address space associated with the member will be terminated and every XCF group member associated with that space (which could include other groups) will be terminated.
- For TERMLEVEL=SYSTEM, the system on which the target member resides will be removed from the sysplex.

The system that performs the termination is always the system on which the target member resides. If termination of the member is to be performed by a system that is still being initialized (running during NIP), the system will be removed from the sysplex.

The abnormal end might cause other members who were associated with the same task, address space, or system as the target member to terminate. XCF does not allow the recovery routines for that task to retry. Use the IXCTERM macro to remove from the XCF group another active member of the same group. The actual termination of the target member occurs asynchronously.

After the target member terminates, XCF notifies the remaining members in the group that have defined a group user-routine that the member is no longer associated with the group. The target member's recovery routine determines the new state of the target member: quiesced, failed, or not-defined.

### Environment

The following are the environment requirements for the caller.

Environment	Environment requirement
Minimum authorization:	Supervisor state or PKM allowing key 0-7
Dispatchable unit mode:	Task
Cross memory mode:	Any PASN, any HASN, any SASN. The primary address space must be the same as the primary address space of the caller of the IXCJOIN that placed the calling member in the active state.
AMODE:	31-bit
ASC mode:	Primary or access register (AR)
Interrupt status:	Enabled for I/O and external interrupts

Environment	Environment requirement
Locks:	No locks held
Control parameters:	Must be in the primary address space or be in an address/data space that is addressable through a public entry on the caller's dispatchable unit access list (DU-AL)

## Programming Requirements

If the program is in AR mode, issue the SYSSTATE ASCENV=AR macro before IXCTERM. SYSSTATE ASCENV=AR tells the system to generate code appropriate for AR mode.

## Restrictions

- The caller and the target member must be active members of the same XCF group.
- The caller can have no enabled, unlocked task (EUT) FRRs established.
- The target member cannot be address space associated.

## Input Register Information

Before issuing the IXCTERM macro, the caller does not have to place any information into any register unless using it in register notation for a particular parameter, or using it as a base register.

## Output Register Information

When control returns to the caller, the general purpose registers (GPRs) contain:

### Register Contents

**0**

If GPR 15 contains a zero or X'10', GPR 0 is used as a work register by the system; otherwise, GPR 0 contains a reason code.

**1**

Used as a work register by the system.

**2-13**

Unchanged.

**14**

Used as a work register by the system.

**15**

Return code.

When control returns to the caller, the access registers (ARs) contain:

### Register Contents

**0-1**

Used as work registers by the system

**2-13**

Unchanged

**14-15**

Used as work registers by the system

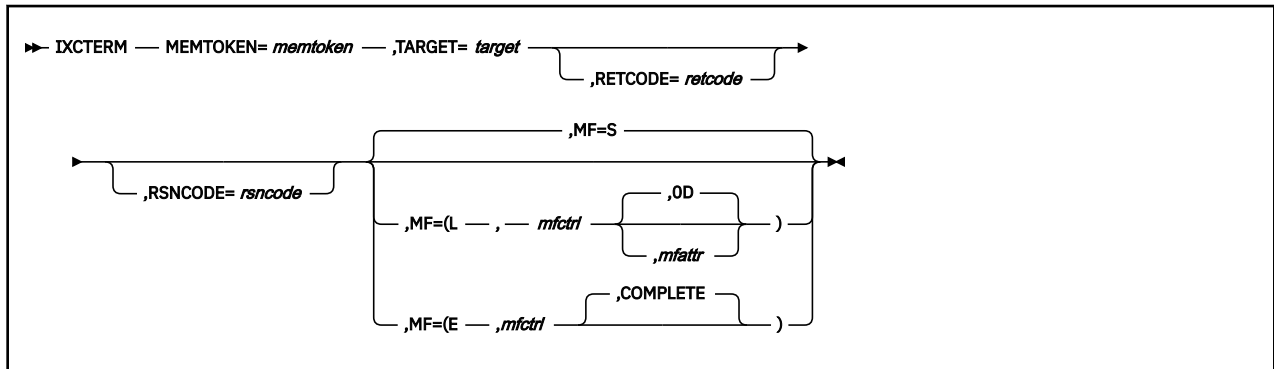
Some callers depend on register contents remaining the same before and after issuing a service. If the system changes the contents of registers on which the caller depends, the caller must save them before issuing the service, and restore them after the system returns control.

## Performance Implications

None.

## Syntax Diagram

The syntax of the IXCTERM macro is as follows:



## Parameter Descriptions

The parameter descriptions are listed in alphabetical order. Default values are underlined:

### MEMTOKEN=memtoken

Use this input parameter to specify the 64-bit token of the member issuing IXCTERM. IXCJOIN provided this token when it activated the member.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-byte field that contains the member token for the member issuing the IXCTERM.

### ,MF=S

### ,MF=(L,mfctrl)

### ,MF=(L,mfctrl,mfattr)

### ,MF=(L,mfctrl,OD)

### ,MF=(M,mfctrl)

### ,MF=(M,mfctrl,COMPLETE)

### ,MF=(M,mfctrl,NOCHECK)

### ,MF=(E,mfctrl)

### ,MF=(E,mfctrl,COMPLETE)

### ,MF=(E,mfctrl,NOCHECK)

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

### ,mfctrl

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

**,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

**,COMPLETE****,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

**COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if *SMILE=var* were an optional parameter and the default is *SMILE=NO\_SMILE* then it would not be documented. However, if the default was *SMILE=:)*, then it would be documented because a value would be the default.

**NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

**,RETCODE=retcode**

Use this output parameter to specify the location in which the system is to copy the return code from GPR 15.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 4-byte field that will contain the return code when the request is completed.

**,RSNCODE=rsncode**

Use this output parameter to specify the location in which the system is to copy the reason code from GPR 0.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 4-byte field that will contain the reason code (if any) when the request is completed.

**,TARGET=target**

Use this input parameter to specify the 64-bit token of the active member that XCF is to terminate. IXCJOIN provided this token when it activated the member.

**Note:** The target member must be a member of the same XCF group as the issuer of the IXCTERM. The target member cannot be an address space associated member.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-byte field that contains the member token of the member XCF is to terminate.

## ABEND Codes

---

None.

## Return and Reason Codes

---

When the IXCTERM macro returns control to your program:

- GPR 15 (and *retcode*, if you coded RETCODE) contains a return code.
- GPR 0 (and *rsncode*, if you coded RSNCODE) contains a reason code, if applicable.

Macro IXCYCON provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:



**0** IXCRETCODEOK  
**4** IXCRETCODEWARNING  
**8** IXCRETCODEPARMERROR  
**C** IXCRETCODEENVERROR  
**10** IXCRETCODECOMPERROR

The following table identifies the hexadecimal return and reason codes and the equate symbol associated with each reason code.

Table 23. Return and Reason Codes for the IXCTERM Macro		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
00	None.	<b>Meaning:</b> IXCTERM completed successfully; XCF will terminate the member. <b>Action:</b> None.
08	04	<b>Equate Symbol:</b> IXCTERMRSNNOTACTIVE <b>Meaning:</b> Program error. The calling member token does not identify an active member. <b>Action:</b> Take one or more of the following actions: <ul style="list-style-type: none"> <li>• Ensure that the proper MEMTOKEN address and ALET (if appropriate) were used.</li> <li>• Any additional action depends on your application. Because the member (MEMTOKEN) is no longer active, your program might want to stop any further activity associated with the member.</li> </ul>
08	08	<b>Equate Symbol:</b> IXCTERMRSNINAPPROPRIATEPRIMARY <b>Meaning:</b> Program error. The primary address space is not the same as the primary address space of the caller of the IXCJOIN that placed the calling member in the active state. <b>Action:</b> Take one or more of the following actions: <ul style="list-style-type: none"> <li>• Ensure that the correct MEMTOKEN address and ALET (if appropriate) were used.</li> <li>• Change your program to issue IXCTERM from the address space in which the calling member joined the group and retry the request.</li> </ul>

Table 23. Return and Reason Codes for the IXCTERM Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
08	0C	<p><b>Equate Symbol:</b> IXCTERMRSNTARGETNOTACTIVE</p> <p><b>Meaning:</b> Program error. The target member is not an active member.</p> <p><b>Action:</b> If the member is expected to be in an active state, ensure that the TARGET MEMTOKEN is correct, and retry the request.</p> <p>If the target member is required to be in a not-defined state, you can use the IXCQUERY service to determine if the member is in a created, quiesced, or failed state. You can use the IXCDELET service to place the member in a not-defined state.</p>
08	10	<p><b>Equate Symbol:</b> IXCTERMRSNTARGETNOTDEFINED</p> <p><b>Meaning:</b> Program error. The target member is not defined to XCF.</p> <p><b>Action:</b> If the TARGET member is expected to be in an active, created, quiesced, or failed state, ensure that the correct TARGET MEMTOKEN was specified. Otherwise, no action is required because the target member has terminated.</p>
08	14	<p><b>Equate Symbol:</b> IXCTERMRSNTARGETDIFFERENTGROUP</p> <p><b>Meaning:</b> Program error. The target member and the issuing member are in different XCF groups.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• Ensure that the correct TARGET and MEMTOKEN member tokens were specified. Check the addresses and ALETs (if appropriate).</li> <li>• You might have to change your program to ensure that it issues IXCTERM only with TARGET and caller (MEMTOKEN) members that are in the same group.</li> </ul>
08	18	<p><b>Equate Symbol:</b> IXCTERMRSNTARGETNOTVALID</p> <p><b>Meaning:</b> Program error. The target member token is not valid.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• Ensure that the correct TARGET member token address and ALET (if appropriate) were used.</li> <li>• If your program is running in AR mode, ensure that the SYSSTATE ASCENV=AR macro was issued before the IXCTERM.</li> </ul>
08	1C	<p><b>Equate Symbol:</b> IXCTERMRSNMEMTOKENNOTVALID</p> <p><b>Meaning:</b> Program error. The token of the issuing member is not valid.</p> <p><b>Action:</b> Correct the member token and retry the request.</p>

Table 23. Return and Reason Codes for the IXCTERM Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
08	40	<p><b>Equate Symbol:</b> IXCTERMRSNPLISTRSDNOTVALID</p> <p><b>Meaning:</b> Program error or environmental error. A reserved field in the control parameter list is not zero.</p> <p><b>Action:</b> Check to see if your program inadvertently overlaid the parameter list storage, and that it was assembled with the correct macro library for the release of MVS your program is running on.</p>
08	100	<p><b>Equate Symbol:</b> IXCTERMRSNPLISTBADALET</p> <p><b>Meaning:</b> Program error. Your program is running in AR mode, and the ALET that qualifies the address of the control parameter list is neither zero nor associated with a valid entry on the caller's DU-AL.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• You specified SYSSTATE ASCENV=AR before issuing the IXCJOIN macro.</li> <li>• The ALET for the parameter list is on the DU-AL or is zero (primary address space ALET).</li> </ul>
08	104	<p><b>Equate Symbol:</b> IXCTERMRSNPLISTVERSIONNOTVALID</p> <p><b>Meaning:</b> Program error. The version number in the control parameter list is not valid.</p> <p><b>Action:</b> Check to see if your program inadvertently overlaid the parameter list storage.</p>
08	108	<p><b>Equate Symbol:</b> IXCTERMRSNPLISTBADFUNCTION</p> <p><b>Meaning:</b> Program error. The function code in the control parameter list is not valid.</p> <p><b>Action:</b> Check to see if your program inadvertently overlaid the parameter list storage, and that it was assembled with the correct macro library for the release of MVS your program is running on.</p>
08	10C	<p><b>Equate Symbol:</b> IXCTERMRSNPLISTBADSTG</p> <p><b>Meaning:</b> Program error or environmental error. XCF could not access the control parameter list.</p> <p><b>Action:</b> Take one or more of the following actions:</p> <ul style="list-style-type: none"> <li>• Ensure that the correct parameter list storage area was specified.</li> <li>• If your program is running in AR mode: <ul style="list-style-type: none"> <li>– Ensure that you specified SYSSTATE ASCENV=AR before issuing the IXCTERM macro.</li> <li>– Ensure that the parameter list ALET is correct.</li> </ul> </li> <li>• Ensure that the parameter list storage area was not inadvertently freed by your program.</li> </ul>

Table 23. Return and Reason Codes for the IXCTERM Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
08	118	<b>Equate Symbol:</b> IXCTERMRSNNOTTASKMODE <b>Meaning:</b> Program error. The caller is not in task mode. <b>Action:</b> Correct your program so that it issues IXCTERM only while in task mode.
08	11C	<b>Equate Symbol:</b> IXCTERMRSNNOTENABLED <b>Meaning:</b> Program error. The caller is not enabled. <b>Action:</b> Correct your program so that it issues IXCTERM only while enabled.
08	120	<b>Equate Symbol:</b> IXCTERMRSNTARGETNOTMEMASSOCTASK <b>Meaning:</b> Program error. The target member cannot be terminated because it is associated with an address space instead of a task. <b>Action:</b> Change your program to associate the member with a task, or cancel the member's associated address space.
10	None.	<b>Meaning:</b> System error. XCF processing failed. <b>Action:</b> Retry the request at least once. If the problem persists, record the return code, and supply it to the appropriate IBM support personnel.

## Example

*Operation:* Terminate a member from a group. The token of the caller is TOKEN1; the token of the target member is TOKEN2. XCF is to store the return code and reason code into the variables RETURN and REASON. The code is as follows:

```

IXCTERM  MEMTOKEN=TOKEN1,TARGET=TARGET2,          X
         RETCODE=RETURN,RSNCODE=REASON,MF=S

TOKEN1   DS   CL8                                TOKEN OF MEMBER ISSUING IXCTERM
TOKEN2   DS   CL8                                TOKEN OF MEMBER TO BE          X
                                                TERMINATED
RETURN   DS   1F                                RETURN CODE
REASON   DS   1F                                REASON CODE

```

You can access the member tokens from the QUAMTOKN field in the area returned by IXCJOIN and IXCQUERY, and mapped by the IXCYQUAA mapping macro.

## Chapter 27. IXLADUPX – Synchronize an asynchronously duplexed structure

### Description

The IXLADUPX service routine is given control from the IXLADUPX macro. The IXLADUPX service allows a connected user to ensure that a request that was previously committed in the primary instance of an asynchronously duplexed structure is also committed in the secondary instance of the structure.

### Environment

The following are the environment requirements for the caller.

Environment	Environment requirement
Authorization:	Supervisor state or PKM keys 0 - 7
Dispatchable unit mode:	Task or SRB mode
Cross memory mode:	Any PASN, any HASN, any SASN. The current primary address space must be the same as the primary address space at the time the connection service, IXLCONN, was issued for the structure.
AMODE:	31 or 64-bit
ASC mode:	Primary or access register (AR). If in Access Register ASC mode, specify SYSSTATE ASCENV=AR before invoking this macro.
Interrupt status:	Enabled or disabled for I/O and external interrupts. When OPTYPE=SUSPEND is specified the caller must be enabled.
Locks:	Disabled callers must be legally disabled by holding the CPU lock and cannot hold other disabled locks (invocation by DIE routines is not supported). Enabled callers must not hold any locks.
Control parameters:	Control parameters must be in the primary address space or, for AR-mode callers, must be in an address/data space that is addressable through a public entry on the caller's dispatchable unit access list (DU-AL).  If the caller is running in Access Register ASC mode and specifies a macro parameter by using explicit register notation, the access register corresponding to the general register must appropriately qualify the general register.

### Programming requirements

If the program is in Access Register ASC mode, specify SYSSTATE ASCENV=AR before invoking this macro.  
If the program is in 64-bit mode, specify SYSSTATE AMODE64=YES before invoking the macro.

## Restrictions

---

- IXLADUPX requires system-managed asynchronous duplexing support. Use of IXLADUPX on a system that does not support it will have unpredictable results. Macro IXCYQUAA defines the QuReqRfAsyncDuplex bit in the QuReqFeatures string that can be used to test for system-managed asynchronous duplexing support. Use IXCQUERY REQINFO=FEATURES to get the QuReqFeatures string.
- If the caller is disabled, then the parameter list and all storage areas that are addressed by macro parameters must reside in either fixed or disabled reference storage.
- The caller cannot be running as a Disabled Interrupt Exit (DIE).
- Callers that are running in SRB mode should refrain from specifying OPTYPE=SUSPEND under the following circumstances:
  - After the SRB receives an X'47B' abend.
  - When running in a suspend exit after invoking SUSPEND.

## Input register information

---

Before issuing the IXLADUPX macro, the caller does not have to place any information into any register unless the caller is using it in register notation for a particular parameter, or using it as a base register.

## Output register information

---

When control returns to the caller of the IXLADUPX macro, the general purpose registers (GPRs) contain:

### GPR

#### Contents

**0**

Reason code, if applicable,

**1**

Used as work register by the system

**2-13**

Unchanged

**14**

Used as work register by the system

**15**

Return code

When control returns to the caller of the IXLADUPX macro, the access registers (ARs) contain:

### AR

#### Contents

**0-1**

Used as work registers by the system

**2-13**

Unchanged

**14-15**

Used as work registers by the system

Some callers depend on register contents remaining the same before and after issuing a service. If the system changes the contents of registers on which the caller depends, the caller must save them before issuing the service and restore them after the system returns control.

## Performance implications

---

None.

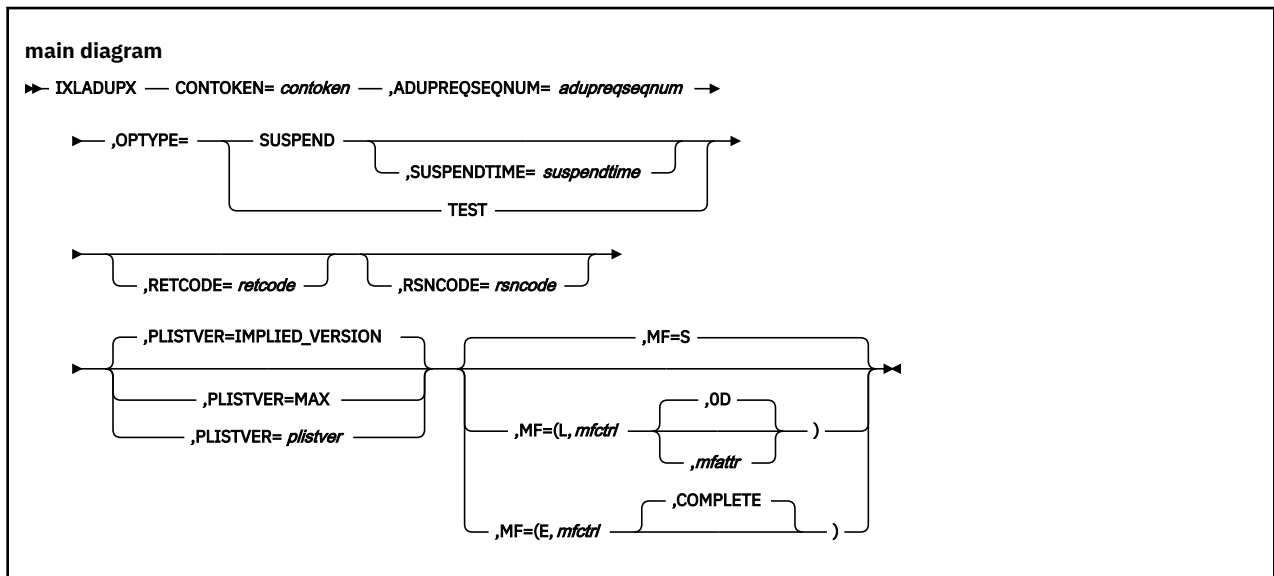
## Understanding IXLADUPX version support

The IXLADUPX macro supports version 0 keywords and functions.

Specify the version of the parameter list that you want generated with the PLISTVER keyword. See Chapter 2, “Specifying a Macro Version Number,” on page 5 for considerations when specifying the version of the parameter list with PLISTVER.

## Syntax diagram

The syntax of IXLADUPX is written as follows:



## Parameter descriptions

The parameter descriptions are listed in alphabetical order. Default values are underlined:

### **ADUPREQSEQNUM=***adupreqseqnum*

Use this input parameter to specify the asynchronous duplexing request sequence number returned on a previous invocation of IXLLOCK, IXLRT, or IXLSYNCH service.

The asynchronous duplexing request sequence number identifies a prior request that was committed in the primary instance of an asynchronously duplexed structure.

Requests with asynchronous duplexing request sequence numbers assigned are committed in the secondary structure in the ascending order of these sequence numbers. If the request identified by ADUPREQSEQNUM is committed in the secondary structure, all requests with a lower asynchronous duplexing request sequence number that are made by the same connector are also committed in the secondary structure.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-character field that contains the asynchronous duplexing request sequence number.

### **,CONTOKEN=***contoken*

Use this input parameter to specify the connect token that is returned by the IXLCONN service.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-character field that contains the connect token.

**,MF=S**  
**,MF=(L,mfctrl)**  
**,MF=(L,mfctrl,mfattr)**  
**,MF=(L,mfctrl,OD)**  
**,MF=(E,mfctrl)**  
**,MF=(E,mfctrl,COMPLETE)**

Use MF=S to specify the standard form of the macro, which builds an in-line parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

**,mfctrl**

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the parameter list.

**,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembler DS pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of OD, which forces the parameter list to a doubleword boundary.

**,COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if SMILE=var were an optional parameter and the default is SMILE=NO\_SMILE then it would not be documented. However, if the default was SMILE=:-), then it would be documented because a value would be the default.

**,OPTYPE=SUSPEND**

**,OPTYPE=TEST**

Use OPTYPE=SUSPEND to indicate that control should be returned to the invoker when the request identified by ADUPREQSEQNUM has been committed in the secondary structure. The invoker must be executing in an enabled state to use this option.

Use OPTYPE=TEST to test the processing status of the request identified by ADUPREQSEQNUM. The return and reason code indicate whether the request has been committed in the secondary structure. This option can be used by invokers that cannot be suspended to poll for request processing status.

**,PLISTVER=IMPLIED\_VERSION**

**,PLISTVER=MAX**

**,PLISTVER=plistver**

Use this input parameter to specify the version of the macro. See [“Understanding IXLADUPX version support” on page 431](#) for a description of the options available with PLISTVER.

**,RETCODE=retcode**

Use this output parameter to specify the location in which the system is to copy the return code from GPR 15.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword to contain the return code.



**,RSNCODE=rsncode**

Use this output parameter to specify the location in which the system is to copy the reason code from GPR 0.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword to contain the reason code.

**,SUSPENDTIME=suspendtime**

Use this output parameter with OPTYPE=SUSPEND to specify a storage area to contain the amount of time, in microseconds, the invoker is suspended waiting for the request to be committed in the secondary structure. If the IXLADUPX request completes successfully without suspending the invoker, the content of this output variable will be zero.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-character field where the system will put the suspend time.

## Return and reason codes

When the IXLADUPX macro returns control to your program:

- GPR 15 (and *retcode*, if you coded RETCODE) contains a return code.
- GPR 0 (and *rsncode*, if you coded RSNCODE) contains a reason code, if applicable.

Macro IXLYCON provides equate symbols for the return and reason codes. The equate symbols that are associated with each hexadecimal return code are as follows:

**0**

IXLRETCODEOK

**4**

IXLRETCODEWARNING

**8**

IXLRETCODEPARMERROR

**C**

IXLRETCODEENVERROR

**10**

IXLRETCODECOMPERROR

Table 24 on page 433 identifies the hexadecimal return and reason codes and the equate symbol that is associated with each reason code. xxxx indicates internal information.

Table 24. Return and reason codes for the IXLADUPX macro		
Hexadecimal return code	Hexadecimal reason code	Equate symbol meaning and action
0	None.	<b>Meaning:</b> Successful completion. The request that was identified by ADUPREQSEQNUM was committed in both the primary and the secondary structures.
4	xxxx0432	<b>Equate Symbol:</b> IXLRSNCODENOTINSECONDARY <b>Meaning:</b> A OPTYPE=TEST operation found that the request that was identified by ADUPREQSEQNUM was not committed in the secondary structure. <b>Action:</b> Issue the IXLADUPX macro again with OPTYPE=TEST to determine whether the request was committed in the secondary structure. Alternatively, issue the IXLADUPX macro with OPTYPE=SUSPEND in an enabled environment to wait for the request to be committed in the secondary structure.

Table 24. Return and reason codes for the IXLADUPX macro (continued)		
Hexadecimal return code	Hexadecimal reason code	Equate symbol meaning and action
4	xxxx0433	<p><b>Equate Symbol:</b> IXLRSNCODESTRNOTASYNCDUPLEX</p> <p><b>Meaning:</b> The structure that the user connects to is not asynchronously duplexed.</p> <p><b>Action:</b> None required. However, you might want to ensure that the asynchronous duplexing request sequence number is correct.</p>
4	xxxx0434	<p><b>Equate Symbol:</b> IXLRSNCODEBADDUPLEXINSTANCE</p> <p><b>Meaning:</b> The request that was identified by ADUPREQSEQNUM is not associated with the current instance of the asynchronously duplexed structure pair.</p> <p><b>Action:</b> None required. However, you might want to ensure that the asynchronous duplexing request sequence number is correct.</p>
8	xxxx0801	<p><b>Equate Symbol:</b> IXLRSNCODEBADPARMLIST</p> <p><b>Meaning:</b> Program error. The parameter list for this request is not accessible.</p> <p><b>Action:</b> Verify that:</p> <ul style="list-style-type: none"> <li>• The parameter list address is uncorrupted.</li> <li>• The parameter list is addressable in the caller's primary address space or, for AR-mode callers, in an address or data space that is addressable through a public entry on the caller's dispatchable unit access list (DU-AL).</li> <li>• If you are issuing this macro while disabled, the parameter list resides in either page-fixed or DREF storage.</li> <li>• If you are issuing this macro in AR-mode and you specified the parameter list address using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If you are issuing this macro in AR-mode, SYSSTATE ASCENV=AR must be issued before issuing this macro.</li> </ul>
8	xxxx0802	<p><b>Equate Symbol:</b> IXLRSNCODEBADPARMLISTALET</p> <p><b>Meaning:</b> Program error. The parameter list ALET is not valid.</p> <p><b>Action:</b> Ensure that the ALET is zero or that the ALET represents a valid entry on the DU-AL.</p>
8	xxxx0804	<p><b>Equate Symbol:</b> IXLRSNCODEBADVERSION#</p> <p><b>Meaning:</b> Program error. Invalid version number in the IXLCONN parameter list.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that your program did not overlay the parameter list storage.</li> <li>• Verify that your program was assembled with the correct macro library for the release of MVS that your program is running on.</li> </ul>

Table 24. Return and reason codes for the IXLADUPX macro (continued)		
Hexadecimal return code	Hexadecimal reason code	Equate symbol meaning and action
8	xxxx080A	<p><b>Equate Symbol:</b> IXLRSNCODEBADCONTOKEN</p> <p><b>Meaning:</b> Program error. The input contoken is not valid. The contoken might no longer be valid for one of the following reasons:</p> <ul style="list-style-type: none"> <li>• A disconnect occurred.</li> <li>• The connector's task ended.</li> <li>• The input contoken is not the contoken that was returned from IXLCONN.</li> <li>• The request was issued outside the connector's address space.</li> <li>• The contoken was invalidated for rebuild.</li> </ul> <p><b>Action:</b> Verify that the CONTOKEN value that was specified is valid and for the correct structure.</p>
8	xxxx0818	<p><b>Equate Symbol:</b> IXLRSNCODENOTLOCKSTR</p> <p><b>Meaning:</b> Program error. The CONTOKEN that was specified does not represent a lock structure.</p> <p><b>Action:</b> Verify that the CONTOKEN value that was specified is valid and is for the intended structure.</p>
8	xxxx0851	<p><b>Equate Symbol:</b> IXLRSNCODENOSUSPENDISABLE</p> <p><b>Meaning:</b> Program error. The request failed because OPTYPE=SUSPEND was specified, but the caller is disabled and cannot be suspended.</p> <p><b>Action:</b> Either specify another OPTYPE value, or become enabled (release the CPU lock), then reissue the request.</p>
8	xxxx08A1	<p><b>Equate Symbol:</b> IXLRSNCODEBADADUPREQSEQNUM</p> <p><b>Meaning:</b> Program error. An invalid asynchronous duplexing request sequence number was specified. The value that is specified in the ADUPREQSEQNUM does not represent a valid asynchronous duplexing request sequence number that was returned on a previous IXLLOCK, IXLRT, or IXLSYNCH invocation.</p> <p><b>Action:</b> Ensure that you use only the asynchronous duplexing request sequence number that is returned on a previous IXLLOCK, IXLRT, or IXLSYNCH invocation.</p>
8	xxxx08B6	<p><b>Equate Symbol:</b> IXLRSNCODEBADXSUSPENDENV</p> <p><b>Meaning:</b> Program error. The IXLADUPX OPTYPE=SUSPEND operation was issued from a SUSPEND exit routine or from an SRB routine that the system abended with a 47B system completion code. The caller cannot be suspended while running in this environment.</p> <p><b>Action:</b> Either specify another OPTYPE value, or reissue the IXLADUPX macro from an environment that allows the caller to be suspended.</p>

Table 24. Return and reason codes for the IXLADUPX macro (continued)		
Hexadecimal return code	Hexadecimal reason code	Equate symbol meaning and action
8	xxxx08B7	<p><b>Equate Symbol:</b> IXLRSNCODEBADLOCKS</p> <p><b>Meaning:</b> Program error. The caller's environment does not match the serialization and interrupt status requirements of the IXLADUPX service routine. For example:</p> <ul style="list-style-type: none"> <li>• Caller is enabled but holds locks.</li> <li>• Caller is disabled but does not hold the CPU lock.</li> <li>• Caller is disabled but holds disabled locks in addition to the CPU lock.</li> </ul> <p><b>Action:</b> Reissue the IXLADUPX macro in an environment that matches the requirements of the service macro.</p>
C	xxxx0C06	<p><b>Equate Symbol:</b> IXLRSNCODENOCONN</p> <p><b>Meaning:</b> Environmental error. This system does not have connectivity to the coupling facility that contains the structure. Possible reasons for this are:</p> <ul style="list-style-type: none"> <li>• The operator issued VARY PATH,OFFLINE.</li> <li>• The operator issued CONFIG CHP,OFFLINE.</li> <li>• Hardware errors to the coupling facility.</li> <li>• Facility or path failure to the coupling facility.</li> </ul> <p><b>Action:</b> Begin rebuilding the structure on a different coupling facility, or disconnect from the structure.</p>
C	xxxx00C25	<p><b>Equate Symbol:</b> IXLRSNCODESTRFAILURE</p> <p><b>Meaning:</b> Environmental error. The structure failed before the request was completed.</p> <p><b>Action:</b> Either rebuild or disconnect from the structure.</p>
10	xxxx10xx	<p><b>Meaning:</b> XES processing failure. The state of the involved structure and the disposition of the request are unpredictable.</p> <p><b>Action:</b> Contact the IBM support center.</p>

## Chapter 28. IXLALTER — Alter a Coupling Facility Structure

### Description

The IXLALTER service allows an authorized program, not necessarily a connected user of a structure, to start or stop the structure alter process. Structure alter is a non-disruptive process to expand or contract the size of a structure, to reapportion the structure's entry-to-element ratio, and to change the percentage of structure storage that is set aside for event monitor controls (EMCs).

The structure to be altered must be allocated in a coupling facility that supports structure alter, that is, a coupling facility with CFLEVEL=1 or higher. For keyed list structures containing EMCs in a coupling facility with CFLEVEL=3, you can alter the size or the entry-to-element ratio of the structure, but the EMC storage percentage will not change. For keyed list structures containing EMCs in a coupling facility with CFLEVEL=4 or higher, you can also alter the EMC storage percentage.

A structure alter can be done only if all connectors to the structure allow structure alter (indicated by specifying ALLOWALTER=YES on IXLCONN). At connect time, connectors can also specify whether changing the entry-to-element ratio and EMC storage percentage is permitted (RATIO=YES or NO). If a change is permitted, the connector can also indicate the minimum threshold levels for entries, elements, and EMC storage (MINENTRY, MINELEMENT, MINEMC). These minimum threshold levels indicate the percent of entries, elements, and EMC storage that must be available when the structure alter completes.

- MINENTRY and MINELEMENT

For list structures, this is the percent of “in-use” entries and/or elements; for cache structures, this is the percent of “in-use and changed” entries and/or elements.

- MINEMC

For keyed list structures in a coupling facility with CFLEVEL=4 or higher, this is the percent of “currently-in-use” EMCs.

A structure alter request will not be accepted for:

- A structure that is already in the rebuild or alter process.
- A structure that is in a user-managed or system-managed duplexing rebuild but not in the Duplex Established phase.
- A structure that has structure objects in storage-class memory or has augmented space other than fixed augmented space in use.

If a structure is in user-managed duplexing rebuild processing, the alter is automatically applied to both structure instances serially. The primary (old) structure is altered first, with corresponding Alter Begin and Alter End events. After the Alter End is delivered for the primary structure, alter begins for the secondary (new) structure, also with its corresponding Alter Begin and Alter End events. The alter events indicate which of the structures is currently being altered. The system will not allow you to start a subsequent alter request for the structure until both instances of the structure have completed the current alter process.

If a structure is in system-managed duplexing rebuild processing, the alter is applied to both structure instances serially, with the primary (old) structure being altered first followed by the alter of the secondary (new) structure. However, only one set of Alter Begin and Alter End events are delivered. The Alter Begin event is delivered at the start of alter processing for the primary structure; the Alter End event is delivered at the end of alter processing for both instances of the structure.

Alter processing ends when:

- The target size, ratio, or EMC storage percentage is satisfied, either completely or to the extent possible based on available coupling facility resources. (When no more progress can be made toward attaining requested targets, IXLALTER processing ends.)
- A stop alter request is received (either a SETXCF command or an IXLALTER request to stop the alter). This does not apply to structures in the Duplex Established phase of a system-managed duplexing rebuild. While an alter of a structure in the Duplex Established phase of system-managed duplexing rebuild is in progress, the system will reject an attempt to stop the alter with either message IXC531I (for SETXCF) or return code IXLRNCODEDUPALTER (for IXLALTER).
- A structure or coupling facility failure occurs. If the SP 5.2 or higher systems in the sysplex all fail or lose connectivity to the coupling facility in which the structure resides, alter processing ends when the first SP 5.2 or higher system regains connectivity to the coupling facility.
- A structure rebuild request is received, except when the alter processing is for a structure in the Duplex Established phase of user-managed or system-managed duplexing rebuild. In that case, the system does not accept the structure rebuild request, and the alter processing continues.
- A request to stop duplexing rebuild is received while the structure is being altered in the Duplex Established phase. Alter processing depends on which structure instance is to be kept and which is being deallocated, and on the amount of alter processing that has been performed or remains to be performed. If the alter of the duplexed structure has already completely processed the structure that is to be kept in simplex mode, then the alter processing is finished and the system issues the Alter End event. If there is still alter processing either in progress against, or not yet performed against, the structure that is to be kept, then the alter processing continues against the simplex structure after duplexing is switched or stopped. When that alter processing against the simplex structure completes, the system issues the Alter End event.
- The coupling facility migrates structure objects into storage-class memory.

When the alter processing is complete, all active connectors are notified of the completion through their event exit.

The security administrator may be using RACF or another security product to restrict the use of installation coupling facility structures. If so, ensure that you are authorized to issue the IXLALTER macro for the structure.

## Environment

The following are the environment requirements for the caller.

Environment	Environment requirement
Authorization:	Supervisor state or PKM allowing key 0 - 7
Dispatchable unit mode:	Task
Cross memory mode:	PASN=HASN, any SASN
AMODE:	31-bit
ASC mode:	Primary or access register (AR)
Interrupt status:	Enabled for I/O and external interrupts
Locks:	No locks held, with no enabled, unlocked task (EUT) FRRs established
Control parameters:	Must be in the primary address space or be in an address/data space that is addressable through a public entry on the caller's dispatchable unit access list (DU-AL)

## Input Register Information

---

Before issuing the IXLALTER macro, the caller does not have to place any information into any register unless using it in register notation for a particular parameter, or using it as a base register.

## Output Register Information

---

When control returns to the caller of the IXLALTER macro, the general purpose registers (GPRs) contain:

### Register Contents

- 0**  
Reason code, if GPR 15 contains a non-zero return code
- 1**  
Used as work register by the system
- 2-13**  
Unchanged
- 14**  
Used as work register by the system
- 15**  
Return code

When control returns to the caller of the IXLALTER macro, the access registers (ARs) contain:

### Register Contents

- 0-1**  
Used as work registers by the system
- 2-13**  
Unchanged
- 14-15**  
Used as work registers by the system

**For registers that the system changes,** a caller who depends on these registers containing the same value before and after issuing the macro must save these registers before issuing the macro and restore them after the system returns control.

## Programming Requirements

---

If the program is in AR mode, issue the SYSSTATE ASCENV=AR macro before IXLALTER. SYSSTATE ASCENV=AR tells the system to generate code appropriate for AR mode.

Include the IXLYCON macro in your program. This macro provides a list of equate symbols for users of XES services and exits.

## Performance Implications

---

IXLALTER processing might involve referencing or updating the CFRM active policy to reflect the operation that has been requested. When invoking this macro, be aware of the I/O to the CFRM couple data set that might be generated.

## Understanding IXLALTER Version Support

---

The IXLALTER macro supports versions in the range of 0 - 1.

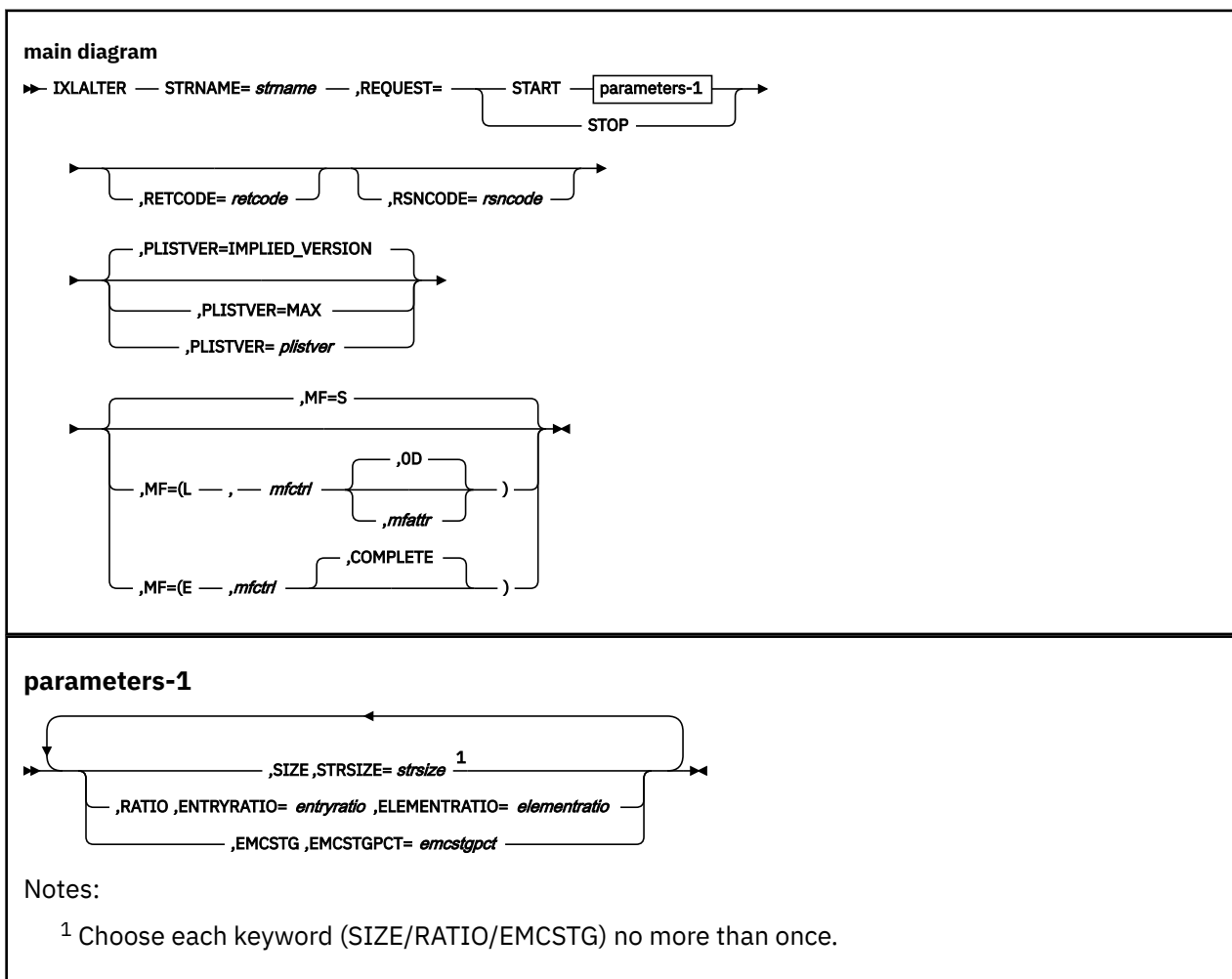
- Keywords not specifically noted here are supported by all versions starting with version 0 and higher of the IXLALTER macro.

- The following keywords are supported by all versions starting with version 1 and higher of the IXLALTER macro.
  - EMCSTG
  - EMCSTGPCT

Specify the version of the parameter list that you want generated with the PLISTVER keyword. See Chapter 2, “Specifying a Macro Version Number,” on page 5 for considerations when specifying the version of the parameter list with PLISTVER. Note that no special version of IXLALTER is required when altering a structure in the Duplex Established phase of structure duplexing.

## Syntax Diagram

The syntax of IXLALTER is written as follows:



## Parameter Descriptions

The parameter descriptions are listed in alphabetical order. Default values are underlined:

### **,ELEMENTRATIO=elementratio**

Use this input parameter to specify the data element part of the entry-to-element ratio.

- A cache structure initially allocated with data elements can be altered to contain no data elements and then changed again to contain data elements.
- A list structure initially allocated with data elements cannot be altered to contain no data elements.



- Cache or list structures initially allocated without data elements cannot be altered to contain data elements.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the halfword field that contains the element part of the ratio.

#### **,EMCSTG**

Use this input parameter to indicate that the percentage of available storage to be set aside for event monitor controls (EMCs) is to be changed.

When EMCSTG is not specified, it is possible to have the EMC storage percent change as a result of a SIZE request unless at least one connector specified `RATIO=NO` at connect time.

This parameter is valid only for keyed list structures allocated in a coupling facility with `CFLEVEL=4` or higher.

#### **,EMCSTGPCT=*emcstgpct***

Use this input parameter to specify the percentage of available list structure storage that is to be set aside for event monitor controls (EMCs).

Specify the percentage value in hundredths of a percent. For example, to request a value of 20%, code `EMCSTGPCT=2000`.

The value of EMCSTGPCT must be in the range of from 0 to 10000.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the halfword field that contains a value to express the percentage of available structure storage that is to be set aside for EMCs.

#### **,ENTRYRATIO=*entryratio***

Use this input parameter to specify the entry part of the entry-to-element ratio.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the halfword input field that contains the entry part of the ratio.

#### **,MF=S**

**,MF=(L,*mfctrl*)**

**,MF=(L,*mfctrl*,*mfattr*)**

**,MF=(L,*mfctrl*,*OD*)**

**,MF=(M,*mfctrl*)**

**,MF=(M,*mfctrl*,*COMPLETE*)**

**,MF=(M,*mfctrl*,*NOCHECK*)**

**,MF=(E,*mfctrl*)**

**,MF=(E,*mfctrl*,*COMPLETE*)**

**,MF=(E,*mfctrl*,*NOCHECK*)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the `PLISTVER` parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

#### **,*mfctrl***

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

**,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

**,COMPLETE****,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

**COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if *SMILE=var* were an optional parameter and the default is *SMILE=NO\_SMILE* then it would not be documented. However, if the default was *SMILE=-:-*, then it would be documented because a value would be the default.

**NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

**,PLISTVER=IMPLIED\_VERSION****,PLISTVER=MAX****,PLISTVER=plistver**

Use this input parameter to specify the version of the macro. See [“Understanding IXLALTER Version Support”](#) on page 439 for a description of the options available with *PLISTVER*.

**,RATIO**

Use this input parameter to indicate that the entry-to-element ratio of the named structure is to be reapportioned.

When *RATIO* is not specified, it is possible to have the entry-to-element ratio change as a result of a *SIZE* request unless at least one connector specified *RATIO=NO* at connect time.

**,REQUEST=START****,REQUEST=STOP**

Use this input parameter to identify the type of structure alter request.

**START**

Start structure alter processing for the specified structure.

**STOP**

Stop alter processing for the specified structure.

**,RETCODE=retcode**

Use this output parameter to specify the location in which the system is to copy the return code from GPR 15.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword to contain the return code.

**,RSNCODE=rsncode**

Use this output parameter to specify the location in which the system is to copy the reason code from GPR 0.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword to contain the reason code.

**,SIZE**

Use this input parameter to indicate that the structure is to be expanded or contracted.

**Note:** At the time of structure allocation, the maximum structure size (MSS) value is set and remains constant as long as this instance of the structure remains allocated. The structure allocation algorithm determines the MSS based on whether the structure is allocated by counts or by size/ratios. The actual structure size may be less than or equal to the MSS value. It may be less due to one or more of the following:

- INITSIZE value in the CFRM active policy
- STRSIZE specified on the IXLCONN macro
- Storage constraints in the coupling facility
- A previous structure alter.

The actual structure size may change with the MSS value being the upper bound.

The minimum structure size (MINSS) is the minimum control space required to allocate the structure with the attributes specified. The MINSS value is established by the coupling facility based on the attributes specified when the structure was allocated. The MINSS value can change when the structure is reapportioned with an entry-to-element ratio that differs substantially from the previous ratio. MINSIZE specified in the CFRM active policy will serve as a minimum bound for the structure size on all structure allocation requests (including alter, connect, and rebuild connect) except for new structure allocation during System-managed rebuild. New structure allocation during System-managed rebuild may cause the structure to be allocated with a size smaller than MINSIZE in the CFRM active policy. The actual structure size may expand to the MSS value or contract to the MINSS value or the MINSIZE value specified in the CFRM policy. When contracting, the larger of the two values will be used.

When SIZE is not specified the structure size will not change as a result of the IXLALTER.

#### **,STRNAME=***strname*

Use this input parameter to specify the name of the target structure for alter start or stop processing. The name must be defined in the CFRM (coupling facility resource management) active policy.

The structure name must be 16 characters long, padded on the right with blanks if necessary. It may contain numeric characters, uppercase alphabetic characters, national characters (\$, @, #), or an underscore (\_). It must begin with an uppercase alphabetic character. IBM names begin with SYS, an IBM component prefix, or letters A-I.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the 16-character input field that contains the structure name.

#### **,STRSIZE=***strsize*

Use this input parameter to specify the desired size of the structure in units of 4K (4096 bytes) blocks.

See *z/OS MVS Programming: Sysplex Services Guide* for information about minimum and maximum structure size.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the fullword input field that contains the number of 4K (4096 bytes) blocks to make up the structure.

## Return and reason codes

---

When the IXLALTER macro returns control to your program:

- GPR 15 (and *retcode*, if you coded RETCODE) contains a return code.
- When the value in GPR 15 is not zero, if applicable, GPR 0 (and *rsncode*, if you coded RSNCODE) contains a reason code.

The IXLYCON macro provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

**0**

IXLRETCODEOK

4

IXLRETCODEWARNING

8

IXLRETCODEPARMERROR

C

IXLRETCODEENVERROR

10

IXLRETCODECOMPERROR

The following table identifies the hexadecimal return and reason codes and the equate symbol associated with each reason code.

Table 25. Return and reason codes for the IXLALTER macro		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0	None.	<b>Meaning:</b> IXLALTER request accepted. <b>Action:</b> None.
4	xxxx0427	<b>Equate Symbol:</b> IXLRSNCODEALREADYALTERING <b>Meaning:</b> The request was rejected because alter is already in progress for the structure. A new alter request will not be accepted until the current alter completes or is stopped. <b>Action:</b> Either wait for the structure alter to complete or issue IXLALTER to stop the structure alter.
8	xxxx0801	<b>Equate Symbol:</b> IXLRSNCODEBADPARMLIST <b>Meaning:</b> Program error. The IXLALTER parameter list is not accessible. <b>Action:</b> Verify that: <ul style="list-style-type: none"> <li>• The parameter list address is uncorrupted</li> <li>• The parameter list is addressable in the caller's primary address space.</li> <li>• If you are calling IXLALTER in AR-mode and you specified the parameter list address using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If you are calling IXLALTER in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLALTER.</li> </ul>

Table 25. Return and reason codes for the IXLALTER macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0802	<p><b>Equate Symbol:</b> IXLRNCODEBADPARMLISTALET</p> <p><b>Meaning:</b> Program error. The IXLALTER parameter list ALET is not valid.</p> <p><b>Action:</b> Verify that:</p> <ul style="list-style-type: none"> <li>• If you issued IXLALTER in AR-mode and you specified the parameter list address using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If you are calling IXLALTER in AR-mode, SYSSTATE ASCENV=AR must be issued before issuing the IXLALTER.</li> <li>• If you are not running in AR-mode, and the parameter list does not need to be ALET-qualified, the corresponding access register should contain zeros.</li> </ul>
8	xxxx0804	<p><b>Equate Symbol:</b> IXLRNCODEBADVERSION#</p> <p><b>Meaning:</b> Program error. The IXLALTER parameter list version number is not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that your program did not overlay the parameter list storage.</li> <li>• Verify that your program was assembled with the correct macro library for the release of MVS on which your program is running.</li> </ul>
8	xxxx0806	<p><b>Equate Symbol:</b> IXLRNCODESRBMODE</p> <p><b>Meaning:</b> Program error. The requestor is in SRB mode.</p> <p><b>Action:</b> Change into task mode before issuing IXLALTER.</p>
8	xxxx0807	<p><b>Equate Symbol:</b> IXLRNCODENOTENABLED</p> <p><b>Meaning:</b> Program error. The requestor is not in an enabled state.</p> <p><b>Action:</b> The requestor must be enabled for I/O and external interrupts.</p>
8	xxxx0809	<p><b>Equate Symbol:</b> IXLRNCODEPRIMARYNOTHOME</p> <p><b>Meaning:</b> Program error. The requestor's primary address space is not the same as the home address space.</p> <p><b>Action:</b> The primary address space must equal the home address space to issue the IXLALTER.</p>
8	xxxx0824	<p><b>Equate Symbol:</b> IXLRNCODEWRONGSTRTYPE</p> <p><b>Meaning:</b> Program error. EMCSTGPCT cannot be altered on a cache or lock structure.</p> <p><b>Action:</b> EMCSTGPCT is meaningful only for a list structure.</p>

Table 25. Return and reason codes for the IXLALTER macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx084C	<b>Equate Symbol:</b> IXLRSNCODENOSAFAUTH <b>Meaning:</b> Program error. The requestor does not have proper SAF authorization. <b>Action:</b> Have the installation set up the proper SAF authorization for access to the structure.
8	xxxx0883	<b>Equate Symbol:</b> IXLRSNCODEBADEMSTGPCT <b>Meaning:</b> Program error. The value specified for EMCSTGPCT is not valid. The value must be between 0 and 10000. <b>Action:</b> Correct the value of EMCSTGPCT.
C	xxxx0C05	<b>Equate Symbol:</b> IXLRSNCODESTRNOTINPOLICY <b>Meaning:</b> Environmental error. The requested structure is not in the CFRM active policy. <b>Action:</b> Specify a structure that is currently defined in the CFRM active policy or switch to a new CFRM policy that contains the structure for which this IXLALTER was invoked.
C	xxxx0C0A	<b>Equate Symbol:</b> IXLRSNCODESTRNOTALLOCATED <b>Meaning:</b> Environmental error. The requested structure is not allocated. <b>Action:</b> Ensure that you have specified the correct structure name on the IXLALTER request. If so, allocate the structure by issuing the appropriate IXLCONN request.
C	xxxx0C25	<b>Equate Symbol:</b> IXLRSNCODESTRFAILURE <b>Meaning:</b> Environmental error. Structure failure has occurred. <b>Action:</b> None.
C	xxxx0C29	<b>Equate Symbol:</b> IXLRSNCODEXESNOTACTIVE <b>Meaning:</b> Environmental error. The CFRM function is not active or not available. <b>Action:</b> Bring the CFRM couple data set into use in the sysplex by issuing the SETXCF COUPLE,TYPE=CFRM,PCOUPLE= command.
C	xxxx0C36	<b>Equate Symbol:</b> IXLRSNCODESTOPINPROGRESS <b>Meaning:</b> Environmental error. Request rejected because structure rebuild stop was in progress for the structure. <b>Action:</b> Wait to initiate structure alter until the structure rebuild stop process completes.

Table 25. Return and reason codes for the IXLALTER macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C51	<p><b>Equate Symbol:</b> IXLRNCDEREBUILDINPROGRESS</p> <p><b>Meaning:</b> Environmental error. Request rejected because structure rebuild was in progress for the structure.</p> <p><b>Action:</b> Wait to initiate structure alter until the structure rebuild process completes or stop the rebuild process.</p>
C	xxxx0C60	<p><b>Equate Symbol:</b> IXLRNCDENOALTERCF</p> <p><b>Meaning:</b> Environmental error. Request rejected because the structure is allocated in a coupling facility that does not support alter. CFLEVEL is equal to zero.</p> <p><b>Action:</b> Reallocate the structure in a coupling facility with CFLEVEL=1 or higher.</p>
C	xxxx0C61	<p><b>Equate Symbol:</b> IXLRNCODEALLOWALTER</p> <p><b>Meaning:</b> Environmental error. Request rejected because at least one active, disconnecting, failing, or failed-persistent connection specified ALLOWALTER=NO on IXLCONN.</p> <p><b>Action:</b> Use IXCQUERY to determine the set of connections that do not support structure alter for the structure. When those connections are no longer connected to the structure, retry the IXLALTER request.</p> <p>If a connector to the structure is failed-persistent, consider the use of IXLFORCE to delete the connection. Be aware that such an action might result in the loss of persistent information.</p>
C	xxxx0C62	<p><b>Equate Symbol:</b> IXLRNCODEALTERRATIOCHG</p> <p><b>Meaning:</b> Environmental error. Request rejected because at least one active, failing, or failed-persistent connection specified RATIO=NO on IXLCONN.</p> <p><b>Action:</b> Use IXCQUERY to determine the set of connections that specified RATIO=NO for the structure. When those connections are no longer connected to the structure, retry the IXLALTER request.</p> <p>If a connector to the structure is failed-persistent, consider the use of IXLFORCE to delete the connection. Be aware that such an action might result in the loss of persistent information.</p>
C	xxxx0C63	<p><b>Equate Symbol:</b> IXLRNCODEALTERNOTINPROG</p> <p><b>Meaning:</b> Environmental error. Request rejected because structure alter stop was requested and structure alter is not in progress for the structure.</p> <p><b>Action:</b> None.</p>

Table 25. Return and reason codes for the IXLALTER macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C66	<b>Equate Symbol:</b> IXLRSNCODEALTERSTOPINPROG <b>Meaning:</b> Environmental error. Request rejected because alter stop was requested and an alter stop is already in progress. <b>Action:</b> None.
C	xxxx0C6E	<b>Equate Symbol:</b> IXLRSNCODEALTERNOTPERMITTED <b>Meaning:</b> CF structure alter is not permitted to start. This restriction might be the result of a SETXCF MODIFY ALTER=DISABLED command. A structure-specific ENF35 is to be issued when alter is permitted. <b>Action:</b> Retry the IXLALTER request when a structure-specific ENF35 is issued.
C	xxxx0C76	<b>Equate Symbol:</b> IXLRSNCODEDUPALTER <b>Meaning:</b> Environmental error. An IXLALTER request was rejected because alter stop was requested and a stop of an alter for a structure in a system-managed duplexing rebuild is not allowed. <b>Action:</b> None.
C	xxxx0CA2	<b>Equate Symbol:</b> IXLRSNCODESTORAGECLASSMEMORYINUSE <b>Meaning:</b> Environmental error. Request rejected because the structure has objects in storage-class memory. <b>Action:</b> Repeat the alter request when all storage objects are migrated back into coupling facility real storage.
C	FFFFFFFF	<b>Equate Symbol:</b> IXLRSNCODENOTAVAILABLE <b>Meaning:</b> Environmental error. There are no coupling facility services available. The hardware support necessary to provide coupling facility services might not be present. <b>Action:</b> Re-IPL the system, or follow your particular failure management protocol.
10	—	<b>Meaning:</b> XES processing has failed. <b>Action:</b> Save the reason code information and contact the IBM support center.



## Chapter 29. IXLAXISN

### Environment

The following are the environment requirements for the caller.

Environment	Environment requirement
Authorization:	
Dispatchable unit mode:	Task or SRB mode
Cross memory mode:	Any PASN, any HASN, any SASN.
AMODE:	31- or 64-bit. If in 64-bit mode, specify SYSSTATE AMODE64=YES before invoking this macro.
ASC mode:	Primary or Access Register. If in Access Register ASC mode, specify SYSSTATE ASCENV=AR before invoking this macro.
Interrupt status:	Enabled or disabled for I/O and external interrupts. When OPTYPE=SUSPEND is specified the caller must be enabled
Locks:	Disabled callers must be legally disabled by holding the CPU lock and cannot hold other disabled locks (invocation by DIE routines is not supported). Enabled callers must not hold any locks.
Control parameters:	Control parameters must be in the primary address space or, for AR-mode callers, must be in an address/data space that is addressable through a public entry on the caller's dispatchable unit access list (DU-AL).  If the caller is running in Access Register ASC mode and specifies a macro parameter using explicit register notation the access register corresponding to the general register must appropriately qualify the general register.

### Programming Requirements

1. Access Register ASC mode invokers must specify SYSSTATE ASCENV=AR prior to invoking this macro.
2. AMODE 64-bit invokers must specify SYSSTATE AMODE64=YES prior to invoking this macro.

### Performance Implications

None.

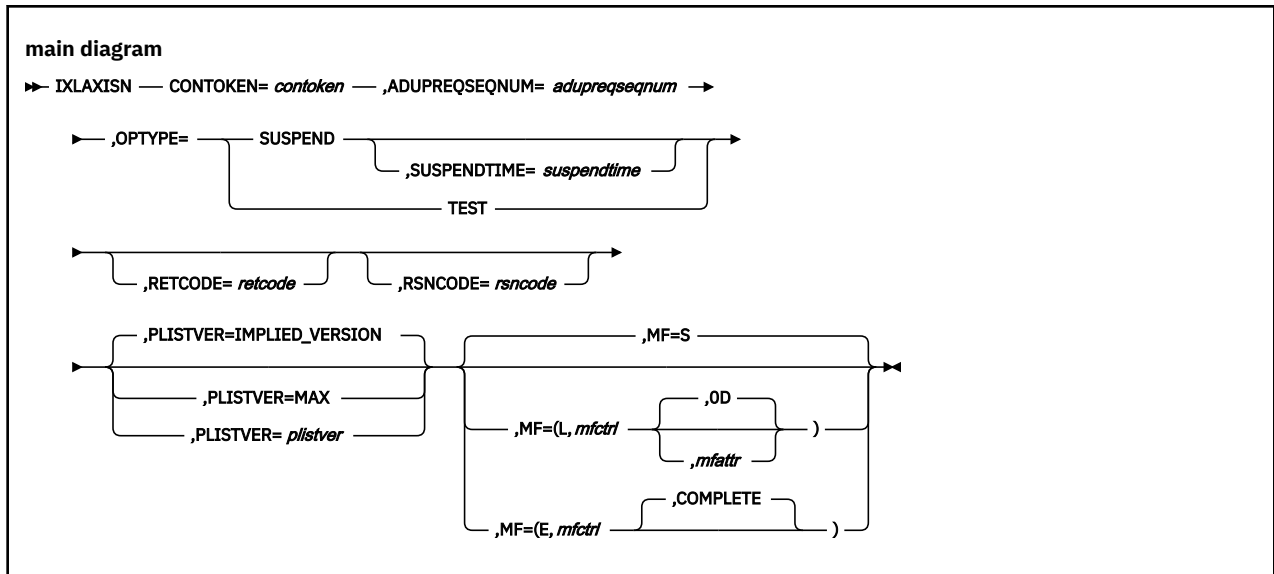
### Understanding IXLAXISN Version Support

The IXLAXISN macro supports version 0 keywords and functions.

Specify the version of the parameter list that you want generated with the PLISTVER keyword. See [Chapter 2, “Specifying a Macro Version Number,” on page 5](#) for considerations when specifying the version of the parameter list with PLISTVER.

### Syntax diagram

The syntax of IXLAXISN is written as follows:



## Input Register Information

Before issuing the IXLAXISN macro, the caller does not have to place any information into any register or AR unless using it in register notation for a particular parameter, or using it as a base register.

## Output Register Information

When control returns to the caller of the IXLALTER macro, the general purpose registers (GPRs) contain:

### Register

#### Contents

**0**

Reason code, if GPR 15 contains a non-zero return code

**1**

Used as work register by the system

**2-13**

Unchanged

**14**

Used as work register by the system

**15**

Return code

When control returns to the caller of the IXLALTER macro, the access registers (ARs) contain:

### Register

#### Contents

**0-1**

Used as work registers by the system

**2-13**

Unchanged

**14-15**

Used as work registers by the system

**For registers that the system changes,** a caller who depends on these registers containing the same value before and after issuing the macro must save these registers before issuing the macro and restore them after the system returns control.

## Parameter Descriptions

---

The parameter descriptions are listed in alphabetical order. Default values are underlined:

### **,ASYNXISEQNUM=***asynxiseqnum*

Use this input parameter to specify an asynchronous cross-invalidation sequence number returned on a previous IXLCACHE request that generated asynchronous cross-invalidations of local caches. An asynchronous cross-invalidation sequence number is returned in the IXLCACHE Answer Area (IXLYCAA) in data field CAAASYNXISEQNUM for IXLCACHE requests that generated asynchronous cross-invalidations.

An asynchronous cross-invalidation sequence number identifies cross-invalidations associated with a cache service request issued by the connector.

Cross-invalidations associated with assigned asynchronous cross-invalidation sequence numbers are completed by the coupling facility asynchronously to the completion of the cache request that caused the cross-invalidations of local caches to be generated.

Cross-invalidations with asynchronous cross-invalidation sequence numbers assigned are completed in the ascending order of these sequence numbers. If cross-invalidations identified by ASYNXISEQNUM have completed, all cross-invalidations for the connector with a lower asynchronous cross-invalidation sequence number are also completed for the connector.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the halfword field that contains the element part of the ratio.

### **,CONTOKEN=***contoken*

Use this input parameter to specify a connect token returned by the IXLCONN service. The CONTOKEN uniquely identifies the user's connection to a cache structure.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-character field that contains the connect token.

### **,MF=S**

### **,MF=(L,***mfctrl***)**

### **,MF=(L,***mfctrl***,***mfattr***)**

### **,MF=(L,***mfctrl***,0D)**

### **,MF=(E,***mfctrl***)**

### **,MF=(E,***mfctrl***,**COMPLETE**)**

Use MF=S to specify the standard form of the macro, which builds an in-line parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

### **,mfctrl**

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the parameter list.

### **,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembler DS pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

**,COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if SMILE=var were an optional parameter and the default is SMILE=NO\_SMILE then it would not be documented. However, if the default was SMILE=:), then it would be documented because a value would be the default.

**,OPTYPE=SUSPEND****,OPTYPE=TEST**

Use OPTYPE=SUSPEND to indicate that control should be returned to the invoker when the request identified by ASYNXISEQNUM has been committed in the secondary structure. The invoker must be executing in an enabled state to use this option.

Use OPTYPE=TEST to test the processing status of the request identified by ASYNXISEQNUM. The return and reason code indicate whether the request has been committed in the secondary structure. This option can be used to poll for request processing status by invokers that cannot be suspended.

**,PLISTVER=IMPLIED\_VERSION****,PLISTVER=MAX****,PLISTVER=plistver**

Use this input parameter to specify the version of the macro. See [“Understanding IXLAXISN Version Support”](#) on page 449 for a description of the options available with PLISTVER.

**,RETCODE=retcode**

Use this output parameter to specify the location in which the system is to copy the return code from GPR 15.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword to contain the return code.

**,RSNCODE=rsncode**

Use this output parameter to specify the location in which the system is to copy the reason code from GPR 0.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword to contain the reason code.

**,SUSPENDTIME=suspendtime**

Use this output parameter with OPTYPE=SUSPEND to specify a storage area to contain the amount of time, in microseconds, the invoker is suspended waiting for the cross-invalidations to complete. If the IXLAXISN request completes successfully without suspending the invoker, the content of this output variable will be zero.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-character field where the system will put the suspend time.

## Return and reason codes

---

When the IXLAXISN macro returns control to your program:

- GPR 15 (and retcode, if you coded RETCODE) contains a return code.
- GPR 0 (and rsncode, if you coded RSNCODE) contains a reason code.

Macro IXLYCON provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

**0**

IXLRETCODEOK

**4**

IXLRETCODEWARNING

8

IXLRETCODEPARMERROR

C

IXLRETCODEENVERROR

10

IXLRETCODECOMPERROR

The following table identifies the hexadecimal return and reason codes and the equate symbol associated with each reason code.

Table 26. Return and reason codes for the IXLALTER macro		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0	xxxx0000	<p><b>Equate Symbol:</b> IxLRsnCodeNoCfAccessRequired</p> <p><b>Meaning:</b> Successful completion. The cross-invalidations of local caches associated with ASYNCXISEQNUM have completed.</p> <p>Cross-invalidation completion information was obtained from local XES connector information for cross-invalidations associated with ASYNCXISEQNUM.</p> <p><b>Action:</b> None required.</p>
0	xxxx0001	<p><b>Equate Symbol:</b> IXLRSNCODECFACCESSREQUIRED</p> <p><b>Meaning:</b> Successful completion. The cross-invalidations of local caches associated with ASYNCXISEQNUM have completed.</p> <p>Cross-invalidation completion information was obtained from the coupling facility for cross-invalidations associated with ASYNCXISEQNUM.</p> <p><b>Action:</b> None required.</p>
4	xxxx043A	<p><b>Equate Symbol:</b> IXLRSNCODECROSSINVALSOUTSTANDING</p> <p><b>Meaning:</b> An OPTYPE=TEST operation found that the cross-invalidations associated with ASYNCXISEQNUM are not complete.</p> <p><b>Action:</b> Issue the IXLAXISN macro again with OPTYPE=TEST to determine whether the cross-invalidations associated with ASYNCXISEQNUM are complete. Alternatively, issue the IXLAXISN macro with OPTYPE=SUSPEND in an enabled environment to wait for the cross-invalidations associated with ASYNCXISEQNUM to complete.</p>
4	xxxx043C	<p><b>Equate Symbol:</b> IxLRsnCodeBadSeqNumInstance</p> <p><b>Meaning:</b> There are no outstanding asynchronous cross-invalidations associated with the specified ASYNCXISEQNUM for the connection because ASYNCXISEQNUM is not associated with the current instance of the structure.</p> <p><b>Action:</b> None required. However, you might want to ensure that the asynchronous cross-invalidation sequence number is correct.</p>

Table 26. Return and reason codes for the IXLALTER macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0801	<p><b>Equate Symbol:</b> IXLRSNCODEBADPARMLIST</p> <p><b>Meaning:</b> Program error. The parameter list for this request is not addressable.</p> <p><b>Action:</b> Verify that:</p> <ul style="list-style-type: none"> <li>• The parameter list address is uncorrupted.</li> <li>• The parameter list is addressable in the caller's primary address space.</li> <li>• If you are issuing this macro while disabled, the parameter list resides in either page-fixed or DREF storage.</li> <li>• If you are issuing this macro in AR-mode and you specified the parameter list address using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If you are issuing this macro in AR-mode, SYSSTATE ASCENV=AR must be issued before issuing this macro.</li> </ul>
8	xxxx0802	<p><b>Equate Symbol:</b> IXLRSNCODEBADPARMLISTALET</p> <p><b>Meaning:</b> Program error. The parameter list ALET is not valid.</p> <p><b>Action:</b> Ensure that the ALET is zero or that the ALET represents a valid entry on the DU-AL</p>
8	xxxx0804	<p><b>Equate Symbol:</b> IXLRSNCODEBADVERSION#</p> <p><b>Meaning:</b> Program error. Invalid version number in the parameter list.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that your program did not overlay the parameter list storage.</li> <li>• Verify that your program was assembled with the correct macro library for the release of MVS your program is running on.</li> </ul>
8	xxxx080A	<p><b>Equate Symbol:</b> IXLRSNCODEBADCONTOKEN</p> <p><b>Meaning:</b> Program error. The input contoken is not valid. The contoken may no longer be valid for one of the following reasons: disconnect has occurred, EOT of the connector's task, input contoken is not the contoken returned from IXLCONN, the request was issued outside the connector's address space, or the contoken has been invalidated for rebuild.</p> <p><b>Action:</b> Verify that the CONTOKEN value specified is valid and for the correct structure.</p>

Table 26. Return and reason codes for the IXLALTER macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0824	<p><b>Equate Symbol:</b> IXLRSNCODEWRONGSTRTYPE</p> <p><b>Meaning:</b> Program error. The connection specified by CONTOKEN is not to a cache structure.</p> <p><b>Action:</b> Specify the appropriate CONTOKEN value, and rerun your program. The CONTOKEN is returned in the answer area provided on the IXLCONN invocation. The type of structure built is passed on the TYPE parameter of the IXLCONN macro.</p>
8	xxxx0851	<p><b>Equate Symbol:</b> IXLRSNCODENOSUSPENDISABLE</p> <p><b>Meaning:</b> Program error. The request failed because OPTYPE=SUSPEND was specified, but the caller is disabled and cannot be suspended.</p> <p><b>Action:</b> Either specify another OPTYPE value, or become enabled (release the CPU lock), then reissue the request.</p>
8	xxxx08A2	<p><b>Equate Symbol:</b> IxlRsnCodeBadAsyncXiSeqNum</p> <p><b>Meaning:</b> Program error. An invalid asynchronous cross-invalidation sequence number was specified. The value specified in ASYNCXISEQNUM does not represent a valid asynchronous cross-invalidation sequence number returned on a previous IXLCACHE invocation.</p> <p><b>Action:</b> Ensure that you use only an asynchronous cross-invalidation sequence number that is returned on a previous IXLCACHE invocation issued against the current logical version of the cache structure.</p>
8	xxxx08A3	<p><b>Equate Symbol:</b> IXLRSNCODENOASYNXCICONN</p> <p><b>Meaning:</b> Program error. The connector did not specify ASYNCXI=IxlConnAsyncXiYes on the IXLCONN invocation when connecting to the cache structure.</p> <p><b>Action:</b> Ensure that the input CONTOKEN identifies a connector that specified ASYNCXI=IxlConnAsyncXiYes on an IXLCONN invocation.</p>
8	xxxx08B6	<p><b>Equate Symbol:</b> IXLRSNCODEBADXSUSPENDENV</p> <p><b>Meaning:</b> Program error. The IXLAXISN OPTYPE=SUSPEND operation was issued from a SUSPEND exit routine or from an SRB routine that the system abended with a 47B system completion code. The caller cannot be suspended while running in this environment.</p> <p><b>Action:</b> Either specify another OPTYPE value, or reissue the IXLAXISN macro from an environment that allows the caller to be suspended.</p>

Table 26. Return and reason codes for the IXLALTER macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
B	xxxx08B7	<p><b>Equate Symbol:</b> IXLRSNCODEBADLOCKS</p> <p><b>Meaning:</b> Program error. The caller's environment does not match the serialization and interrupt status requirements of the IXLAXISN service routine. For example:</p> <ul style="list-style-type: none"> <li>• Caller is enabled but holds locks.</li> <li>• Caller is disabled but does not hold the CPU lock.</li> <li>• Caller is disabled but holds disabled locks in addition to the CPU lock.</li> </ul> <p><b>Action:</b> Reissue the IXLAXISN macro in an environment that matches the requirements of the service macro.</p>
C	xxxx0C06	<p><b>Equate Symbol:</b> IXLRSNCODENOCNN</p> <p><b>Meaning:</b> Environmental error. This system does not have connectivity to the coupling facility that contains the structure. Possible reasons for this are:</p> <ul style="list-style-type: none"> <li>• The operator issued VARY PATH,OFFLINE.</li> <li>• The operator issued CONFIG CHP,OFFLINE.</li> <li>• Hardware errors to the coupling facility.</li> <li>• Facility or path failure to the coupling facility.</li> </ul> <p><b>Action:</b> Begin rebuilding the structure on a different coupling facility, or disconnect from the structure.</p>
C	xxxx0C25	<p><b>Equate Symbol:</b> IXLRSNCODESTRFAILURE</p> <p><b>Meaning:</b> Environmental error. The structure failed prior to completion of the request.</p> <p><b>Action:</b> Either rebuild or disconnect from the structure.</p>
10	xxxx010xx	<p><b>Meaning:</b> XES processing failure. The state of the involved structure and the disposition of the request are unpredictable.</p> <p><b>Action:</b> Contact the IBM support center.</p>



## Chapter 30. IXLCACHE – Cache Services

### Description

A cache structure is a coupling facility structure that enables high-performance sharing of cached data by applications in a sysplex. The IXLCACHE macro provides the means for applications in a sysplex to access and manipulate the data in a coupling facility cache structure.

The IXLCACHE macro has many REQUEST types. These requests enable the cache structure user to manage, manipulate, and share data with other connected users of the cache structure. Each request type is detailed in the following sections. Be sure to read the IXLCACHE guidance information in *z/OS MVS Programming: Sysplex Services Guide*. The information presented here assumes you have done so. The following table tells you where to find the reference information for each request type:

Table 27. Request Types for IXLCACHE	
Request Type	Description
CASTOUT_DATA	Read a data item from an entry in the cache structure to the storage areas specified by BUFFER or BUFLIST and/or ADJAREA. From these areas, the data can be written to permanent storage such as DASD.  See <a href="#">Chapter 31, “IXLCACHE REQUEST=CASTOUT_DATA,” on page 463.</a>
CASTOUT_DATALIST	Read data items from 1 to 8 entries in the cache structure to the storage areas specified by BUFFER, BUFLIST, DEIBAREA, and/or ADJAREA. From these areas, the data can be written to permanent storage such as DASD.  See <a href="#">Chapter 32, “IXLCACHE REQUEST=CASTOUT_DATALIST,” on page 485.</a>
CROSS_INVAL	Deregisters interest and invalidates the copies of a data item in the local cache buffers of users with registered interest in the data item other than the requesting user.  See <a href="#">Chapter 33, “IXLCACHE REQUEST=CROSS_INVAL,” on page 507.</a>
CROSS_INVALLIST	Deregisters interest and invalidates the copies of from 1 to 4096 data items in the local cache buffers of users with registered interest in the data item other than the requesting user.  See <a href="#">Chapter 34, “IXLCACHE REQUEST=CROSS_INVALLIST,” on page 523.</a>
DELETE_NAME	Delete one or more data items identified by NAME and NAMEMASK from the coupling facility cache structure (this frees directory and data entry resources and invalidates any registered users).  See <a href="#">Chapter 35, “IXLCACHE REQUEST=DELETE_NAME,” on page 543.</a>
DELETE_NAMELIST	Delete a set of data entries from the coupling facility cache structure. Each entry to be deleted is represented by a name block contained in the storage area specified by BUFFER or BUFLIST.  See <a href="#">Chapter 36, “IXLCACHE REQUEST=DELETE_NAMELIST,” on page 561.</a>
PROCESS_REFLIST	Process one or more cached data items to indicate that they are recently referenced.  See <a href="#">Chapter 37, “IXLCACHE REQUEST=PROCESS_REFLIST,” on page 587.</a>

Table 27. Request Types for IXLCACHE (continued)	
Request Type	Description
READ_COCLASS	Read directory entry names and directory entry user data for the data items specified by NAME and NAMEMASK, and the cast-out class specified by COCLASS and store in BUFFER/BUFLIST area. See <a href="#">Chapter 38, “IXLCACHE REQUEST=READ_COCLASS,” on page 605.</a>
READ_COSTATS	Retrieve statistical information for the specified cast-out classes. See <a href="#">Chapter 39, “IXLCACHE REQUEST=READ_COSTATS,” on page 629.</a>
READ_DATA	Read a data item from a data entry in the coupling facility cache structure into the storage areas specified by BUFFER or BUFLIST and/or ADJAREA. See <a href="#">Chapter 40, “IXLCACHE REQUEST=READ_DATA,” on page 649.</a>
READ_DIRINFO	Retrieve directory information from the coupling facility cache structure and store it in the storage area specified by BUFFER or BUFLIST. See <a href="#">Chapter 41, “IXLCACHE REQUEST=READ_DIRINFO,” on page 675.</a>
READ_STGSTATS	Retrieve statistical information for a specified storage class and store it in the storage area specified by STGSTATS. See <a href="#">Chapter 42, “IXLCACHE REQUEST=READ_STGSTATS,” on page 697.</a>
REG_NAMELIST	Create a directory entry and register interest in a list of up to 32 data items and store the resulting state information in the storage area specified by NSBAREA. See <a href="#">Chapter 43, “IXLCACHE REQUEST=REG_NAMELIST,” on page 709.</a>
RESET_REFBIT	Reset the reference bit in the directory entries for the specified cached data items to indicate that the data items were not recently referenced and obtain a count of referenced items. See <a href="#">Chapter 44, “IXLCACHE REQUEST=RESET_REFBIT,” on page 727.</a>
SET_RECLVCTR	Define and activate, or deactivate, a reclaiming vector for a specified storage class. See <a href="#">Chapter 45, “IXLCACHE REQUEST=SET_RECLVCTR,” on page 743.</a>
UNLOCK_CASTOUT	Releases the cast-out lock for the data items named in BUFFER or BUFLIST. See <a href="#">Chapter 46, “IXLCACHE REQUEST=UNLOCK_CASTOUT,” on page 757.</a>
UNLOCK_CO_NAME	Releases the cast-out lock for a single data item named in CUNBAREA. See <a href="#">Chapter 47, “IXLCACHE REQUEST=UNLOCK_CO_NAME,” on page 777.</a>
WRITE_DATA	Write data from the storage areas specified by BUFFER or BUFLIST and/or ADJAREA to a data entry in the coupling facility cache structure. See <a href="#">Chapter 48, “IXLCACHE REQUEST=WRITE_DATA,” on page 791.</a>
WRITE_DATA LIST	Write data from a list of entries specified in either BUFFER or BUFLIST to data entries in the coupling facility cache structure. See <a href="#">Chapter 49, “IXLCACHE REQUEST=WRITE_DATA LIST,” on page 821.</a>

## Environment

The following are the environment requirements for the caller.

Environment	Environment requirement
Minimum authorization:	Supervisor state and PSW key 0-7
Dispatchable unit mode:	Task or SRB
Cross memory mode:	Any PASN, any HASN, any SASN. The primary address space must be the same as the primary address space at the time the connection service (IXLCONN) was issued.
AMODE:	31-bit
ASC mode:	Primary or access register (AR).
Interrupt status:	Enabled or disabled for I/O and external interrupts
Locks:	Disabled callers must be legally disabled by holding the CPU lock and cannot hold other disabled locks. Enabled callers must not hold any locks. If MODE=SYNCSUSPEND is specified, the caller must be enabled.
Control parameters:	See “Restrictions” on page 460

## Programming Requirements

- If your program is in AR mode, issue the SYSSTATE ASCENV=AR macro before IXLCACHE. SYSSTATE ASCENV=AR tells the system to generate code appropriate for AR mode.
- Include the IXLYCON mapping macro in your program. This macro provides a list of equate symbols for users of XES services and exits.
- Include mapping macros IXLYCAA, IXLYCANB, IXLYCCIH, IXLYCRRB, IXLYCSCS, IXLYCUNB, IXLYDEIB, IXLYNSB, IXLYDNNB, IXLYWOB, and IXLYWORB in your program as necessary. Table 28 on page 459 lists these macros, the information and areas they map, and the particular IXLCACHE requests and parameters to which they apply.

*Table 28. Mapping Macros for IXLCACHE*

Mapping Macro	Information	Area	Request/Parameter
IXLYCAA	Answer area output	ANSAREA area	All requests
IXLYCANB	Entry name and associated directory information	BUFLIST buffers, BUFFER area	READ_DIRINFO (when DIRINFOFMT=NAMELIST), READ_COCLASS
IXLYCCIH	Cast-out class range	BUFLIST buffers, BUFFER area	READ_COSTATS
IXLYCRRB	Single registration block	BUFFER area	REG_NAMELIST
IXLYCSCS	Storage class statistics	STGSTATS area	READ_STGSTATS
IXLYCUNB	Entry name and associated control information	BUFLIST buffers, BUFFER area, CUNBAREA	UNLOCK_CASTOUT UNLOCK_CO_NAME
IXLYDEIB	Directory entry	BUFLIST buffers, BUFFER area, DEIBAREA	READ_DIRINFO (when DIRINFOFMT=DIRENTRYLIST), READ_COCLASS (when DIRINFOFMT=DIRENTRYLIST), CASTOUT_DATA LIST

Table 28. Mapping Macros for IXLCACHE (continued)

Mapping Macro	Information	Area	Request/Parameter
IXLYDNNB	Entry name and optional comparative version	BUFLIST buffers, BUFFER area	DELETE_NAMELIST
IXLYNSB	Name and state information for registration blocks	NSBAREA	REG_NAMELIST
IXLYWOB	Entry name and associated control information.	BUFLIST buffers, BUFFER area	WRITE_DATA LIST
IXLYWORB	Single write-operation-response block	WORBAREA	WRITE_DATA LIST

## Restrictions

- This service cannot be invoked by callers running as a disabled interrupt exit (DIE).
- The caller's parameter list must be addressable in the caller's primary address space.
- If the caller is running in AR ASC mode and specifies a parameter using register notation, the access register corresponding to the general register must appropriately qualify the general register.
- The virtual storage areas specified by ADJAREA, ANSAREA, NSBAREA, and STGSTATS must be addressable in the caller's primary address space or from the caller's PASN access list. On a REG\_NAMELIST request, the virtual storage area specified by BUFFER must be addressable in the caller's primary address space or from the caller's PASN access list.
- The virtual storage areas specified by parameters other than ADJAREA, ANSAREA, NSBAREA, and STGSTATS (and BUFFER on a REG\_NAMELIST request) can be addressable in the caller's primary, secondary, or home address spaces, from the PASN access list, or from the dispatchable unit access list (DU-AL).
- If the caller is disabled, then the parameter list and all storage areas addressed by the parameters must reside in either nonpageable or disabled reference storage.
- If you specify BUFLIST and PAGEABLE=YES, all the buffers in the list must be in the same area of storage; you cannot mix common and private storage in the list of buffers.

## Input Register Information

Before issuing the IXLCACHE macro, the caller does not have to place any information into any register unless using it in register notation for a particular parameter, or using it as a base register.

## Output Register Information

When control returns to the caller, the GPRs contain:

### Register Contents

**0**

Reason code

**1**

Used as work register

**2-13**

Unchanged

**14**

Used as work register

**15**

Return code

When control returns to the caller, the ARs contain:

**Register****Contents****0-1**

Used as work register

**2-13**

Unchanged

**14-15**

Used as work register

## Performance Implications

---

See *z/OS MVS Programming: Sysplex Services Guide* for performance information.

## Understanding IXLCACHE Version Support

---

The IXLCACHE macro supports versions in the range of 0 - 8.

- Keywords not specifically noted here are supported by all versions starting with version 0 and higher of the IXLCACHE macro.
- The following keywords and functions are supported by all versions starting with version 1 and higher of the IXLCACHE macro.
  - ENDINDEX
  - NSBAREA
  - STARTINDEX
- The following keywords and functions are supported by all versions starting with version 2 and higher of the IXLCACHE macro.
  - CUNBAREA
  - RETURNDATA
  - REQUEST=UNLOCK\_CO\_NAME
- The following keywords and functions are supported by all versions starting with version 3 and higher of the IXLCACHE macro.
  - COSTATSFMT
  - DELETETYPE
  - ERRORACTION
  - NEWVERS
  - VERSCOMP
  - VERSCOMPTYPE
  - VERSUPDATE
  - REQUEST=DELETE\_NAMELIST
- The following keyword is supported by all versions starting with version 4 and higher of the IXLCACHE macro. IXLCACHE macro invocations at the version 4 and higher level require an answer area length of CAALEVEL1LEN.
  - EXTRESTOKEN
- The following keywords and functions are supported by all versions starting with version 6 and higher of the IXLCACHE macro.

- CASTOUTLIST
- DATAOFFSET
- DEIBAREA
- WORBAREA
- REQUEST=CASTOUT\_DATALIST
- REQUEST=CROSS\_INVALLIST
- REQUEST=WRITE\_DATALIST
- 
- The following keywords and functions are supported by all versions starting with version 7 and higher of the IXLCACHE macro.
  - LOCALREGCNTL
- The following keywords and functions are supported by all versions starting with version 8 and higher of the IXLCACHE macro.
  - HALTONCHANGED
  - SUPPCROSSINVAL
  - AXIOVERRIDE

Specify the version of the parameter list that you want generated with the PLISTVER keyword. See Chapter 2, [“Specifying a Macro Version Number,”](#) on page 5 for considerations when specifying the version of the parameter list with PLISTVER.

---

## Chapter 31. IXLCACHE REQUEST=CASTOUT\_DATA

### Description

---

A CASTOUT\_DATA request allows you to read a data item from an entry in the cache structure to the storage areas specified by BUFFER or BUFLIST and/or ADJAREA. From these areas, the data can be written to permanent storage such as DASD. This is part of the cast-out process.

When you issue a CASTOUT\_DATA request:

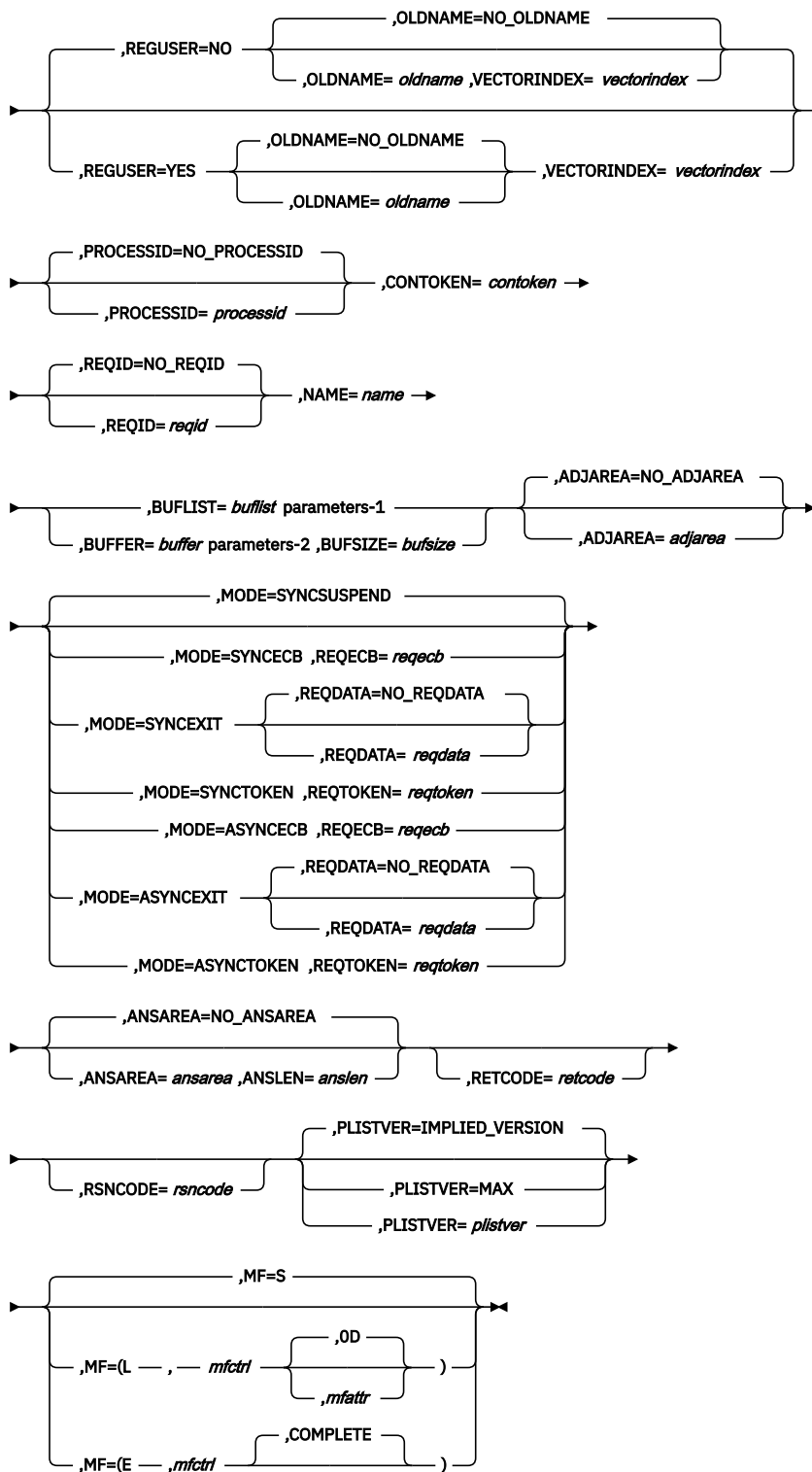
- A cast-out lock is obtained for the data item specified by NAME. This lock protects the data item from further cast-out processing until the cast-out lock is released. (To release the cast-out lock, you must issue an UNLOCK\_CASTOUT request.)
- The change indicator in the directory entry for the data item is updated to indicate the data entry contains unchanged data. (Note that the data is considered changed until the cast-out lock is released.)

This request also allows you to register your interest (REGUSER=YES) in the data item. To register or update your interest in a data item, you must specify an index into your local cache vector (VECTORINDEX) to be associated with the named data item in the structure. This index identifies a vector entry that cache services uses to indicate both your interest in the data item and the validity of your locally cached copy of the data item. Having registered interest in the data item **means** that you have a valid copy of the data item in your local cache buffer. When another user updates the data item in the structure, cache services will update the data item's associated vector entry, thereby deregistering your interest in the data item and invalidating your locally cached copy.

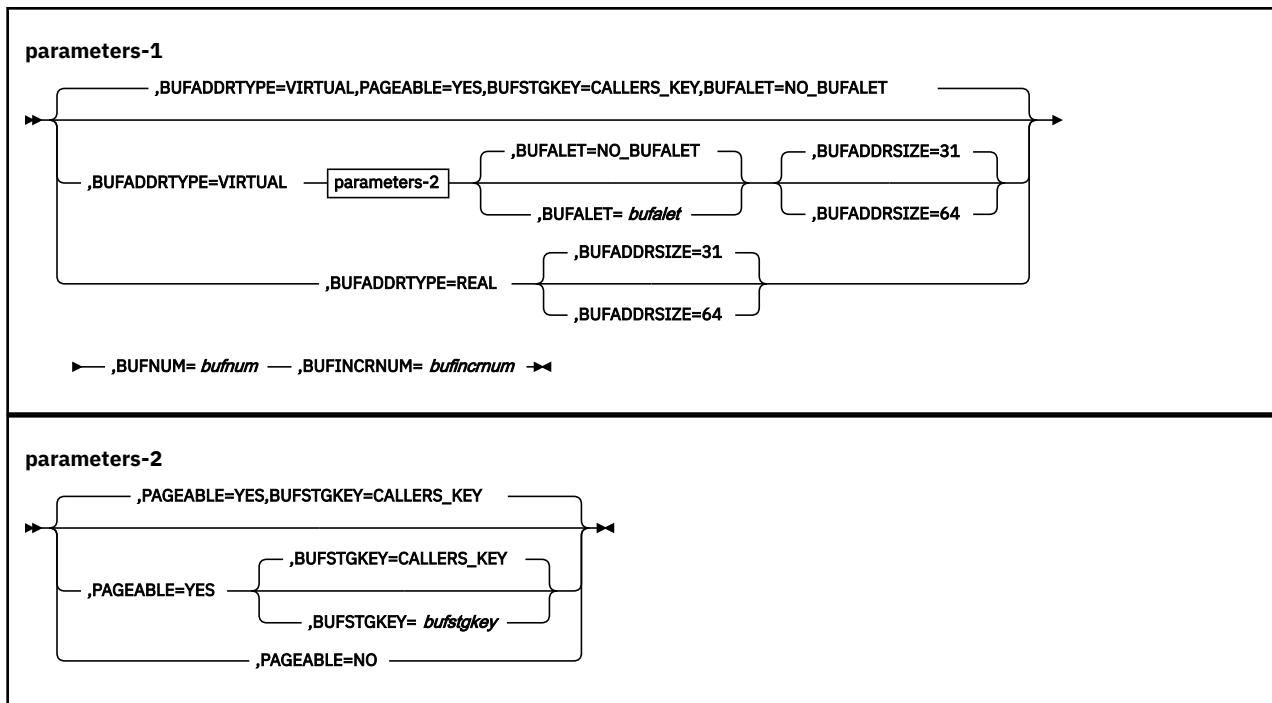
### Syntax Diagram

---

The syntax diagram for IXLCACHE REQUEST=CASTOUT\_DATA is as follows:







**Note:** If you specify `MODE=SYNCTOKEN` or `MODE=ASYNCTOKEN`, then you must also specify `ANSAREA=ansarea`, `ANSLEN=anslen`.

## Parameter Descriptions

The parameter descriptions for `REQUEST=CASTOUT_DATA` are listed in alphabetical order. Default values are underlined:

### **REQUEST=CASTOUT\_DATA**

Use this input parameter to:

- Specify that a cast-out lock be obtained for the data item identified by `NAME`, and that the named data item be read into the storage areas specified `BUFFER` or `BUFLIST` and/or `ADJAREA`.
- Optionally register interest in the data item (Specify `REGUSER=YES` also).

### **,ADJAREA=NO\_ADJAREA**

### **,ADJAREA=adjarea**

Use this output parameter to specify a storage area to contain the adjunct data that was read from one or more buffers designated entry.

If the structure supports adjunct data and `ADJAREA` is not specified, or if `ADJAREA=NO_ADJAREA` is specified, then no adjunct data is returned.

If the structure does not support adjunct data and `ADJAREA` is specified, then no adjunct data is returned, and the request will complete with a return code `X'4'` and a reason code of `IXLRSNCODENOADJUNCTDATA`.

If the structure supports adjunct data and `ADJAREA` is specified, but there is no data and adjunct associated with the entry, then:

- For `READ_DATA` requests, the request completes with return code `X'4'` and reason code `IXLRSNCODENOREADDATA`.
- For `CASTOUT_DATA` requests, the request completes with return code `X'8'` and reason code `IXLRSNCODECOUNCHANGED`.

(Adjunct areas for a structure are established through the `IXLCONN` macro.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 64-byte virtual storage area that the adjunct data will be read in.

**,ANSAREA=NO\_ANSAREA****,ANSAREA=ansarea**

Use this output parameter to specify an answer area to contain information returned from the request. The format of the answer area is described by the IXLYCAA mapping macro. On a successful completion of the request, the following information is returned in the answer area:

- The parity of the returned data item (field CAAPARITY).
- If REGUSER=YES is specified and the request replaced a previously specified local cache vector index for the entry, the replaced index is returned (fields CAAINVLCVI and CAAINVLCVINUM).
- If BUFFER or BUFLIST is specified, the number of elements in the retrieved entry is returned (field CAAELEMNUM).

**To Code:** Specify the RS-type name or address (using a register from 2 - 12) of an area (with a length of ANSLLEN) where the information returned from the request will go.

**,ANSLLEN=anslen**

Use this inppaameter to specify the size of the storage area specified by ANSAREA.

Use either CAALEVEL0LEN, CAALEVEL1LEN or CAALEVEL2LEN of the IXLYCAA mapping macro to determine the minimum size of the answer area. The answer area length must be at least large enough to accommodate the level of the IXLYCAA mapping appropriate to the requested function.

- When the connection specified ASYNCXI=1 on the IXLCONN invocation and AXIOVERRIDE=0 was specified or defaulted to for the IXLCACHE request, the answer area length is a required parameter and must be a minimum value of CAALEVEL2LEN to contain a returned asynchronous cross-invalidation sequence number (CAAASYNCXISEQNUM).
- When the value of PLISTVER is 4 or above, the minimum answer area length is CAALEVEL1LEN.
- When the value of PLISTVER is 0 - 3, the minimum answer area length is CAALEVEL0LEN.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12 2 to ) of a 2-byte field that contains the length of the answer area (ANSAREA).

**,BUFADDRSIZE=31****,BUFADDRSIZE=64**

Use this input parameter to specify whether a 31-bit or a 64-bit address is specified by a BUFLIST entry.

**31**

The entry in BUFLIST is 31 bits in size.

**64**

The entry in BUFLIST is 64 bits in size.

**,BUFADDRTYPE=VIRTUAL****,BUFADDRTYPE=REAL**

Use this input parameter to specify whether the buffer addresses specified in the BUFLIST list are virtual storage or real storage addresses.

**VIRTUAL**

The buffer addresses are virtual storage addresses. The virtual storage can be pageable or nonpageable. See the PAGEABLE parameter for information about managing storage binds when specifying virtual storage addresses.

**REAL**

The buffer addresses are real storage addresses.

It is the caller's responsibility to manage the binds between the data buffer virtual storage and the real storage addresses provided. The caller must ensure that the data buffer virtual storage remains bound to the real storage addresses provided until the request completes.

**,BUFALET=NO\_BUFALET****,BUFALET=bufalet**

Use this input parameter to specify an access list entry token (ALET) to be used in referencing all of the buffers specified by BUFLIST.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 4-byte field that contains the ALET.

**,BUFFER=buffer**

Use this output parameter to specify a buffer area to hold the entry data that is cast out from the cache structure. Only 31-bit addressable storage areas (below 2GB) are supported by the BUFFER specification. High virtual storage areas (above 2GB) can only be specified via the BUFLIST specification.

You can define the buffer size to be a total of up to 65536 bytes. Other requirements depend on the size you select:

- If you specify a buffer size of less than or equal to 4096 bytes, you must ensure that the buffer:
  - Is 256, 512, 1024, 2048, or 4096 bytes.
  - Starts on a 256-byte boundary.
  - Does not cross a 4096-byte boundary.
  - Does not start below storage address 512.
- If you specify a buffer size of greater than 4096 bytes, you must ensure that the buffer:
  - Is a multiple of 4096 bytes.
  - Is less than or equal to 65536 bytes.
  - Starts on a 4096-byte boundary.
  - Does not start below storage address 512.

See the BUFSIZE parameter description for defining the size of the buffer.

See *z/OS MVS Programming: Sysplex Services Guide* for more information on buffers.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an area (with a length of BUFSIZE) to receive the entry requested.

**,BUFINCRNUM=bufincrnum**

Use this input parameter to specify the number of 256-byte segments comprising each buffer in the BUFLIST list.

Valid BUFINCRNUM values are 1, 2, 4, 8, or 16, which correspond to BUFLIST buffer sizes of 256, 512, 1024, 2048, and 4096 bytes respectively.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 1-byte field that contains 1, 2, 4, 8, or 16.

**BUFLIST=buflist**

Use this output parameter to specify a list of buffers to hold the entry data that is cast out of the cache structure. BUFLIST specifies a 128-byte storage area that consists of a list of 0 to 16 buffer addresses.

Either 31-bit addressable (below 2GB) or 64-bit addressable (above 2GB) real or virtual storage areas are supported for the BUFLIST specification, depending on the specification for the BUFADDRTYPE and BUFFADDRSIZE keywords. However, pageable high shared virtual storage areas (above 2GB) may not be used.

The format of the list is a set of 8-byte elements. The BUFADDRSIZE keyword denotes whether four or eight bytes of the element are used.

- If BUFADDRSIZE=31 is specified, then the first four bytes of each element are reserved space and the last four bytes contain the address of the buffer.
- If BUFADDRSIZE=64 is specified, then the full eight bytes specify the address of the buffer.

**The BUFLIST buffers must:**

- Reside in the same address space or data space.
- Be the same size: either 256, 512, 1024, 2048, or 4096 bytes.

- Start on a 256-byte boundary and not cross a 4096-byte boundary.

**Note:** The buffers do not have to be contiguous in storage. XES treats BUFLIST buffers as a single buffer even if the buffers are not contiguous.

See the BUFNUM and BUFINCRNUM keyword descriptions for specifying the number and size of buffers.

See *z/OS MVS Programming: Sysplex Services Guide* for more information on buffers.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 128-byte field that contains a list of buffer addresses.

#### **,BUFNUM=*bufnum***

Use this input parameter to specify the number of buffers in the BUFLIST list. Valid BUFNUM values are 0 - 16. A value of zero indicates that no data is to be read into the buffers.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 1-byte field that contains the number of buffers (0 - 16) in the list (BUFLIST).

#### **,BUFSIZE=*bufsize***

Use this input parameter to specify the size of the BUFFER area. See the BUFFER parameter description for valid buffer sizes.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a fullword field that contains the size of the buffer (BUFFER) in bytes.

#### **,BUFSTGKEY=CALLERS\_KEY**

#### **,BUFSTGKEY=*bufstgkey***

Use this input parameter to specify a storage key that you define and use when referencing the buffers specified by BUFLIST or the buffer that is specified by BUFFER.

If you do not specify BUFSTGKEY, or if you specify BUFSTGKEY=CALLERS\_KEY, all references to one or more buffers are performed by using the caller's PSW key.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of an 8-bit field that contains the storage key in the format B'kkkkxxxx', where kkkk is the key and xxxx is ignored.

#### **,CONTO=*conken***

Use this input parameter to specify the connect token that was returned by the IXLCONN service in the IXLCONN answer area, which is mapped which is by IXLYCONA. The connect token uniquely identifies your connection to the cache structure, and must be specified on each IXLCACHE invocation.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field that contains the connect token.

#### **,MF=S**

#### **,MF=(L,*mfctrl*)**

#### **,MF=(L,*mfctrl*,*mfattr*)**

#### **,MF=(L,*mfctrl*,0D)**

#### **,MF=(M,*mfctrl*)**

#### **,MF=(M,*mfctrl*,COMPLETE)**

#### **,MF=(M,*mfctrl*,NOCHECK)**

#### **,MF=(E,*mfctrl*)**

#### **,MF=(E,*mfctrl*,COMPLETE)**

#### **,MF=(E,*mfctrl*,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

#### **,mfctrl**

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

#### **,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

#### **,COMPLETE**

#### **,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

#### **COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if SMILE=*var* were an optional parameter and the default is SMILE=NO\_SMILE then it would not be documented. However, if the default was SMILE=:~), then it would be documented because a value would be the default.

#### **NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

#### **,MODE=SYNCSUSPEND**

#### **,MODE=SYNCECB**

#### **,MODE=SYNCEXIT**

#### **,MODE=SYNCTOKEN**

#### **,MODE=ASYNCECB**

#### **,MODE=ASYNCEXIT**

#### **,MODE=ASYNCTOKEN**

Use this input parameter to specify:

- Whether the request is to be performed synchronously or asynchronously
- How you want to be notified of request completion if the request is processed asynchronously.

See *z/OS MVS Programming: Sysplex Services Guide* for more information on understanding synchronous and asynchronous cache operations.

#### **SYNCSUSPEND**

The request processes synchronously. If necessary, the request is suspended until it can complete synchronously. To use this option, your program must be enabled for I/O and external interrupts.

#### **SYNCECB**

The request processes synchronously if possible. If the request processes asynchronously, the ECB specified by REQECB is posted when the request completes.

#### **SYNCEXIT**

The request processes synchronously if possible. If the request processes asynchronously, your complete exit is given control when the request completes.

**SYNCTOKEN**

The request processes synchronously if possible. If the request processes asynchronously, an asynchronous request token is returned to the area specified by REQTOKEN. Use the returned request token on the IXLFCOMP macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=SYNCTOKEN is specified.

**ASYNCECB**

The request processes asynchronously. The ECB specified by REQECB is posted when the request completes.

**ASYNCEXIT**

The request processes asynchronously. Your complete exit is given control when the request completes.

**ASYNCTOKEN**

The request processes asynchronously. An asynchronous request token is returned to the area specified by REQTOKEN. Use the returned request token on the IXLFCOMP macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=ASYNCTOKEN is specified.

**,NAME=name**

Use this input parameter to specify the name of the data item to be cast-out and for which the cast-out lock will be obtained.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-byte field that contains the name of the data item.

**,OLDNAME=NO\_OLDNAME****,OLDNAME=oldname**

Use this input parameter to specify the name of the data item for which your interest should be unregistered.

- For structures allocated in a coupling facility of CFLEVEL=4 or lower, you must also specify the name of the data item for which interest is to be registered after the interest is unregistered in OLDNAME. Identify the data item to which interest is to be registered with NAME. The VECTORINDEX currently associated with the entry specified by OLDNAME will be reassigned to the name of the data item specified by NAME.
- For structures allocated in a coupling facility of CFLEVEL=5 or higher, you can specify that interest in the name of the data item specified by OLDNAME is to be unregistered without registering interest in the entry specified by NAME.

In either case, whenever deregistration of interest is requested, VECTORINDEX must be specified.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field that contains the name of the old data item that was associated with VECTORINDEX.

**,PAGEABLE=YES****,PAGEABLE=NO**

Use this input parameter to identify whether the storage areas specified by BUFFER or BUFLIST are in pageable or potentially pageable storage.

**YES**

Specify this option to indicate that the BUFFER or BUFLIST buffers reside in pageable virtual storage. XES performs the required page fixing to fix the buffers in real storage while the cache or list request transfers data to or from the coupling facility.

This includes storage obtained from pageable subpools, disabled reference (DREF) storage, and may include storage that has the potential to become pageable during the processing of a request. (An example is address space storage owned by any swappable address space, for which a PGSER FIX has been successfully processed, but for which the owning address space gets swapped during processing of a cache or list request.) This does not include implicitly non-pageable storage (for example, storage obtained from non-pageable subpools).

The system takes responsibility for managing binds to central storage for the duration of the cache or list request, regardless of what address space owns the storage or whether the storage-owning address space is swappable or nonswappable. The storage can be owned by any address space.

High shared virtual storage areas (above 2GB) may not be used.

## NO

Specify this option to indicate that the BUFFER or BUFLIST buffers reside in non-pageable virtual storage. XES does not page fix the buffers in real storage.

This includes implicitly non-pageable storage areas (for example, storage obtained from non-pageable subpools), and may include storage that has the potential to become pageable during the processing of a request (An example is address space storage owned by any swappable address space, for which a PGSER FIX has been successfully processed, but for which the owning address space gets swapped-out during processing of a cache or list request.)

The system takes responsibility for managing binds to central storage for the duration of the cache or list request, if and only if the non-pageable storage is owned by either the requester's address space or the connector's address space. If the storage is owned by any other address space, then the invoker is responsible for ensuring that the virtual storage remains non-pageable for the duration of the request (including the case in which the storage is owned by a swappable address space that is swapped during processing of an IXLCACHE or IXLLIST request). Subject to this consideration, the storage can be owned by any address space. See *z/OS MVS Programming: Sysplex Services Guide*.

### **,PLISTVER=IMPLIED\_VERSION**

### **,PLISTVER=MAX**

### **,PLISTVER=plistver**

Use this input parameter to specify the version of the macro. See “[Understanding IXLCACHE Version Support](#)” on page 461 for a description of the options available with PLISTVER.

### **,PROCESSID=NO\_PROCESSID**

### **,PROCESSID=processid**

Use this input parameter to specify a user-defined process identifier to be compared with the cast-out lock along with the connection identifier.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12 2 to 12) of a 1-byte field that contains the user-defined process identifier.

### **,REGUSER=NO**

### **,REGUSER=YES**

Use this input parameter to specify whether the request should register (or update) interest in the data item to be cast-out.

## NO

Connection interest in the data item will not be registered.

For structures allocated in a coupling facility with CFLEVEL=5 or higher, OLDNAME may be specified to deregister any outstanding interest for the specified local cache vector index for a different entry. If OLDNAME is specified, then VECTORINDEX is required.

## YES

Connection interest in the data item will be registered for the connection specified by CONTOKEN. If interest is already registered, the vector index specified by VECTORINDEX replaces any previously specified index for the data item.

Specify OLDNAME to deregister any outstanding interest for the specified vector index for a different entry.

### **,REQDATA=NO\_REQDATA**

### **,REQDATA=reqdata**

Use this input parameter with MODE=SYNCEXIT or MODE=ASYNCEXIT to pass any data you choose to the complete exit. The exit will get control only if the request is processed asynchronously.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the data to be passed to the complete exit.

**,REQECB=***reqecb*

Use this output parameter with either MODE=SYNCECB or MODE=ASYNCECB to specify the address of an ECB, which is to be posted when the request completes if the request was processed asynchronously.

Before coding REQECB, you must ensure that:

- You initialize the ECB before you issue the request.
- The ECB resides in either common storage or the home address space where IXLCONN was issued.
- Any tasks that wait for the ECB to be posted reside in the home address space where IXLCONN was issued.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that contains the address of the ECB to be posted when the request completes. The ECB must be aligned on a fullword boundary.

**,REQID=NO REQID**

**,REQID=***reqid*

Use this input parameter to specify a user-defined request identifier to be associated with the request. You can specify this request identifier on the IXPURGE macro to cancel a request that has not yet been processed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the user-defined request identifier.

**,REQTOKEN=***reqtoken*

Use this output parameter with either MODE=SYNCTOKEN or MODE=ASYNCTOKEN to specify the address of a storage area to receive the request token that is returned when the request will be processed asynchronously. This token, which uniquely identifies the request, must be used as input to the IXLFCOMP macro, which you use to determine if the request has completed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field where the system will put the request token.

**,RETCODE=***retcode*

Use this output parameter to specify a field to contain the return code. (The return code is also returned in register 15.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the return code when the request has completed.

**,RSNCODE=***rsncode*

Use this output parameter to specify a field to contain the reason code returned, if applicable. (The reason code is also returned in register 0.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the reason code (if any) when the request has completed.

**,VECTORINDEX=***vectorindex*

Use this input parameter to specify an index into your local cache vector to be associated with the data item specified by NAME, to be disassociated with the data item specified by OLDNAME, or both. The vector index identifies a vector entry that cache services will use to indicate both your interest in the data item and the validity of the copy of the data item in your local cache buffer.

If the vector index you specify is already associated with another data item, you must disassociate the vector index from the old name by specifying OLDNAME before the vector index can be associated with the data item specified by NAME.

On CASTOUT\_DATA requests, for structures allocated in a coupling facility of CFLEVEL=5 or higher, VECTORINDEX is required when REGUSER=YES or OLDNAME is specified. For structures allocated in a coupling facility of CFLEVEL=4 or lower, VECTORINDEX is required when REGUSER=YES is specified.



**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 4-byte field that contains the vector index for NAME or OLDNAME.

## ABEND Codes

Abend X'026' (See *z/OS MVS Programming: Sysplex Services Guide* for more information on this abend.)

## Return and Reason Codes

When the system returns control to your program:

- GPR 15 (and RETCODE, if specified) contains a return code.
- If the return code is not zero, GPR 0 (and RSNCODE, if specified) contains a reason code.

**Note:** The return and reason codes will also be put into the answer area (mapped by IXLYCAA), if there is one, when the request completes.

The IXLYCON macro provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

<b>0</b>	IXLRETCODEOK
<b>4</b>	IXLRETCODEWARNING
<b>8</b>	IXLRETCODEPARMERROR
<b>C</b>	IXLRETCODEENVERROR
<b>10</b>	IXLRETCODECOMPERROR

The following table contains hexadecimal return and reason codes, the equate symbols associated with each reason code, and the meaning and suggested action for each return and reason code.

Table 29. Return and Reason Codes for the IXLCACHE REQUEST=CASTOUT_DATA Macro		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0	None.	<p><b>Meaning:</b> If you specified a MODE value of ASYNCECB, ASYNCEXIT, or ASYNCTOKEN, the request has been successfully initiated. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed. For any other MODE value, the request has successfully completed.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=ASYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=ASYNCEXIT, the connection's complete exit will be given control when the request has completed.</li> <li>• If you specified MODE=ASYNCTOKEN, use REQTOKEN when calling IXLFCOMP to determine when the request has completed.</li> </ul>

Table 29. Return and Reason Codes for the IXLCACHE REQUEST=CASTOUT\_DATA Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	xxxx0402	<p><b>Equate Symbol:</b> IXLRSNCODEASYNCH</p> <p><b>Meaning:</b> The request could not be processed synchronously. It will be processed asynchronously. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed.</p> <ul style="list-style-type: none"> <li>• If you specified MODE=SYNCECB, the ECB specified in REQECB will be posted when the request has finished.</li> <li>• If you specified MODE=SYNCEXIT, the connection's complete exit will be given control when the request has finished.</li> <li>• If you specified MODE=SYNCTOKEN, the REQTOKEN contains an asynchronous request token that may be used on the IXLFCOMP macro to determine when the request has completed.</li> </ul> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=SYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=SYNCEXIT, the connection's completion exit will be given control when the request has completed.</li> <li>• If you specified MODE=SYNCTOKEN, use REQTOKEN when calling IXLFCOMP to determine when the request has completed.</li> </ul>
4	xxxx040C	<p><b>Equate Symbol:</b> IXLRSNCODENOADJUNCTDATA</p> <p><b>Meaning:</b> The request specified that adjunct data was to be read, but the structure does not support adjunct areas. No adjunct data was retrieved; however, entry data was returned in BUFLIST or BUFFER if requested.</p> <p>The following fields were returned in the answer area:</p> <ul style="list-style-type: none"> <li>• CAAELEMNUM</li> <li>• CAAINVLCVI</li> <li>• CAAINVLCVINUM</li> <li>• CAAPARITY</li> <li>• CAAVERSION (for CFLEVEL=5 or higher)</li> </ul> <p><b>Action:</b> No action is necessary. However, if you expected this structure to have adjunct data, you may want to:</p> <ul style="list-style-type: none"> <li>• Verify the CONTOKEN is from the correct invocation of IXLCONN.</li> <li>• Check the IXLCONN macro that defined the structure to be sure adjunct data was specified.</li> </ul>

Table 29. Return and Reason Codes for the IXLCACHE REQUEST=CASTOUT\_DATA Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	xxxx040D	<p><b>Equate Symbol:</b> IXLRNCODEBADREADADJDATA</p> <p><b>Meaning:</b> Program error. The request specified that adjunct data was to be read, but the specified storage area for adjunct data (ADJAREA) is not addressable. The adjunct data was not retrieved; however, the entry data was returned in BUFLIST or BUFFER, if requested.</p> <p>The following fields are returned in the answer area:</p> <ul style="list-style-type: none"> <li>• CAAELEMNUM</li> <li>• CAAINVLCVI</li> <li>• CAAINVLCVINUM</li> <li>• CAAPARITY</li> </ul> <p>For CFLEVEL=4 and higher coupling facilities, the following additional answer area field is returned:</p> <ul style="list-style-type: none"> <li>• CAADATAACACHED</li> </ul> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the ADJAREA address.</li> <li>• ADJAREA must be addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are calling IXLCACHE while disabled, ADJAREA must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLCACHE in AR-mode and you specified the ADJAREA address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are calling IXLCACHE in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLCACHE macro.</li> </ul> <p>Correct the address specified by ADJAREA and rerun the request asking for adjunct data only.</p>
8	xxxx0801	<p><b>Equate Symbol:</b> IXLRNCODEBADPARMLIST</p> <p><b>Meaning:</b> The parameter list (mfctrl) for this request is not addressable. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the parameter list address.</li> <li>• The parameter list must be addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are issuing this macro while disabled, the parameter list must reside in either page-fixed or DREF storage.</li> <li>• If you are issuing this macro in AR-mode and you specified the parameter list address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are issuing this macro in AR-mode, SYSSTATE ASCENV=AR must be issued before this macro.</li> </ul>

Table 29. Return and Reason Codes for the IXLCACHE REQUEST=CASTOUT\_DATA Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0804	<p><b>Equate Symbol:</b> IXLRSNCODEBADVERSION#</p> <p><b>Meaning:</b> The version number in the macro parameter list is not compatible with the level of XES currently being used. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that your program did not overlay the parameter list storage.</li> <li>• Verify that your program was assembled with the correct macro library for the release of MVS on which your program is running.</li> <li>• Verify that your program is running on an MVS system that supports the version of the macro you are using.</li> </ul>

Table 29. Return and Reason Codes for the IXLCACHE REQUEST=CASTOUT\_DATA Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx080A	<p><b>Equate Symbol:</b> IXLSRNCODEBADCONTOKEN</p> <p><b>Meaning:</b> Program error. The specified connect token (CONTOKEN) was not valid for one of the following reasons:</p> <ol style="list-style-type: none"> <li>1. The user with the connection identifier represented by the token has disconnected from the structure.</li> <li>2. The connector's task (the task that issued IXLCONN) ended.</li> <li>3. The specified token is not the token that was returned from IXLCONN.</li> <li>4. The request was issued from an address space other than the address space in which IXLCONN was issued.</li> <li>5. The connect token was invalidated during rebuild.</li> <li>6. The connect token was invalidated by XES.</li> </ol> <p><b>Note:</b> The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Verify that the CONTOKEN value specified was valid and for the correct structure. The numbers on the following actions correspond to the numbers above.</p> <ol style="list-style-type: none"> <li>1. Discontinue use. Perform recovery and cleanup for the structure and check protocol for use of structure to determine why you did not know of the disconnection.</li> <li>2. Discontinue use. Perform recovery and cleanup for the structure and check protocol for use of structure to determine why you did not know that the task ended.</li> <li>3. Issue IXLCONN to connect to the structure. The contoken is returned in the CONACONTOKEN field of the answer area specified on the IXLCONN request.</li> <li>4. Issue your request from the same address space the IXLCONN was issued.</li> <li>5. Participate in the rebuild. When it is complete, try again.</li> <li>6. Discontinue use. Perform recovery and cleanup for the structure and check protocol for use of structure.</li> </ol> <p>You may want to issue IXCQUERY to get more information about the structure.</p>
8	xxxx0819	<p><b>Equate Symbol:</b> IXLSRNCODEBADVECTOROP</p> <p><b>Meaning:</b> The VECTORINDEX specified is not valid. Request processing was suppressed.</p> <p><b>Action:</b> Specify a vector index that is within the current range of the number of vector entries for the local vector requested for this structure. The number of vector entries is determined by the VECTORLEN parameter specified on the IXLCONN macro and may be changed by issuing the IXLVECTR macro.</p>

Table 29. Return and Reason Codes for the IXLCACHE REQUEST=CASTOUT\_DATA Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0824	<p><b>Equate Symbol:</b> IXLRSNCODEWRONGSTRTYPE</p> <p><b>Meaning:</b> The connect token specified by CONTOKEN is not to a cache structure.</p> <p><b>Action:</b> Verify the connect token for this cache structure.</p> <p><b>Note:</b> The connect token can be found in the IXLCONN answer area. Verify the value specified on the TYPE= parameter of the IXLCONN request for this structure.</p>
8	xxxx0825	<p><b>Equate Symbol:</b> IXLRSNCODENOENTRY</p> <p><b>Meaning:</b> Program error. The request failed because the entry designated by NAME is not in the cache structure.</p> <p><b>Action:</b></p> <p>If this is unexpected, try the following:</p> <ul style="list-style-type: none"> <li>• Verify that NAME is uncorrupted.</li> <li>• Verify that the CONTOKEN is for the correct structure.</li> <li>• Issue IXLCACHE REQUEST=READ_COCLASS to get the names of all the data items in this cast-out class.</li> <li>• Issue IXLCACHE REQUEST=READ_DIRINFO to get the cast-out class this name belongs in.</li> </ul>
8	xxxx0827	<p><b>Equate Symbol:</b> IXLRSNCODECOLOCKHELD</p> <p><b>Meaning:</b> Program error. The cast-out lock for NAME is already held. This request did not lock the entry for cast-out. No data was returned in BUFFER, BUFLIST, or ADJAREA.</p> <p><b>Action:</b> A cast-out is already in progress for this data. The answer area (ANSAREA) contains the connection ID and the process ID in the CAACOLCKVAL field of the user/process that currently holds the cast-out lock. The state of the cast-out lock is returned in the CAACOLCKSTATE field. Check these values to determine if a problem exists in your code.</p>

Table 29. Return and Reason Codes for the IXLCACHE REQUEST=CASTOUT\_DATA Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0828	<p><b>Equate Symbol:</b> IXLSRNCODECOUNCHANGED</p> <p><b>Meaning:</b> Program error. The entry specified for cast-out does not contain changed data. The entry was not cast out, and no data was returned in BUFFER, BUFLIST, or ADJAREA.</p> <p><b>Action:</b> A cast-out may not be done on unchanged data. (Unchanged indicates that the copy in the coupling facility cache structure is the same as the permanent storage copy.) Determine why you thought this entry should be cast-out (a WRITE_DATA with CHANGED=YES must have been done to indicate changed status).</p> <p>You could try one of the following to get more information about the cast-out class:</p> <ul style="list-style-type: none"> <li>• Issue IXLCACHE REQUEST=READ_COCLASS to get the names of all the data items in the cast-out class.</li> <li>• Issue IXLCACHE REQUEST=READ_DIRINFO to more information about changed data items.</li> </ul>
8	xxxx0833	<p><b>Equate Symbol:</b> IXLSRNCODEBADPGBLATTR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list is specified as being pageable (PAGEABLE=YES) but is not.</p> <p><b>Action:</b> Change the buffer area(s) to pageable storage, or specify PAGEABLE=NO. See the PAGEABLE parameter description for specification instructions.</p>
8	xxxx0834	<p><b>Equate Symbol:</b> IXLSRNCODEBADNONPGBLATTR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list is specified as being nonpageable (PAGEABLE=NO) but is either pageable or not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The type of storage in the buffer area corresponds to the value you specified on the PAGEABLE parameter. See the PAGEABLE parameter description for specification instructions.</li> <li>• The correct buffer address was used.</li> <li>• The buffer area(s) were not previously freed.</li> <li>• If BUFLIST was specified and your program is running in AR mode, ensure that: <ul style="list-style-type: none"> <li>– The BUFALET specification is correct.</li> <li>– You specified SYSSTATE ASCENV=AR before issuing the macro.</li> </ul> </li> <li>• If the caller is running in AR-mode and the BUFFER or BUFLIST parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> </ul>

Table 29. Return and Reason Codes for the IXLCACHE REQUEST=CASTOUT\_DATA Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0835	<p><b>Equate Symbol:</b> IXLRNCODEBADDATAADDR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list, is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The correct buffer address was used for BUFFER or for a buffer within the BUFLIST.</li> <li>• The buffer area(s) were not previously freed.</li> <li>• The buffer area(s) were allocated in a storage key that matches the key specified using the BUFSTGKEY parameter, or, if BUFSTGKEY is omitted, the caller's PSW key.</li> <li>• If the caller is running in AR-mode and the BUFFER or BUFLIST parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If the caller is running in AR-mode, SYSSTATE ASCENV=AR must be specified before issuing this macro.</li> <li>• If BUFLIST was specified and your program is running in AR mode the BUFALET specification is correct.</li> </ul>
8	xxxx0836	<p><b>Equate Symbol:</b> IXLRNCODEBADREALADDR</p> <p><b>Meaning:</b> Program error. Real storage addresses were provided in a BUFLIST list, but one of the buffers is not addressable in central storage.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that BUFADDRTYPE was specified as you intended.</li> <li>• Ensure that the real buffer addresses specified by BUFLIST are valid.</li> </ul>
8	xxxx0838	<p><b>Equate Symbol:</b> IXLRNCODEBADANSAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by ANSAREA is not addressable.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the ANSAREA address.</li> <li>• ANSAREA must be addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are calling IXLCACHE while disabled, ANSAREA must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLCACHE in AR-mode and you specified the ANSAREA address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are calling IXLCACHE in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLCACHE macro.</li> </ul>



Table 29. Return and Reason Codes for the IXLCACHE REQUEST=CASTOUT\_DATA Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0839	<p><b>Equate Symbol:</b> IXLRNCODEBADREQTOKENAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by REQTOKEN is not addressable.</p> <p><b>Action:</b> Ensure that the request token area specified by REQTOKEN is valid:</p> <ul style="list-style-type: none"> <li>• Verify the REQTOKEN address.</li> <li>• If you are calling IXLCACHE while disabled, REQTOKEN must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLCACHE in AR-mode and you specified the REQTOKEN address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are calling IXLCACHE in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLCACHE macro.</li> </ul>
8	xxxx083D	<p><b>Equate Symbol:</b> IXLRNCODEBADANSLEN</p> <p><b>Meaning:</b> Program error. The length of the answer area, as specified by ANSLEN, is not sufficient to contain answer area information. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Increase the size of the answer area provided for the request, and rerun your program. Depending on the macro version number, specify the length of the answer area as follows:</p> <ul style="list-style-type: none"> <li>• When the connection specified ASYNCXI=1 on the IXLCONN invocation and AXIOVERRIDE=0 was specified or defaulted to for the IXLCACHE request, the answer area length must be a minimum value of CAALEVEL2LEN</li> <li>• When the value of the macro version number (PLISTVER) is 4 or more, the minimum answer area length is CAALEVEL1LEN.</li> <li>• When the value of the macro version number (PLISTVER) is 0-3, the minimum answer area length is CAALEVEL0LEN.</li> </ul>
8	xxxx0851	<p><b>Equate Symbol:</b> IXLRNCODENOSUSPENDISABLE</p> <p><b>Meaning:</b> Program error. The request failed because MODE=SYNCSUSPEND was specified, but the caller is disabled and cannot be suspended.</p> <p><b>Action:</b> Either specify another MODE value or become enabled (release the CPU lock); then reissue the request.</p>
8	xxxx0864	<p><b>Equate Symbol:</b> IXLRNCODEBADBUFSIZE</p> <p><b>Meaning:</b> Program Error. The size of the BUFLIST areas or BUFFER area is not large enough to contain the data being read. The number of elements in the retrieved entry is returned in the answer area in the field CAAELEMNUM.</p> <p><b>Action:</b> If more space is available, specify a larger buffer size and reissue the request.</p>

Table 29. Return and Reason Codes for the IXLCACHE REQUEST=CASTOUT\_DATA Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0865	<p><b>Equate Symbol:</b> IXLRNCODEBADBUFSPEC</p> <p><b>Meaning:</b> Program error. There is an error in the buffer specification.</p> <p><b>Action:</b> Check the following:</p> <ul style="list-style-type: none"> <li>• If BUFLIST was specified, check the requirements for BUFLIST, BUFNUM, and BUFINCRNUM.</li> <li>• If BUFFER was specified, check the requirements for BUFFER and BUFSIZE.</li> <li>• Buffer pointer(s) in BUFLIST.</li> <li>• Buffer boundaries.</li> </ul>
8	xxxx0866	<p><b>Equate Symbol:</b> IXLRNCODEBADBUFKEY</p> <p><b>Meaning:</b> Program error. The buffer storage key specified by BUFSTGKEY is incorrect or BUFSTGKEY was not specified and the PSW key does not match the key of the buffers.</p> <p>The data cannot be stored in the specified buffer area.</p> <p><b>Action:</b> Check the following:</p> <ul style="list-style-type: none"> <li>• Determine if the key of the storage being used for the buffers is different from the PSW key.</li> <li>• If BUFSTGKEY was specified, verify that the key was put in the correct bits (see the explanation of this parameter for more information).</li> <li>• If you are calling IXLCACHE in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLCACHE macro.</li> </ul>
8	xxxx0867	<p><b>Equate Symbol:</b> IXLRNCODEBADBUFLIST</p> <p><b>Meaning:</b> Program error. The 128-byte storage area specified by BUFLIST is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The correct BUFLIST address was used.</li> <li>• The BUFLIST area was not previously freed.</li> <li>• If the caller is running in AR-mode and the BUFLIST parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If the caller is disabled, then the BUFLIST must reside in either nonpageable or disabled reference storage.</li> <li>• If your program is running in AR mode, ensure that you specified SYSSTATE ASCENV=AR before issuing the macro.</li> </ul>
8	xxxx08AD	<p><b>Equate Symbol:</b> IXLRNCODEBADHIGHSHAREDVIRT</p> <p><b>Meaning:</b> Program error. The request specified a high shared virtual storage area (above 2GB).</p> <p><b>Action:</b> None required.</p>

Table 29. Return and Reason Codes for the IXLCACHE REQUEST=CASTOUT\_DATA Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C06	<p><b>Equate Symbol:</b> IXLRNOCODENOCONE</p> <p><b>Meaning:</b> Environmental error. No connectivity to the cache structure exists. This may occur due to operator commands such as VARY PATH,OFFLINE or CONFIG CHP,OFFLINE or hardware errors such as facility or path failures. The CONTOKEN is invalidated. The request fails.</p> <p><b>Action:</b> Either disconnect from the structure (using IXLDISC) or initiate a rebuild of the structure, if allowed (using IXLREBLD).</p>
C	xxxx0C13	<p><b>Equate Symbol:</b> IXLRNOCODEREQPURGED</p> <p><b>Meaning:</b> Environmental error. The request was purged prior to completion of the request. Possible reasons include:</p> <ul style="list-style-type: none"> <li>• The connector failed.</li> <li>• The connector disconnected.</li> <li>• The requestor failed.</li> <li>• The request was purged by IXLPURGE.</li> <li>• Requests were purged when the connector provided an IXLEERSP response for the Rebuild Quiesce, Rebuild Stop, or Rebuild Cleanup event.</li> <li>• The connector invoked IXLREBLD REQUEST=COMPLETE.</li> <li>• The secondary address space was no longer valid.</li> </ul> <p><b>Action:</b> None if this is expected. Otherwise, determine why the connector failed.</p>
C	xxxx0C14	<p><b>Equate Symbol:</b> IXLRNOCODESTATUSUNKNOWN</p> <p><b>Meaning:</b> Environmental error. The IXLCACHE request has completed, but the final disposition of the request cannot be determined. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Verify the validity of your data by comparing the expected results with what is in the coupling facility.</p>
C	xxxx0C25	<p><b>Equate Symbol:</b> IXLRNOCODESTRFAILURE</p> <p><b>Meaning:</b> Environmental error. The cache structure failed prior to completion of the request.</p> <p><b>Action:</b> Attempt to rebuild the structure using IXLREBLD, or disconnect from the structure using IXLDISC.</p>
C	xxxx0CA0	<p><b>Equate Symbol:</b> IXLRNOCODEQUIESCEDSUSPENDFAIL</p> <p><b>Meaning:</b> Environmental error. The request is failed because the structure is quiesced for a system-managed process and SUSPEND=FAIL is specified on the IXLCONN.</p> <p><b>Action:</b> None, if this is expected.</p>

Table 29. Return and Reason Codes for the IXLCACHE REQUEST=CASTOUT\_DATA Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxxFFFF	<p><b>Equate Symbol:</b> IXLRSNCODENOTAVAILABLE</p> <p><b>Meaning:</b> Environmental error. XES functions are not available. The coupling facility hardware might not be present.</p> <p><b>Action:</b> XES may not be used when XCF is running local mode. XES requires that the coupling facility be installed.</p>
10	xxxx10xx	<p><b>Equate Symbol:</b> IXLRSNCODECOMPERROR</p> <p><b>Meaning:</b> System error. XES processing failure. The state of the structure and the disposition of the request are unpredictable. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Save the reason code information, and contact the IBM support center.</p>

## Chapter 32. IXLCACHE REQUEST=CASTOUT\_DATALIST

### Description

A CASTOUT\_DATALIST request allows you to read a set of data items from entries in the cache structure to the storage areas specified by BUFFER, BUFLIST, DEIBAREA, and/or ADJAREA. From these areas, the data can be written to permanent storage such as DASD. This is part of the cast-out process. The set of entries identified for castout processing is specified by CASTOUTLIST, which contains a list of up to 8 16-byte entry names.

The system references the entries in the CASTOUTLIST by an index into the list and processes them sequentially beginning with the entry specified by the first index (STARTINDEX) and ending with the entry identified by the last index (ENDINDEX). The entries are numbered starting with 1.

When you issue a CASTOUT\_DATALIST request:

- A cast-out lock is obtained for the set of entries specified in CASTOUTLIST. This lock protects the data item from further cast-out processing until the cast-out lock is released. (To release the cast-out lock, you must issue an UNLOCK\_CASTOUT request.)
- The change indicator in the directory entry for each entry is updated to indicate the data entry contains unchanged data. (Note that the data is considered changed until the cast-out lock is released.)

When processing is initiated for a CASTOUT\_DATALIST request, the system moves the DEIB, which contains directory information for the entry, to DEIBAREA and adjunct data, if present in the structure, to ADJAREA for the first processed name in the CASTOUTLIST pointed to by STARTINDEX. The data area from the first processed entry is moved to the output BUFFER or BUFLIST. Subsequent processing of entries in the CASTOUTLIST move the DEIB, adjunct data if it exists, and the data area from the processed entry to the output BUFFER or BUFLIST. This continues until the end of the CASTOUTLIST pointed to by ENDINDEX is reached or until processing is discontinued.

Processing for a CASTOUT\_DATALIST request can be discontinued in the following instances:

- Another connection or process currently holds the cast-out lock for the entry being processed. The entry is not processed and ANSAREA contains:
  - The index of the name in the CASTOUTLIST
  - The value of the cast-out lock
  - The value of the cast-out lock state

All prior entries in the CASTOUTLIST will have been processed.

- Entry data is not cached, or the cached entry data is not changed. The entry is not processed and ANSAREA contains:
  - The index of the name in the CASTOUTLIST
  - The changed data indicator
  - The data-cached indicator

All prior entries in the CASTOUTLIST will have been processed.

- The entry name being processed in the CASTOUTLIST is not in the directory. The entry is not processed and ANSAREA contains:
  - The index of the name in the CASTOUTLIST

All prior entries in the CASTOUTLIST will have been processed.

- The size of the BUFFER or BUFLIST is not large enough to contain the data area for the entry specified by STARTINDEX. The entry is not processed and ANSAREA contains:
  - The number of elements in the indexed entryNo entries are processed.
- The remaining space in the BUFFER or BUFLIST is not large enough to contain the data area for the current entry in the CASTOUTLIST. The entry is not processed and ANSAREA contains:
  - The index of the entry in CASTOUTLIST being processed
  - The number of elements in the desired entryAll prior entries in the CASTOUTLIST will have been processed.

A CASTOUT\_DATALIST can complete prematurely because the request exceeds the time-out criteria for the coupling facility. (Time-out criteria is model-dependent.) When a request completes prematurely, the system returns an index value (CAACDLINDEX) in the answer area that you can use to restart the CASTOUT\_DATALIST request. The index value when a CASTOUT\_DATALIST completes prematurely is the index of the first unprocessed entry in the CASTOUTLIST.

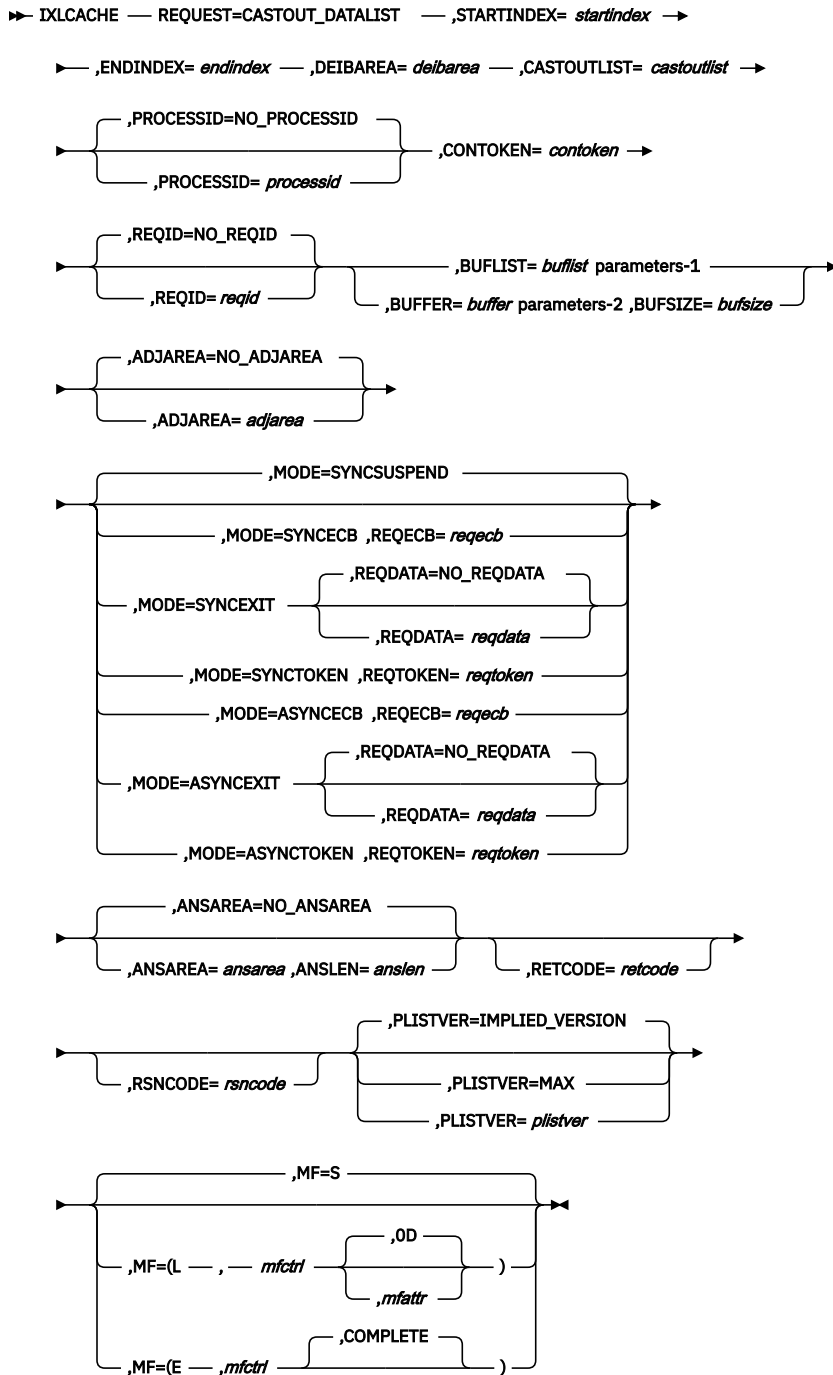
A CASTOUT\_DATALIST request can be issued only for cache structures allocated in a coupling facility of CFLEVEL=12 or higher. CASTOUT\_DATALIST requests issued for a cache structure allocated in a coupling facility of CFLEVEL=0 through 11 will fail.

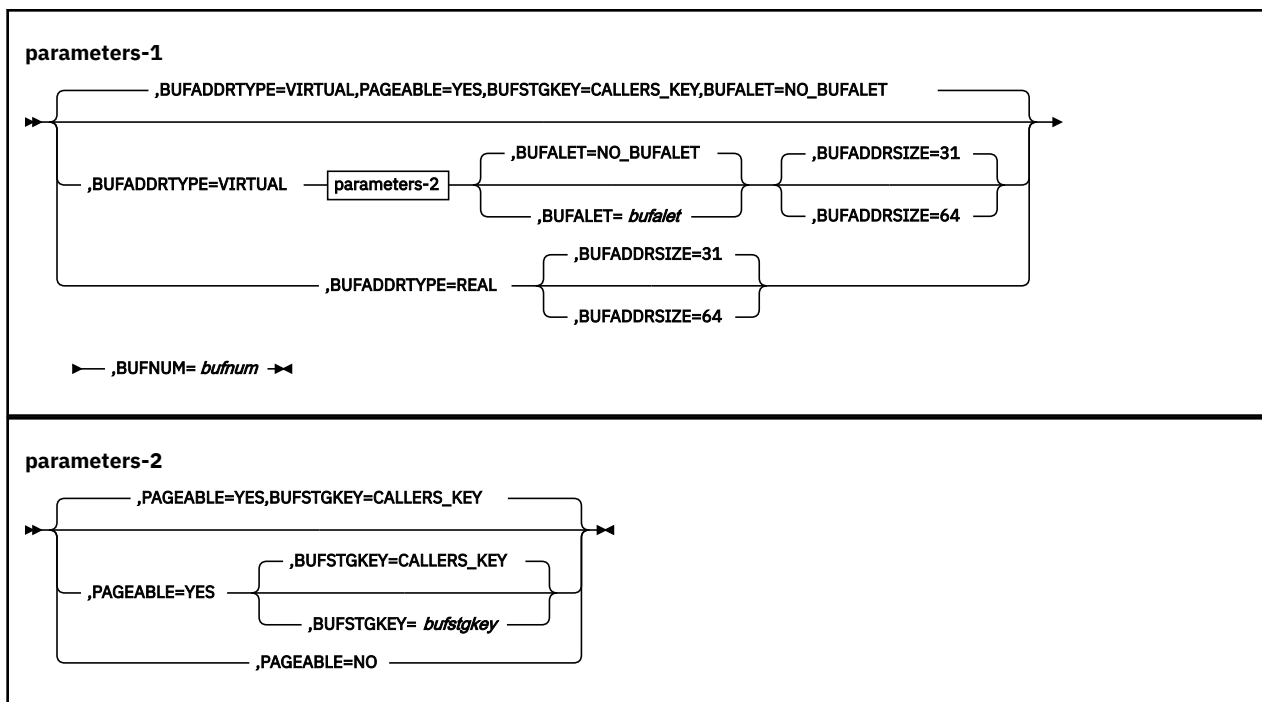
## Syntax Diagram

---

The syntax diagram for IXLCACHE REQUEST=CASTOUT\_DATALIST is as follows:

## main diagram





**Note:** If you specify `MODE=SYNCTOKEN` or `MODE=ASYNCTOKEN`, then you must also specify `ANSAREA=ansarea`, `ANSLEN=anslen`.

## Parameter Descriptions

The parameter descriptions for `REQUEST=CASTOUT_DATALIST` are listed in alphabetical order. Default values are underlined:

### **REQUEST=CASTOUT\_DATALIST**

Use this input parameter to:

- Specify that a cast-out lock be obtained for the set of data items identified by `CASTOUTLIST`, and that the named data items be read into the storage areas specified `BUFFER` or `BUFLIST`, `DEIBAREA`, and/or `ADJAREA`.
- Update the directory entry change bit for each data item to indicate that each entry contains unchanged subsystem data.

### **,ADJAREA=NO\_ADJAREA**

### **,ADJAREA=adjarea**

Use this output parameter to specify a storage area to contain the adjunct data that was read from the first processed entry.

If the structure supports adjunct data and `ADJAREA` is not specified, or if `ADJAREA=NO_ADJAREA` is specified, then no adjunct data is returned for the first entry processed.

If the structure does not support adjunct data and `ADJAREA` is specified, then entry data is returned, but no adjunct data is returned for every entry processed. `CAAADJAREAVALID`, which is returned in `ANSAREA`, will indicate that adjunct data did not exist.

(Adjunct areas for a structure are established through the `IXLCONN` macro.)

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 64-byte virtual storage area that the adjunct data will be read into.

### **,ANSAREA=NO\_ANSAREA**

### **,ANSAREA=ansarea**

Use this output parameter to specify an answer area to contain information returned from the request. The format of the answer area is described by the `IXLYCAA` mapping macro.



**To Code:** Specify the RS-type name or address (using a register from 2 - 12) of an area (with a length of ANSLEN) where the information returned from the request will go.

**,ANSLEN=*anslen***

Use this inppaameter to specify the size of the storage area specified by ANSAREA.

Use either CAALEVEL0LEN, CAALEVEL1LEN or CAALEVEL2LEN of the IXLYCAA mapping macro to determine the minimum size of the answer area. The answer area length must be at least large enough to accommodate the level of the IXLYCAA mapping appropriate to the requested function.

- When the connection specified ASYNCXI=1 on the IXLCONN invocation and AXIOVERRIDE=0 was specified or defaulted to for the IXLCACHE request, the answer area length is a required parameter and must be a minimum value of CAALEVEL2LEN to contain a returned asynchronous cross-invalidation sequence number (CAAASYNXISEQNUM).
- When the value of PLISTVER is 4 or above, the minimum answer area length is CAALEVEL1LEN.
- When the value of PLISTVER is 0 - 3, the minimum answer area length is CAALEVEL0LEN.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12 2 to ) of a 2-byte field that contains the length of the answer area (ANSAREA).

**,BUFADDRSIZE=31**

**,BUFADDRSIZE=64**

Use this input parameter to specify whether a 31-bit or a 64-bit address is specified by a BUFLIST entry.

**31**

The entry in BUFLIST is 31 bits in size.

**64**

The entry in BUFLIST is 64 bits in size.

**,BUFADDRTYPE=VIRTUAL**

**,BUFADDRTYPE=REAL**

Use this input parameter to specify whether the buffer addresses specified in the BUFLIST list are virtual storage or real storage addresses.

**VIRTUAL**

The buffer addresses are virtual storage addresses. The virtual storage can be pageable or nonpageable. See the PAGEABLE parameter for information about managing storage binds when specifying virtual storage addresses.

**REAL**

The buffer addresses are real storage addresses.

It is the caller's responsibility to manage the binds between the data buffer virtual storage and the real storage addresses provided. The caller must ensure that the data buffer virtual storage remains bound to the real storage addresses provided until the request completes.

**,BUFALET=NO\_BUFALET**

**,BUFALET=*bufalet***

Use this input parameter to specify an access list entry token (ALET) to be used in referencing all of the buffers specified by BUFLIST.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 4-byte field that contains the ALET.

**,BUFFER=*buffer***

Use this output parameter to specify a buffer area to hold the data area from the first processed name in the CASTOUTLIST followed by the DEIB, adjunct data if it exists, and the data area from the second processed name in the CASTOUTLIST. This will continue until the end of the CASTOUTLIST pointed to by ENDINDEX is reached or until the request completes prematurely for some other reason.

Only 31-bit addressable virtual storage areas (below 2GB) are supported by the BUFFER specification. High virtual storage areas (above 2GB) can only be specified via the BUFLIST specification.

You must ensure that the storage area specified by BUFFER:

- Is a multiple of 4096 bytes.
- Is less than or equal to 65536 bytes.
- Starts on a 4096-byte boundary.

See the BUFSIZE parameter description for defining the size of the buffer.

See *z/OS MVS Programming: Sysplex Services Guide* for more information on buffers.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an area (with a length of BUFSIZE) to receive the entries requested.

#### **BUFLIST=buflist**

Use this output parameter to specify a list of buffers to hold the data area from the first processed name in the CASTOUTLIST followed by the DEIB, adjunct data if it exists, and the data area from the second processed name in the CASTOUTLIST. This will continue until the end of the CASTOUTLIST pointed to by ENDINDEX is reached or until the request completes prematurely for some other reason. BUFLIST specifies a 128-byte storage area that consists of a list of 0 to 16 buffer addresses.

Either 31-bit addressable (below 2GB) or 64-bit addressable (above 2GB) real or virtual storage areas are supported for the BUFLIST specification, depending on the specification for the BUFADDRTYPE and BUFADDRSIZE keywords. However, pageable high shared virtual storage areas (above 2GB) may not be used.

The format of the list is a set of 8-byte elements. The BUFADDRSIZE keyword denotes whether four or eight bytes of the element are used.

- If BUFADDRSIZE=31 is specified, then the first four bytes of each element are reserved space and the last four bytes contain the address of the buffer.
- If BUFADDRSIZE=64 is specified, then the full eight bytes specify the address of the buffer.

#### **The BUFLIST buffers must:**

- Reside in the same address space or data space.
- Be 4096 bytes.
- Start on a 4096-byte boundary.

**Note:** The buffers do not have to be contiguous in storage. Cache services treat BUFLIST buffers as a single buffer even if the buffers are not contiguous.

See the BUFNUM parameter to specify the number of buffers in the buffer list.

See *z/OS MVS Programming: Sysplex Services Guide* for more information on buffers.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 128-byte field that contains a list of buffer addresses.

#### **,BUFNUM=bufnum**

Use this input parameter to specify the number of buffers in the BUFLIST list. Valid BUFNUM values are 0 - 16. A value of zero indicates that no data is to be read into the buffers.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 1-byte field that contains the number of buffers (0 - 16) in the list (BUFLIST).

#### **,BUFSIZE=bufsize**

Use this input parameter to specify the size of the BUFFER area. See the BUFFER parameter description for valid buffer sizes.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a fullword field that contains the size of the buffer (BUFFER) in bytes.

**,BUFSTGKEY=CALLERS\_KEY****,BUFSTGKEY=bufstgkey**

Use this input parameter to specify a storage key that you define and use when referencing the buffers specified by BUFLIST or the buffer that is specified by BUFFER.

If you do not specify BUFSTGKEY, or if you specify BUFSTGKEY=CALLERS\_KEY, all references to one or more buffers are performed by using the caller's PSW key.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of an 8-bit field that contains the storage key in the format B'kkkkxxxx', where kkkk is the key and xxxx is ignored.

**,CASTOUTLIST=castoutlist**

Use this input parameter to specify a 128-byte area that contains a list of up to 8 16-byte entry names to be locked for castout processing.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 128-byte field containing a list of entry names.

**,CONTO=conken**

Use this input parameter to specify the connect token that was returned by the IXLCONN service in the IXLCONN answer area, which is mapped which is by IXLYCONA. The connect token uniquely identifies your connection to the cache structure, and must be specified on each IXLCACHE invocation.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field that contains the connect token.

**,DEIBAREA=deibarea**

Use this output parameter to specify an output area to contain the DEIB from the first entry processed in CASTOUTLIST, as specified by STARTINDEX. The DEIBs for the rest of the entries will be returned in the data buffer specified by BUFFER or BUFLIST.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 128-byte field to contain the DEIB from the first entry processed.

**,ENDINDEX=endindex**

Use this input parameter to specify the ending index for the last entry in CASTOUTLIST to be processed. The index value must be greater than or equal to the value specified for STARTINDEX, but less than or equal to 8.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a halfword field containing the ending index value.

**,MF=S****,MF=(L,mfctrl)****,MF=(L,mfctrl,mfattr)****,MF=(L,mfctrl,0D)****,MF=(M,mfctrl)****,MF=(M,mfctrl,COMPLETE)****,MF=(M,mfctrl,NOCHECK)****,MF=(E,mfctrl)****,MF=(E,mfctrl,COMPLETE)****,MF=(E,mfctrl,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

**,mfctrl**

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

**,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

**,COMPLETE**

**,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

**COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if SMILE=*var* were an optional parameter and the default is SMILE=NO\_SMILE then it would not be documented. However, if the default was SMILE=:), then it would be documented because a value would be the default.

**NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

**,MODE=SYNCSUSPEND**

**,MODE=SYNCECB**

**,MODE=SYNCEXIT**

**,MODE=SYNCTOKEN**

**,MODE=ASYNCECB**

**,MODE=ASYNCEXIT**

**,MODE=ASYNCTOKEN**

Use this input parameter to specify:

- Whether the request is to be performed synchronously or asynchronously
- How you want to be notified of request completion if the request is processed asynchronously.

See *z/OS MVS Programming: Sysplex Services Guide* for more information on understanding synchronous and asynchronous cache operations.

**SYNCSUSPEND**

The request processes synchronously. If necessary, the request is suspended until it can complete synchronously. To use this option, your program must be enabled for I/O and external interrupts.

**SYNCECB**

The request processes synchronously if possible. If the request processes asynchronously, the ECB specified by REQECB is posted when the request completes.

**SYNCEXIT**

The request processes synchronously if possible. If the request processes asynchronously, your complete exit is given control when the request completes.

**SYNCTOKEN**

The request processes synchronously if possible. If the request processes asynchronously, an asynchronous request token is returned to the area specified by REQTOKEN. Use the returned request token on the IXLFCOMP macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=SYNCTOKEN is specified.

**ASYNCECB**

The request processes asynchronously. The ECB specified by REQECB is posted when the request completes.

**ASYNCEXIT**

The request processes asynchronously. Your complete exit is given control when the request completes.

**ASYNCTOKEN**

The request processes asynchronously. An asynchronous request token is returned to the area specified by REQTOKEN. Use the returned request token on the IXLFCOMP macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=ASYNCTOKEN is specified.

**,PAGEABLE=YES****,PAGEABLE=NO**

Use this input parameter to identify whether the storage areas specified by BUFFER or BUFLIST are in pageable or potentially pageable storage.

**YES**

Specify this option to indicate that the BUFFER or BUFLIST buffers reside in pageable virtual storage. XES performs the required page fixing to fix the buffers in real storage while the cache or list request transfers data to or from the coupling facility.

This includes storage obtained from pageable subpools, disabled reference (DREF) storage, and may include storage that has the potential to become pageable during the processing of a request. (An example is address space storage owned by any swappable address space, for which a PGSER FIX has been successfully processed, but for which the owning address space gets swapped during processing of a cache or list request.) This does not include implicitly non-pageable storage (for example, storage obtained from non-pageable subpools).

The system takes responsibility for managing binds to central storage for the duration of the cache or list request, regardless of what address space owns the storage or whether the storage-owning address space is swappable or nonswappable. The storage can be owned by any address space.

High shared virtual storage areas (above 2GB) may not be used.

**NO**

Specify this option to indicate that the BUFFER or BUFLIST buffers reside in non-pageable virtual storage. XES does not page fix the buffers in real storage.

This includes implicitly non-pageable storage areas (for example, storage obtained from non-pageable subpools), and may include storage that has the potential to become pageable during the processing of a request (An example is address space storage owned by any swappable address space, for which a PGSER FIX has been successfully processed, but for which the owning address space gets swapped-out during processing of a cache or list request.)

The system takes responsibility for managing binds to central storage for the duration of the cache or list request, if and only if the non-pageable storage is owned by either the requester's address space or the connector's address space. If the storage is owned by any other address space, then the invoker is responsible for ensuring that the virtual storage remains non-pageable for the duration of the request (including the case in which the storage is owned by a swappable address space that is swapped during processing of an IXLCACHE or IXLLIST request). Subject to this consideration, the storage can be owned by any address space. See [z/OS MVS Programming: Sysplex Services Guide](#).

**,PLISTVER=IMPLIED\_VERSION****,PLISTVER=MAX****,PLISTVER=*plistver***

Use this input parameter to specify the version of the macro. See “[Understanding IXLCACHE Version Support](#)” on page 461 for a description of the options available with PLISTVER.

**,PROCESSID=NO\_PROCESSID****,PROCESSID=*processid***

Use this input parameter to specify a user-defined process identifier to be compared with the cast-out lock along with the connection identifier.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12 2 to 12) of a 1-byte field that contains the user-defined process identifier.

**,REQDATA=NO\_REQDATA****,REQDATA=*reqdata***

Use this input parameter with MODE=SYNCEXIT or MODE=ASYNCEXIT to pass any data you choose to the complete exit. The exit will get control only if the request is processed asynchronously.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the data to be passed to the complete exit.

**,REQECB=*reqecb***

Use this output parameter with either MODE=SYNCECB or MODE=ASYNCECB to specify the address of an ECB, which is to be posted when the request completes if the request was processed asynchronously.

Before coding REQECB, you must ensure that:

- You initialize the ECB before you issue the request.
- The ECB resides in either common storage or the home address space where IXLCONN was issued.
- Any tasks that wait for the ECB to be posted reside in the home address space where IXLCONN was issued.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that contains the address of the ECB to be posted when the request completes. The ECB must be aligned on a fullword boundary.

**,REQID=NO\_REQID****,REQID=*reqid***

Use this input parameter to specify a user-defined request identifier to be associated with the request. You can specify this request identifier on the IXPURGE macro to cancel a request that has not yet been processed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the user-defined request identifier.

**,REQTOKEN=*reqtoken***

Use this output parameter with either MODE=SYNCTOKEN or MODE=ASYNCTOKEN to specify the address of a storage area to receive the request token that is returned when the request will be processed asynchronously. This token, which uniquely identifies the request, must be used as input to the IXLFCOMP macro, which you use to determine if the request has completed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field where the system will put the request token.

**,RETCODE=*retcode***

Use this output parameter to specify a field to contain the return code. (The return code is also returned in register 15.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the return code when the request has completed.

**,RSNCODE=rsncode**

Use this output parameter to specify a field to contain the reason code returned, if applicable. (The reason code is also returned in register 0.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the reason code (if any) when the request has completed.

**,STARTINDEX=startindex**

Use this input parameter to specify the index into CASTOUTLIST for the first entry to be processed. Valid STARTINDEX values are from 1 to the value of ENDINDEX. The first name in CASTOUTLIST block has index number 1.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a halfword field containing the starting index value.

## ABEND Codes

---

Abend X'026' (See [z/OS MVS Programming: Sysplex Services Guide](#) for more information on this abend.)

## Return and Reason Codes

---

When the system returns control to your program:

- GPR 15 (and RETCODE, if specified) contains a return code.
- If the return code is not zero, GPR 0 (and RSNCODE, if specified) contains a reason code.

**Note:** The return and reason codes will also be put into the answer area (mapped by IXLYCAA), if there is one, when the request completes.

The IXLYCON macro provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

**0**

IXLRETCODEOK

**4**

IXLRETCODEWARNING

**8**

IXLRETCODEPARMERROR

**C**

IXLRETCODEENVERROR

**10**

IXLRETCODECOMPERROR

The following table contains hexadecimal return and reason codes, the equate symbols associated with each reason code, and the meaning and suggested action for each return and reason code.

Table 30. Return and Reason Codes for the IXLCACHE REQUEST=CASTOUT\_DATA LIST Macro

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0	None.	<p><b>Meaning:</b> If you specified a MODE value of ASYNCECB, ASYNCEXIT, or ASYNCTOKEN, the request has been successfully initiated. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed. For any other MODE value, the request has successfully completed.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=ASYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=ASYNCEXIT, the connection's complete exit will be given control when the request has completed.</li> <li>• If you specified MODE=ASYNCTOKEN, use REQTOKEN when calling IXLFCOMP to determine when the request has completed.</li> </ul>
4	xxxx0402	<p><b>Equate Symbol:</b> IXLRNCODEASYNCH</p> <p><b>Meaning:</b> The request could not be processed synchronously. It will be processed asynchronously. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed.</p> <ul style="list-style-type: none"> <li>• If you specified MODE=SYNCECB, the ECB specified in REQECB will be posted when the request has finished.</li> <li>• If you specified MODE=SYNCEXIT, the connection's complete exit will be given control when the request has finished.</li> <li>• If you specified MODE=SYNCTOKEN, the REQTOKEN contains an asynchronous request token that may be used on the IXLFCOMP macro to determine when the request has completed.</li> </ul> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=SYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=SYNCEXIT, the connection's completion exit will be given control when the request has completed.</li> <li>• If you specified MODE=SYNCTOKEN, use REQTOKEN when calling IXLFCOMP to determine when the request has completed.</li> </ul>



Table 30. Return and Reason Codes for the IXLCACHE REQUEST=CASTOUT\_DATALIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	xxxx0409	<p><b>Equate Symbol:</b> IXLRSCODETIMEOUT</p> <p><b>Meaning:</b> The request has completed prematurely because the model-dependent time-out criteria of the coupling facility has been exceeded. The index of the first unprocessed entry in the CASTOUTLIST has been returned in the answer area (field CAACDLINDEX). All prior entries have been processed.</p> <p>Be sure to process the information returned from this request before reissuing the request. The data returned from this request will be overwritten if you specify the same buffer address. Continue to reissue the request until the return code indicates that all processing has completed.</p> <p><b>Action:</b> To restart the request, update STARTINDEX with the value of CAACDLINDEX, the index of the next entry in CASTOUTLIST to be processed.</p> <p>For more information about premature request completion, see <a href="#">z/OS MVS Programming: Sysplex Services Guide</a>.</p>
4	xxxx040F	<p><b>Equate Symbol:</b> IXLRSCODEBUFFERFULL</p> <p><b>Meaning:</b> The request completed prematurely because the buffer specified by BUFFER, or the buffers specified by BUFLIST, is full. The index of the name in the CASTOUTLIST returned in the answer area (field CAACDLINDEX) contains the index of the entry currently being processed and the number of elements in the desired entry is also returned in the answer area (field CAAELEMNUM).</p> <p><b>Action:</b> If ADJAREA is specified, check the CAAADJAREAAVALID bit in the answer area to see if adjunct was returned. If ADJAREA was specified, check the CAAADJAREANONADDR bit to determine if the provided storage for the adjunct data is addressable. If the provided storage for the DEIBAREA is not addressable, then the CAADEIBAREANONADDR bit will be set.</p> <p>Be sure to process the information returned from this request before reissuing the request. The data returned from this request will be overwritten if you specify the same buffer address. Continue to reissue the request until the return code indicates that all processing has completed.</p> <p>For more information about premature request completion, see <a href="#">z/OS MVS Programming: Sysplex Services Guide</a>.</p>

Table 30. Return and Reason Codes for the IXLCACHE REQUEST=CASTOUT\_DATALIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0801	<p><b>Equate Symbol:</b> IXLRSNCODEBADPARMLIST</p> <p><b>Meaning:</b> The parameter list (mfctrl) for this request is not addressable. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the parameter list address.</li> <li>• The parameter list must be addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are issuing this macro while disabled, the parameter list must reside in either page-fixed or DREF storage.</li> <li>• If you are issuing this macro in AR-mode and you specified the parameter list address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are issuing this macro in AR-mode, SYSSTATE ASCENV=AR must be issued before this macro.</li> </ul>
8	xxxx0804	<p><b>Equate Symbol:</b> IXLRSNCODEBADVERSION#</p> <p><b>Meaning:</b> The version number in the macro parameter list is not compatible with the level of XES currently being used. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that your program did not overlay the parameter list storage.</li> <li>• Verify that your program was assembled with the correct macro library for the release of MVS on which your program is running.</li> <li>• Verify that your program is running on an MVS system that supports the version of the macro you are using.</li> </ul>

Table 30. Return and Reason Codes for the IXLCACHE REQUEST=CASTOUT\_DATALIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx080A	<p><b>Equate Symbol:</b> IXLRNCODEBADCONTOKEN</p> <p><b>Meaning:</b> Program error. The specified connect token (CONTOKEN) was not valid for one of the following reasons:</p> <ol style="list-style-type: none"> <li>1. The user with the connection identifier represented by the token has disconnected from the structure.</li> <li>2. The connector's task (the task that issued IXLCONN) ended.</li> <li>3. The specified token is not the token that was returned from IXLCONN.</li> <li>4. The request was issued from an address space other than the address space in which IXLCONN was issued.</li> <li>5. The connect token was invalidated during rebuild.</li> <li>6. The connect token was invalidated by XES.</li> </ol> <p><b>Note:</b> The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Verify that the CONTOKEN value specified was valid and for the correct structure. The numbers on the following actions correspond to the numbers above.</p> <ol style="list-style-type: none"> <li>1. Discontinue use. Perform recovery and cleanup for the structure and check protocol for use of structure to determine why you did not know of the disconnection.</li> <li>2. Discontinue use. Perform recovery and cleanup for the structure and check protocol for use of structure to determine why you did not know that the task ended.</li> <li>3. Issue IXLCONN to connect to the structure. The contoken is returned in the CONACONTOKEN field of the answer area specified on the IXLCONN request.</li> <li>4. Issue your request from the same address space the IXLCONN was issued.</li> <li>5. Participate in the rebuild. When it is complete, try again.</li> <li>6. Discontinue use. Perform recovery and cleanup for the structure and check protocol for use of structure.</li> </ol> <p>You may want to issue IXCQUERY to get more information about the structure.</p>
8	xxxx0824	<p><b>Equate Symbol:</b> IXLRNCODEWRONGSTRTYPE</p> <p><b>Meaning:</b> The connect token specified by CONTOKEN is not to a cache structure.</p> <p><b>Action:</b> Verify the connect token for this cache structure.</p> <p><b>Note:</b> The connect token can be found in the IXLCONN answer area. Verify the value specified on the TYPE= parameter of the IXLCONN request for this structure.</p>

Table 30. Return and Reason Codes for the IXLCACHE REQUEST=CASTOUT\_DATALIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0825	<p><b>Equate Symbol:</b> IXLRSNCODENOENTRY</p> <p><b>Meaning:</b> Program error. The request failed because a name element specified an entry name that did not exist in the cache structure. The index of the failing entry is returned and all prior names were processed.</p> <p><b>Action:</b></p> <p>If this is unexpected, try the following:</p> <ul style="list-style-type: none"> <li>• Verify that entry name is uncorrupted.</li> <li>• Verify that the CONTOKEN is for the correct structure.</li> </ul>
8	xxxx0827	<p><b>Equate Symbol:</b> IXLRSNCODECOLOCKHELD</p> <p><b>Meaning:</b> Program error. The cast-out lock for the entry name currently being processed in the CASTOUTLIST is already held. This request did not lock the entry for cast-out. The index of the name in the CASTOUTLIST, the value of the cast-out lock (CAACOLCKVAL), and the value of the cast-out lock state (CAACOLCKSTATE) are returned in the answer area. All prior entries in the CASTOUTLIST were processed.</p> <p><b>Action:</b> If ADJAREA was specified, check the CAAADJAREAVVALID bit in the answer area to see if adjunct was returned. If ADJAREA was specified, check the CAAADJAREANONADDR bit to determine if the provided storage for the adjunct data is addressable. If the provided storage for the DEIBAREA is not addressable, then the CAADDEIBAREANONADDR bit will be set.</p>
8	xxxx0828	<p><b>Equate Symbol:</b> IXLRSNCODECOUNCHANGED</p> <p><b>Meaning:</b> Program error. The entry specified for cast-out does not contain changed data. The entry was not cast out. The system returns the index of the name in the CASTOUTLIST, the changed indicator, and the data cached indicator in the answer area. All prior entries in the CASTOUTLIST were processed.</p> <p><b>Action:</b> If ADJAREA was specified, check the CAAADJAREAVVALID bit in the answer area to see if adjunct was returned. If ADJAREA was specified, check the CAAADJAREANONADDR bit to determine if the provided storage for the adjunct data is addressable. If the provided storage for the DEIBAREA is not addressable, then the CAADDEIBAREANONADDR bit will be set.</p>
8	xxxx082B	<p><b>Equate Symbol:</b> IXLRSNCODEBADIDINDEX</p> <p><b>Meaning:</b> Program error. The name index specified by either STARTINDEX or ENDINDEX is not valid. No elements have been processed.</p> <p><b>Action:</b> Ensure that STARTINDEX and ENDINDEX specify valid indexes into the name elements stored in CASTOUTLIST. The value of ENDINDEX must be greater than or equal to STARTINDEX and ENDINDEX must be less than or equal to 8.</p>

Table 30. Return and Reason Codes for the IXLCACHE REQUEST=CASTOUT\_DATALIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0833	<p><b>Equate Symbol:</b> IXLRSNCODEBADPGBLATTR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list is specified as being pageable (PAGEABLE=YES) but is not.</p> <p><b>Action:</b> Change the buffer area(s) to pageable storage, or specify PAGEABLE=NO. See the PAGEABLE parameter description for specification instructions.</p>
8	xxxx0834	<p><b>Equate Symbol:</b> IXLRSNCODEBADNONPGBLATTR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list is specified as being nonpageable (PAGEABLE=NO) but is either pageable or not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The type of storage in the buffer area corresponds to the value you specified on the PAGEABLE parameter. See the PAGEABLE parameter description for specification instructions.</li> <li>• The correct buffer address was used.</li> <li>• The buffer area(s) were not previously freed.</li> <li>• If BUFLIST was specified and your program is running in AR mode, ensure that: <ul style="list-style-type: none"> <li>– The BUFALET specification is correct.</li> <li>– You specified SYSSTATE ASCENV=AR before issuing the macro.</li> </ul> </li> <li>• If the caller is running in AR-mode and the BUFFER or BUFLIST parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> </ul>

Table 30. Return and Reason Codes for the IXLCACHE REQUEST=CASTOUT\_DATALIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0835	<p><b>Equate Symbol:</b> IXLRNCODEBADDATAADDR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list, is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The correct buffer address was used for BUFFER or for a buffer within the BUFLIST.</li> <li>• The buffer area(s) were not previously freed.</li> <li>• The buffer area(s) were allocated in a storage key that matches the key specified using the BUFSTGKEY parameter, or, if BUFSTGKEY is omitted, the caller's PSW key.</li> <li>• If the caller is running in AR-mode and the BUFFER or BUFLIST parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If the caller is running in AR-mode, SYSSTATE ASCENV=AR must be specified before issuing this macro.</li> <li>• If BUFLIST was specified and your program is running in AR mode the BUFALET specification is correct.</li> </ul>
8	xxxx0836	<p><b>Equate Symbol:</b> IXLRNCODEBADREALADDR</p> <p><b>Meaning:</b> Program error. Real storage addresses were provided in a BUFLIST list, but one of the buffers is not addressable in central storage.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that BUFADDRTYPE was specified as you intended.</li> <li>• Ensure that the real buffer addresses specified by BUFLIST are valid.</li> </ul>
8	xxxx0838	<p><b>Equate Symbol:</b> IXLRNCODEBADANSAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by ANSAREA is not addressable.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the ANSAREA address.</li> <li>• ANSAREA must be addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are calling IXLCACHE while disabled, ANSAREA must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLCACHE in AR-mode and you specified the ANSAREA address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are calling IXLCACHE in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLCACHE macro.</li> </ul>

Table 30. Return and Reason Codes for the IXLCACHE REQUEST=CASTOUT\_DATALIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx083D	<p><b>Equate Symbol:</b> IXLRNCODEBADANSLEN</p> <p><b>Meaning:</b> Program error. The length of the answer area, as specified by ANSLEN, is not sufficient to contain answer area information. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Increase the size of the answer area provided for the request, and rerun your program. Depending on the macro version number, specify the length of the answer area as follows:</p> <ul style="list-style-type: none"> <li>• When the connection specified ASYNCXI=1 on the IXLCONN invocation and AXIOVERRIDE=0 was specified or defaulted to for the IXLCACHE request, the answer area length must be a minimum value of CAALEVEL2LEN</li> <li>• When the value of the macro version number (PLISTVER) is 4 or more, the minimum answer area length is CAALEVEL1LEN.</li> <li>• When the value of the macro version number (PLISTVER) is 0-3, the minimum answer area length is CAALEVEL0LEN.</li> </ul>
8	xxxx0851	<p><b>Equate Symbol:</b> IXLRNCODENOSUSPENDISABLE</p> <p><b>Meaning:</b> Program error. The request failed because MODE=SYNCSUSPEND was specified, but the caller is disabled and cannot be suspended.</p> <p><b>Action:</b> Either specify another MODE value or become enabled (release the CPU lock); then reissue the request.</p>
8	xxxx0864	<p><b>Equate Symbol:</b> IXLRNCODEBADBUFSIZE</p> <p><b>Meaning:</b> Program Error. The size of the BUFLIST areas or BUFFER area is not large enough to contain the data area for the entry in the CASTOUTLIST specified by STARTINDEX. The number of elements in the retrieved entry is returned in the answer area in the field CAAELEMNUM.</p> <p><b>Action:</b> If more space is available, specify a larger buffer size and reissue the request.</p>
8	xxxx0865	<p><b>Equate Symbol:</b> IXLRNCODEBADBUFSPEC</p> <p><b>Meaning:</b> Program error. There is an error in the buffer specification.</p> <p><b>Action:</b> Check the following:</p> <ul style="list-style-type: none"> <li>• If BUFLIST was specified, check the requirements for BUFLIST, BUFNUM, and BUFINCRNUM.</li> <li>• If BUFFER was specified, check the requirements for BUFFER and BUFSIZE.</li> <li>• Buffer pointer(s) in BUFLIST.</li> <li>• Buffer boundaries.</li> </ul>

Table 30. Return and Reason Codes for the IXLCACHE REQUEST=CASTOUT\_DATALIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0866	<p><b>Equate Symbol:</b> IXLRSNCODEBADBUFKEY</p> <p><b>Meaning:</b> Program error. The buffer storage key specified by BUFSTGKEY is incorrect or BUFSTGKEY was not specified and the PSW key does not match the key of the buffers.</p> <p>The data cannot be stored in the specified buffer area.</p> <p><b>Action:</b> Check the following:</p> <ul style="list-style-type: none"> <li>• Determine if the key of the storage being used for the buffers is different from the PSW key.</li> <li>• If BUFSTGKEY was specified, verify that the key was put in the correct bits (see the explanation of this parameter for more information).</li> <li>• If you are calling IXLCACHE in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLCACHE macro.</li> </ul>
8	xxxx0867	<p><b>Equate Symbol:</b> IXLRSNCODEBADBUFLIST</p> <p><b>Meaning:</b> Program error. The 128-byte storage area specified by BUFLIST is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The correct BUFLIST address was used.</li> <li>• The BUFLIST area was not previously freed.</li> <li>• If the caller is running in AR-mode and the BUFLIST parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If the caller is disabled, then the BUFLIST must reside in either nonpageable or disabled reference storage.</li> <li>• If your program is running in AR mode, ensure that you specified SYSSTATE ASCENV=AR before issuing the macro.</li> </ul>
8	xxxx08AD	<p><b>Equate Symbol:</b> IXLRSNCODEBADHIGHSHAREDVIRT</p> <p><b>Meaning:</b> Program error. The request specified a high shared virtual storage area (above 2GB).</p> <p><b>Action:</b> None required.</p>
C	xxxx0C06	<p><b>Equate Symbol:</b> IXLRSNCODENOCNN</p> <p><b>Meaning:</b> Environmental error. No connectivity to the cache structure exists. This may occur due to operator commands such as VARY PATH,OFFLINE or CONFIG CHP,OFFLINE or hardware errors such as facility or path failures. The CONTOKEN is invalidated. The request fails.</p> <p><b>Action:</b> Either disconnect from the structure (using IXLDISC) or initiate a rebuild of the structure, if allowed (using IXLREBLD).</p>



Table 30. Return and Reason Codes for the IXLCACHE REQUEST=CASTOUT\_DATALIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C13	<p><b>Equate Symbol:</b> IXLRSNCODEREQPURGED</p> <p><b>Meaning:</b> Environmental error. The request was purged prior to completion of the request. Possible reasons include:</p> <ul style="list-style-type: none"> <li>• The connector failed.</li> <li>• The connector disconnected.</li> <li>• The requestor failed.</li> <li>• The request was purged by IXLPURGE.</li> <li>• Requests were purged when the connector provided an IXLEERSP response for the Rebuild Quiesce, Rebuild Stop, or Rebuild Cleanup event.</li> <li>• The connector invoked IXLREBLD REQUEST=COMPLETE.</li> <li>• The secondary address space was no longer valid.</li> </ul> <p><b>Action:</b> None if this is expected. Otherwise, determine why the connector failed.</p>
C	xxxx0C14	<p><b>Equate Symbol:</b> IXLRSNCODESTATUSUNKNOWN</p> <p><b>Meaning:</b> Environmental error. The IXLCACHE request has completed, but the final disposition of the request cannot be determined. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Verify the validity of your data by comparing the expected results with what is in the coupling facility.</p>
C	xxxx0C25	<p><b>Equate Symbol:</b> IXLRSNCODESTRFAILURE</p> <p><b>Meaning:</b> Environmental error. The cache structure failed prior to completion of the request.</p> <p><b>Action:</b> Attempt to rebuild the structure using IXLREBLD, or disconnect from the structure using IXLDISC.</p>
C	xxxx0C68	<p><b>Equate Symbol:</b> IXLRSNCODEBADREQCFLEVEL</p> <p><b>Meaning:</b> Environmental error. The request type is not permitted for the level of coupling facility in which the target structure is allocated.</p> <p><b>Action:</b> Disconnect from the structure (using IXLDISC) and request that the installation provide a coupling facility of the correct CFLEVEL (CFLEVEL=12 or higher).</p>
C	xxxx0CA0	<p><b>Equate Symbol:</b> IXLRSNCODEQUIESCEDSUSPENDFAIL</p> <p><b>Meaning:</b> Environmental error. The request is failed because the structure is quiesced for a system-managed process and SUSPEND=FAIL is specified on the IXLCONN.</p> <p><b>Action:</b> None, if this is expected.</p>

Table 30. Return and Reason Codes for the IXLCACHE REQUEST=CASTOUT\_DATALIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxxFFFF	<b>Equate Symbol:</b> IXLRSNCODENOTAVAILABLE <b>Meaning:</b> Environmental error. XES functions are not available. The coupling facility hardware might not be present. <b>Action:</b> XES may not be used when XCF is running local mode. XES requires that the coupling facility be installed.
10	xxxx10xx	<b>Equate Symbol:</b> IXLRSNCODECOMPERROR <b>Meaning:</b> System error. XES processing failure. The state of the structure and the disposition of the request are unpredictable. The answer area (ANSAREA) fields do not contain valid information. <b>Action:</b> Save the reason code information, and contact the IBM support center.

---

## Chapter 33. IXLCACHE REQUEST=CROSS\_INVALID

---

### Description

A CROSS\_INVALID request allows you to invalidate the copies of the data item in the local cache buffers of other users with registered interest in the data item. A cross-invalidate operation:

- Deregisters user interest. With the exception of your connection, all users with registered interest in the data item have their interest deregistered.
- Invalidates local copies of the data item. With the exception of the copy in your local cache buffer, all users' copies of the data item are invalidated.

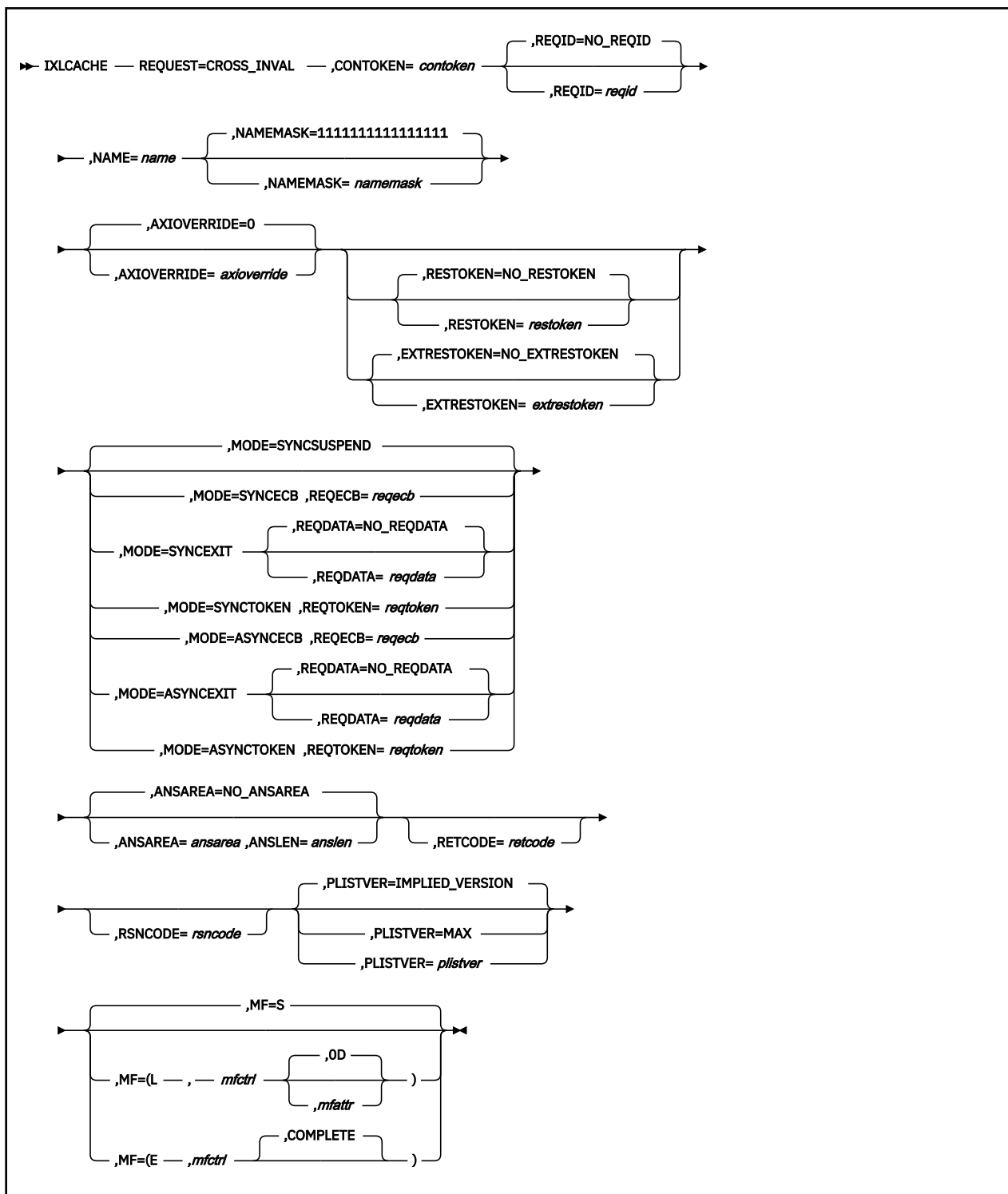
The data items against which a cross-invalidate operation is performed are identified by NAME and NAMEMASK.

If the request exceeds the model-dependent time-out criteria before processing completes, either a restart token (RESTOKEN) or an extended restart token (EXTRESTOKEN) is returned in the answer area. The token can be specified on the next CROSS\_INVALID request to resume processing with the next data item to be processed. Resumed requests are processed identically whether using the RESTOKEN or EXTRESTOKEN to specify the starting location.

---

### Syntax Diagram

The syntax diagram for IXLCACHE REQUEST=CROSS\_INVALID is as follows:



**Note:** You must specify `ANSAREA=ansarea`, `ANSLEN=anslen` if you:

- Specify `MODE=SYNCTOKEN` or `MODE=ASYNCTOKEN`, or
- Specify `AXIOVERRIDE=0` (or default to 0) and specified `ASYNCXI=1` on the `IXLCONN` invocation when connecting to the cache structure.

## Parameter Descriptions

The parameter descriptions for `REQUEST=CROSS_INVAL` are listed in alphabetical order. Default values are underlined:

**REQUEST=CROSS\_INVALID**

Use this input parameter to specify that a cross-invalidate operation be performed for one or more data items specified by NAME and NAMEMASK. All users except for your connection have their interest in the data item deregistered and the copies of the data item in their local cache buffers invalidated.

**,ANSAREA=NO\_ANSAREA****,ANSAREA=ansarea**

Use this output parameter to specify an answer area to contain:

- A restart token (CAARESTOKEN) or extended restart token (CAAEXTRESTOKEN) that is returned from a request that exceeds the model-dependent time-out criteria. See the RESTOKEN and EXTRESTOKEN parameters for a description of the restart token.
- An asynchronous cross-invalidation sequence number (CAAASYNXISEQNUM) that is returned from a request that initiates cross-invalidates of local caches asynchronously to the completion of the request.

See the Chapter 29, “IXLAXISN,” on page 449 service for a description of how to use the returned CAAASYNXISEQNUM to determine when cross-invalidates of local caches associated with the request have completed

The format of the answer area is described by the IXLYCAA mapping macro.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an area (with a length of ANSLLEN) where the information returned from the request will be put.

**,ANSLLEN=anslen**

Use this inppaameter to specify the size of the storage area specified by ANSAREA.

Use either CAALEVELOLEN, CAALEVEL1LEN or CAALEVEL2LEN of the IXLYCAA mapping macro to determine the minimum size of the answer area. The answer area length must be at least large enough to accommodate the level of the IXLYCAA mapping appropriate to the requested function.

- When the connection specified ASYNXCI=1 on the IXLCONN invocation and AXIOVERRIDE=0 was specified or defaulted to for the IXLCACHE request, the answer area length is a required parameter and must be a minimum value of CAALEVEL2LEN to contain a returned asynchronous cross-invalidation sequence number (CAAASYNXISEQNUM).
- When the value of PLISTVER is 4 or above, the minimum answer area length is CAALEVEL1LEN.
- When the value of PLISTVER is 0 - 3, the minimum answer area length is CAALEVELOLEN.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12 2 to ) of a 2-byte field that contains the length of the answer area (ANSAREA).

**,AXIOVERRIDE=0****,AXIOVERRIDE=axioveride**

Use this input parameter to specify whether the asynchronous cross-invalidation control setting of IxlConnAsyncXiYes (1) for the connection identified by CONTOKEN should be overridden for this request. Valid values are 0 (IxlCacheAXiOverrideNo) or 1 (IxlCacheAXiOverrideYes).

The AXIOVERRIDE keyword is meaningful to processing only when the connection specified ASYNXCI=1 on the IXLCONN invocation when connecting to the cache structure and the cache structure is allocated in a CFLEVEL=23 or higher coupling facility.

A value of 0 (IxlCacheAXiOverrideNo) indicates that the asynchronous cross-invalidation control for the connection as specified on the IXLCONN invocation should be used for the request. Cross-invalidates against local caches for this request will preferentially be initiated asynchronously to the completion of the request when asynchronous cross-invalidations are supported by the coupling facility where the cache structure is allocated.

A value of 1 (IxlCacheAXiOverrideYes) indicates that the ASYNXCI specification of IxlConnAsyncXiYes (1) by the connector on the IXLCONN invocation should be overridden for this request only. Cross-invalidations generated by this request will be processed synchronously to the completion of the request.

Any value other than 0 or 1 for AXIOVERRIDE will have the same behavior as specifying a value of 0 (IxIcAchEAXiOvErriDeNo).

If cross-invalidates against local caches for this request were initiated asynchronously to the completion of the request, an asynchronous cross-invalidation sequence number is returned in CAAASYNCXISEQNUM of the cache answer area (ANSAREA).

The asynchronous cross-invalidation sequence number can be used on a subsequent invocation of IXLAXISN to ensure that the asynchronous cross-invalidations associated with this request have completed.

When cross-invalidations are initiated synchronously to the completion of the request or no cross-invalidations occurred for the request, no asynchronous cross-invalidation sequence number is returned.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a one-byte input field that contains the value indicating whether the asynchronous cross-invalidation control setting of the connector should be overridden

**,CONTO=conken**

Use this input parameter to specify the connect token that was returned by the IXLCONN service in the IXLCONN answer area, which is mapped which is by IXLYCONA. The connect token uniquely identifies your connection to the cache structure, and must be specified on each IXLCACHE invocation.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field that contains the connect token.

**,EXTRESTOKEN=NO\_EXTRESTOKEN**

**,EXTRESTOKEN=extrestoken**

Use this input parameter to specify an extended restart token that can be used to resume processing of a CROSS\_INVAL request that exceeded the model-dependent time-out criteria. The extended restart token is returned in the answer area (field CAAEXTRESTOKEN), and should be specified on the next CROSS\_INVAL request to resume processing with the next data item to be processed.

If the request does not exceed the model-dependent time-out criteria, the extended restart token will not be provided.

**Note:**

1. Specifying an extended restart token of all zeros causes cache services to treat all of the entries as unprocessed.
2. Do not specify an extended restart token other than the one returned in the answer area or one set to all zeros, because results will be unpredictable.
3. Specifying an extended restart token requires that the length of the answer area be at least the length of CAALEVEL1LEN.

Requestors that specify IXLCONN ALLOWAUTO=YES must use the extended restart token. Requestors that specify or default to IXLCONN ALLOWAUTO=NO must use the standard restart token (RESTOKEN).

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-byte field that contains the extended restart token.

```
,MF=S
,MF=(L,mfctrl)
,MF=(L,mfctrl,mfattr)
,MF=(L,mfctrl,0D)
,MF=(M,mfctrl)
,MF=(M,mfctrl,COMPLETE)
,MF=(M,mfctrl,NOCHECK)
,MF=(E,mfctrl)
,MF=(E,mfctrl,COMPLETE)
,MF=(E,mfctrl,NOCHECK)
```

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

#### **,mfctrl**

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

#### **,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

#### **,COMPLETE**

#### **,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

#### **COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if SMILE=*var* were an optional parameter and the default is SMILE=NO\_SMILE then it would not be documented. However, if the default was SMILE=:~), then it would be documented because a value would be the default.

#### **NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

**,MODE=SYNCSUSPEND**  
**,MODE=SYNCECB**  
**,MODE=SYNCEXIT**  
**,MODE=SYNCTOKEN**  
**,MODE=ASYNCECB**  
**,MODE=ASYNCEXIT**  
**,MODE=ASYNCTOKEN**

Use this input parameter to specify:

- Whether the request is to be performed synchronously or asynchronously
- How you want to be notified of request completion if the request is processed asynchronously.

See *z/OS MVS Programming: Sysplex Services Guide* for more information on understanding synchronous and asynchronous cache operations.

#### **SYNCSUSPEND**

The request processes synchronously. If necessary, the request is suspended until it can complete synchronously. To use this option, your program must be enabled for I/O and external interrupts.

#### **SYNCECB**

The request processes synchronously if possible. If the request processes asynchronously, the ECB specified by REQECB is posted when the request completes.

#### **SYNCEXIT**

The request processes synchronously if possible. If the request processes asynchronously, your complete exit is given control when the request completes.

#### **SYNCTOKEN**

The request processes synchronously if possible. If the request processes asynchronously, an asynchronous request token is returned to the area specified by REQTOKEN. Use the returned request token on the IXLFComp macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=SYNCTOKEN is specified.

#### **ASYNCECB**

The request processes asynchronously. The ECB specified by REQECB is posted when the request completes.

#### **ASYNCEXIT**

The request processes asynchronously. Your complete exit is given control when the request completes.

#### **ASYNCTOKEN**

The request processes asynchronously. An asynchronous request token is returned to the area specified by REQTOKEN. Use the returned request token on the IXLFComp macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=ASYNCTOKEN is specified.

#### **,NAME=name**

Use this input parameter to specify the name of the data item against which a cross-invalidate operation will be performed. With the exception of your connection, all users with registered interest in the data item will have their interest deregistered and the copies of the data item in their local cache buffers invalidated.

You may cross-invalidate multiple data items with one invocation of this macro, if they have similar names. See the description of the NAMEMASK parameter for how to do this.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-byte field that contains the name of the data item.

#### **,NAMEMASK=1111111111111111**

#### **,NAMEMASK=namemask**

Use this input parameter to specify which characters in the name specified by NAME are to be used in selecting data items for processing. This parameter allows you to select multiple data items based on common characters in NAME.



The position of each bit in NAMEMASK corresponds to the same relative character position in NAME. A one indicates that the corresponding letter should be used in selecting entries; a zero indicates that the corresponding letter should not be used.

Specifying a name mask with all zeros causes all names to be selected for processing. Specifying a name mask with all ones causes only the name specified by NAME to be selected.

For more information on how NAMEMASK may be used, see *z/OS MVS Programming: Sysplex Services Guide*.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 2-byte field that contains the bit-mask for the name specified on the NAME keyword.

**,PLISTVER=IMPLIED\_VERSION**

**,PLISTVER=MAX**

**,PLISTVER=plistver**

Use this input parameter to specify the version of the macro. See “Understanding IXLCACHE Version Support” on page 461 for a description of the options available with PLISTVER.

**,REQDATA=NO\_REQDATA**

**,REQDATA=reqdata**

Use this input parameter with MODE=SYNCEXIT or MODE=ASYNCEXIT to pass any data you choose to the complete exit. The exit will get control only if the request is processed asynchronously.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the data to be passed to the complete exit.

**,REQECB=reqcb**

Use this output parameter with either MODE=SYNCECB or MODE=ASYNCECB to specify the address of an ECB, which is to be posted when the request completes if the request was processed asynchronously.

Before coding REQECB, you must ensure that:

- You initialize the ECB before you issue the request.
- The ECB resides in either common storage or the home address space where IXLCONN was issued.
- Any tasks that wait for the ECB to be posted reside in the home address space where IXLCONN was issued.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that contains the address of the ECB to be posted when the request completes. The ECB must be aligned on a fullword boundary.

**,REQID=NO\_REQID**

**,REQID=reqid**

Use this input parameter to specify a user-defined request identifier to be associated with the request. You can specify this request identifier on the IXPURGE macro to cancel a request that has not yet been processed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the user-defined request identifier.

**,REQTOKEN=reqtoken**

Use this output parameter with either MODE=SYNCTOKEN or MODE=ASYNCTOKEN to specify the address of a storage area to receive the request token that is returned when the request will be processed asynchronously. This token, which uniquely identifies the request, must be used as input to the IXLFCOMP macro, which you use to determine if the request has completed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field where the system will put the request token.

**,RESTOKEN=NO\_RESTOKEN**

**,RESTOKEN=restoken**

Use this input parameter to specify a restart token that can be used to resume processing of a CROSS\_INVALID request that exceeded the model-dependent time-out criteria. The restart token is

returned in the answer area (field CAARESTOKEN), and should be specified on the next CROSS\_INVALID request to resume processing with the next data item to be processed.

If the request does not exceed the model-dependent time-out criteria, the returned token will not be provided

**Note:**

1. Specifying a restart token of all zeros causes cache services to treat all of the entries as unprocessed.
2. Do not specify a restart token other than the one returned in the answer area or one set to all zeros, because results will be unpredictable.

Requestors that specify or default to IXLCONN ALLOWAUTO=NO must use the standard restart token. Requestors that specify IXLCONN ALLOWAUTO=YES must use the extended restart token (EXTRESTOKEN).

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-byte field that contains the restart token.

**,RETCODE=retcode**

Use this output parameter to specify a field to contain the return code. (The return code is also returned in register 15.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the return code when the request has completed.

**,RSNCODE=rsncode**

Use this output parameter to specify a field to contain the reason code returned, if applicable. (The reason code is also returned in register 0.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the reason code (if any) when the request has completed.

## ABEND Codes

---

Abend X'026' (See *z/OS MVS System Codes* for more information on this abend.)

## Return and Reason Codes

---

When the system returns control to your program:

- GPR 15 (and RETCODE, if specified) contains a return code.
- If the return code is not zero, GPR 0 (and RSNCODE, if specified) contains a reason code.

**Note:** The return and reason codes will also be put into the answer area (mapped by IXLYCAA), if there is one, when the request completes.

The IXLYCON macro provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

<b>0</b>	IXLRETCODEOK
<b>4</b>	IXLRETCODEWARNING
<b>8</b>	IXLRETCODEPARMERROR
<b>C</b>	IXLRETCODEENVERROR
<b>10</b>	IXLRETCODECOMPERROR

The following table contains hexadecimal return and reason codes, the equate symbols associated with each reason code, and the meaning and suggested action for each return and reason code.

Table 31. Return and Reason Codes for the IXLCACHE REQUEST=CROSS_INVALID Macro		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0	None.	<p><b>Meaning:</b> If you specified a MODE value of ASYNCECB, ASYNCEXIT, or ASYNCTOKEN, the request has been successfully initiated. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed. For any other MODE value, the request has successfully completed.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=ASYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=ASYNCEXIT, the connection's complete exit will be given control when the request has completed.</li> <li>• If you specified MODE=ASYNCTOKEN, use REQTOKEN when calling IXLFComp to determine when the request has completed.</li> </ul>
4	xxxx0402	<p><b>Equate Symbol:</b> IXLRNCodeASYNCH</p> <p><b>Meaning:</b> The request could not be processed synchronously. It will be processed asynchronously. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed.</p> <ul style="list-style-type: none"> <li>• If you specified MODE=SYNCECB, the ECB specified in REQECB will be posted when the request has finished.</li> <li>• If you specified MODE=SYNCEXIT, the connection's complete exit will be given control when the request has finished.</li> <li>• If you specified MODE=SYNCTOKEN, the REQTOKEN contains an asynchronous request token that may be used on the IXLFComp macro to determine when the request has completed.</li> </ul> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=SYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=SYNCEXIT, the connection's completion exit will be given control when the request has completed.</li> <li>• If you specified MODE=SYNCTOKEN, use REQTOKEN when calling IXLFComp to determine when the request has completed.</li> </ul>

Table 31. Return and Reason Codes for the IXLCACHE REQUEST=CROSS\_INVAL Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	xxxx0409	<p><b>Equate Symbol:</b> IXLRSNCODETIMEOUT</p> <p><b>Meaning:</b> The request has completed prematurely because the model-dependent time-out criteria of the coupling facility has been exceeded. A token for restarting the request has been returned in the answer area (field CAARESTOKEN or CAAEXTRESTOKEN).</p> <p><b>Action:</b> Reissue the request using the restart token (RESTOKEN or CAAEXTRESTOKEN). For more information about premature request completion, see <a href="#">z/OS MVS Programming: Sysplex Services Guide</a>.</p>
8	xxxx0801	<p><b>Equate Symbol:</b> IXLRSNCODEBADPARMLIST</p> <p><b>Meaning:</b> The parameter list (mfctrl) for this request is not addressable. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the parameter list address.</li> <li>• The parameter list must be addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are issuing this macro while disabled, the parameter list must reside in either page-fixed or DREF storage.</li> <li>• If you are issuing this macro in AR-mode and you specified the parameter list address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are issuing this macro in AR-mode, SYSSTATE ASCENV=AR must be issued before this macro.</li> </ul>
8	xxxx0804	<p><b>Equate Symbol:</b> IXLRSNCODEBADVERSION#</p> <p><b>Meaning:</b> The version number in the macro parameter list is not compatible with the level of XES currently being used. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that your program did not overlay the parameter list storage.</li> <li>• Verify that your program was assembled with the correct macro library for the release of MVS on which your program is running.</li> <li>• Verify that your program is running on an MVS system that supports the version of the macro you are using.</li> </ul>

Table 31. Return and Reason Codes for the IXLCACHE REQUEST=CROSS\_INVALID Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx080A	<p><b>Equate Symbol:</b> IXLRNCODEBADCONTOKEN</p> <p><b>Meaning:</b> Program error. The specified connect token (CONTOKEN) was not valid for one of the following reasons:</p> <ol style="list-style-type: none"> <li>1. The user with the connection identifier represented by the token has disconnected from the structure.</li> <li>2. The connector's task (the task that issued IXLCONN) ended.</li> <li>3. The specified token is not the token that was returned from IXLCONN.</li> <li>4. The request was issued from an address space other than the address space in which IXLCONN was issued.</li> <li>5. The connect token was invalidated during rebuild.</li> <li>6. The connect token was invalidated by XES.</li> </ol> <p><b>Note:</b> The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Verify that the CONTOKEN value specified was valid and for the correct structure. The numbers on the following actions correspond to the numbers above.</p> <ol style="list-style-type: none"> <li>1. Discontinue use. Perform recovery and cleanup for the structure and check protocol for use of structure to determine why you did not know of the disconnection.</li> <li>2. Discontinue use. Perform recovery and cleanup for the structure and check protocol for use of structure to determine why you did not know that the task ended.</li> <li>3. Issue IXLCONN to connect to the structure. The contoken is returned in the CONACONTOKEN field of the answer area specified on the IXLCONN request.</li> <li>4. Issue your request from the same address space the IXLCONN was issued.</li> <li>5. Participate in the rebuild. When it is complete, try again.</li> <li>6. Discontinue use. Perform recovery and cleanup for the structure and check protocol for use of structure.</li> </ol> <p>You may want to issue IXCQUERY to get more information about the structure.</p>
8	xxxx0824	<p><b>Equate Symbol:</b> IXLRNCODEWRONGSTRTYPE</p> <p><b>Meaning:</b> The connect token specified by CONTOKEN is not to a cache structure.</p> <p><b>Action:</b> Verify the connect token for this cache structure.</p> <p><b>Note:</b> The connect token can be found in the IXLCONN answer area. Verify the value specified on the TYPE= parameter of the IXLCONN request for this structure.</p>

Table 31. Return and Reason Codes for the IXLCACHE REQUEST=CROSS\_INVALID Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0838	<p><b>Equate Symbol:</b> IXLRNCODEBADANSAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by ANSAREA is not addressable.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the ANSAREA address.</li> <li>• ANSAREA must be addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are calling IXLCACHE while disabled, ANSAREA must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLCACHE in AR-mode and you specified the ANSAREA address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are calling IXLCACHE in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLCACHE macro.</li> </ul>
8	xxxx0839	<p><b>Equate Symbol:</b> IXLRNCODEBADREQTOKENAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by REQTOKEN is not addressable.</p> <p><b>Action:</b> Ensure that the request token area specified by REQTOKEN is valid:</p> <ul style="list-style-type: none"> <li>• Verify the REQTOKEN address.</li> <li>• If you are calling IXLCACHE while disabled, REQTOKEN must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLCACHE in AR-mode and you specified the REQTOKEN address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are calling IXLCACHE in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLCACHE macro.</li> </ul>
8	xxxx083D	<p><b>Equate Symbol:</b> IXLRNCODEBADANSLEN</p> <p><b>Meaning:</b> Program error. The length of the answer area, as specified by ANSLEN, is not sufficient to contain answer area information. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Increase the size of the answer area provided for the request, and rerun your program. Depending on the macro version number, specify the length of the answer area as follows:</p> <ul style="list-style-type: none"> <li>• When the connection specified ASYNCCI=1 on the IXLCONN invocation and AXIOVERRIDE=0 was specified or defaulted to for the IXLCACHE request, the answer area length must be a minimum value of CAALEVEL2LEN</li> <li>• When the value of the macro version number (PLISTVER) is 4 or more, the minimum answer area length is CAALEVEL1LEN.</li> <li>• When the value of the macro version number (PLISTVER) is 0-3, the minimum answer area length is CAALEVEL0LEN.</li> </ul>

Table 31. Return and Reason Codes for the IXLCACHE REQUEST=CROSS\_INVAL Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0849	<p><b>Equate Symbol:</b> IXLRNCODEBADRESTOKEN</p> <p><b>Meaning:</b> Program error. The restart token specified by RESTOKEN is not valid.</p> <p><b>Action:</b> Determine why the restart token cannot be used to resume the request. Possible causes are:</p> <ul style="list-style-type: none"> <li>• The specified token does not correspond to the restart token returned in the answer area of the previous request.</li> <li>• The user specified RESTOKEN when EXTRESTOKEN was required.</li> <li>• The user specified EXTRESTOKEN when RESTOKEN was required.</li> </ul> <p>For more information about premature request completion, see <a href="#">z/OS MVS Programming: Sysplex Services Guide</a>.</p>
8	xxxx0851	<p><b>Equate Symbol:</b> IXLRNCODENOSUSPENDISABLE</p> <p><b>Meaning:</b> Program error. The request failed because MODE=SYNCSUSPEND was specified, but the caller is disabled and cannot be suspended.</p> <p><b>Action:</b> Either specify another MODE value or become enabled (release the CPU lock); then reissue the request.</p>
8	xxxx0887	<p><b>Equate Symbol:</b> IXLRNCODEBADEXTRESTOKEN</p> <p><b>Meaning:</b> Program error. The extended restart token specified by EXTRESTOKEN is not valid. The specified token refers to an older instance of the target structure. A system-managed process occurred between the time a request returned the extended restart token and the time the connector tried to continue the request using that token.</p> <p><b>Action:</b> Discard the results of the initial request and reissue the request with an EXTRESTOKEN value of zero. For more information about restarting requests, see <a href="#">z/OS MVS Programming: Sysplex Services Guide</a>.</p>

Table 31. Return and Reason Codes for the IXLCACHE REQUEST=CROSS\_INVALID Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx08B9	<p><b>Equate Symbol:</b> IXLRSNCODENOANSAREA</p> <p><b>Meaning:</b> An answer area was not specified when one is required. The requested service determined that conditions exist that require an ANSAREA to complete the request.</p> <p><b>Action:</b> Provide an answer area (ANSAREA) and answer area length (ANSLEN) on the IXLCACHE macro invocation for the request. ANSAREA is required when the connection specified ASYNCCI=1 on the IXLCONN invocation when connecting to the cache structure, AXIOVERRIDE=0 was specified or defaulted to for the IXLCACHE request and the request is one of the following:</p> <ul style="list-style-type: none"> <li>• CROSS_INVALID</li> <li>• CROSS_INVALIDLIST</li> <li>• DELETE_NAME</li> <li>• DELETE_NAMELIST</li> <li>• READ_DATA</li> <li>• REG_NAMELIST</li> <li>• WRITE_DATA</li> <li>• WRITE_DATA_LIST</li> </ul>
C	xxxx0C06	<p><b>Equate Symbol:</b> IXLRSNCODENOCN</p> <p><b>Meaning:</b> Environmental error. No connectivity to the cache structure exists. This may occur due to operator commands such as VARY PATH,OFFLINE or CONFIG CHP,OFFLINE or hardware errors such as facility or path failures. The CONTOKEN is invalidated. The request fails.</p> <p><b>Action:</b> Either disconnect from the structure (using IXLDISC) or initiate a rebuild of the structure, if allowed (using IXLREBLD).</p>
C	xxxx0C13	<p><b>Equate Symbol:</b> IXLRSNCODEREQPURGED</p> <p><b>Meaning:</b> Environmental error. The request was purged prior to completion of the request. Possible reasons include:</p> <ul style="list-style-type: none"> <li>• The connector failed.</li> <li>• The connector disconnected.</li> <li>• The requestor failed.</li> <li>• The request was purged by IXLPURGE.</li> <li>• Requests were purged when the connector provided an IXLEERSP response for the Rebuild Quiesce, Rebuild Stop, or Rebuild Cleanup event.</li> <li>• The connector invoked IXLREBLD REQUEST=COMPLETE.</li> <li>• The secondary address space was no longer valid.</li> </ul> <p><b>Action:</b> None if this is expected. Otherwise, determine why the connector failed.</p>



Table 31. Return and Reason Codes for the IXLCACHE REQUEST=CROSS\_INVAL Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C14	<p><b>Equate Symbol:</b> IXLRSNCODESTATUSUNKNOWN</p> <p><b>Meaning:</b> Environmental error. The IXLCACHE request has completed, but the final disposition of the request cannot be determined. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Verify the validity of your data by comparing the expected results with what is in the coupling facility.</p>
C	xxxx0C25	<p><b>Equate Symbol:</b> IXLRSNCODESTRFAILURE</p> <p><b>Meaning:</b> Environmental error. The cache structure failed prior to completion of the request.</p> <p><b>Action:</b> Attempt to rebuild the structure using IXLREBLD, or disconnect from the structure using IXLDISC.</p>
C	xxxx0CA0	<p><b>Equate Symbol:</b> IXLRSNCODEQUIESCEDSUSPENDFAIL</p> <p><b>Meaning:</b> Environmental error. The request is failed because the structure is quiesced for a system-managed process and SUSPEND=FAIL is specified on the IXLCONN.</p> <p><b>Action:</b> None, if this is expected.</p>
C	xxxxFFFF	<p><b>Equate Symbol:</b> IXLRSNCODENOTAVAILABLE</p> <p><b>Meaning:</b> Environmental error. XES functions are not available. The coupling facility hardware might not be present.</p> <p><b>Action:</b> XES may not be used when XCF is running local mode. XES requires that the coupling facility be installed.</p>
10	xxxx10xx	<p><b>Equate Symbol:</b> IXLRSNCODECOMPERROR</p> <p><b>Meaning:</b> System error. XES processing failure. The state of the structure and the disposition of the request are unpredictable. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Save the reason code information, and contact the IBM support center.</p>



---

## Chapter 34. IXLCACHE REQUEST=CROSS\_INVALLIST

---

### Description

A CROSS\_INVALLIST request allows you to invalidate the copies of a set of data items in the local cache buffers of other users with registered interest in the data item. A cross-invalidate operation:

- Deregisters user interest. With the exception of your connection, all users with registered interest in the data item have their interest deregistered.
- Invalidates local copies of the data item. With the exception of the copy in your local cache buffer, all users' copies of the data item are invalidated.

A CROSS\_INVALLIST request allows you to specify that a cross-invalidate operation be performed against a set of 1 to 4096 entries identified by a list of names in the BUFFER storage area or the buffers specified by BUFLIST.

The system references the name elements by an index into the buffer and processes them sequentially beginning with the name block identify by the first index (STARTINDEX) and ending with the data item identified by the last index (ENDINDEX). The name elements are numbered starting with 1.

If any name element fails to identify an existing structure entry, processing either continues or terminates, depending on the value specified for the ERRORACTION keyword. If ERRORACTION=TERMINATE, processing stops and the index value of the name element that caused the problem is returned in the CAACILINDEX field in the IXLYCAA answer area. If ERRORACTION=CONTINUE, processing continues with the next name element.

A CROSS\_INVALLIST can complete prematurely because the request exceeds the time-out criteria for the coupling facility. (Time-out criteria is model-dependent.) When a request completes prematurely, the system returns an index value (CAACILINDEX) in the answer area which you can use to restart the CROSS\_INVALLIST request. The index value returned when a CROSS\_INVALLIST request completes prematurely is the index of the first unprocessed name element.

A CROSS\_INVALLIST request can be issued only for cache structures allocated in a coupling facility of CFLEVEL=12 or higher. CROSS\_INVALLIST requests issued for a cache structure allocated in a coupling facility of CFLEVEL=0 through 11 will fail.

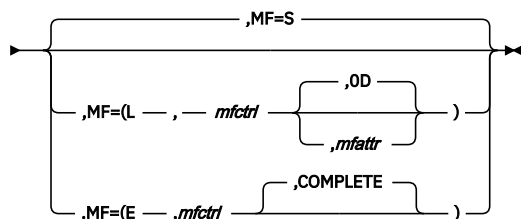
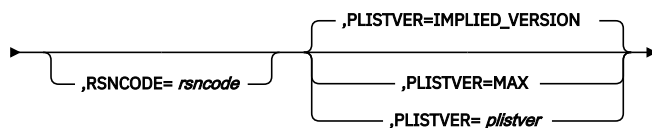
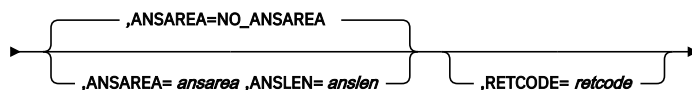
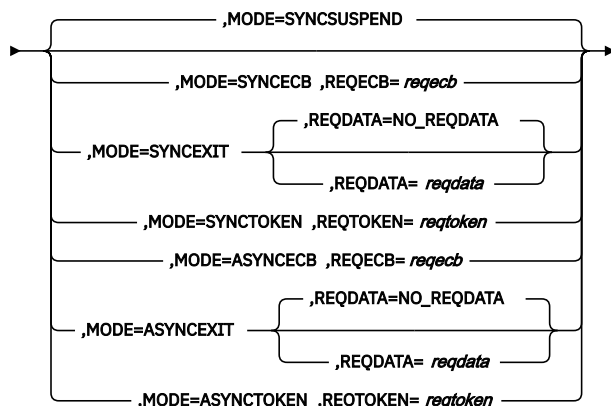
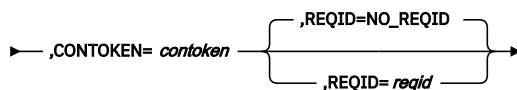
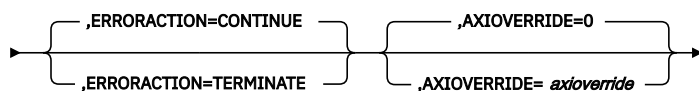
---

### Syntax Diagram

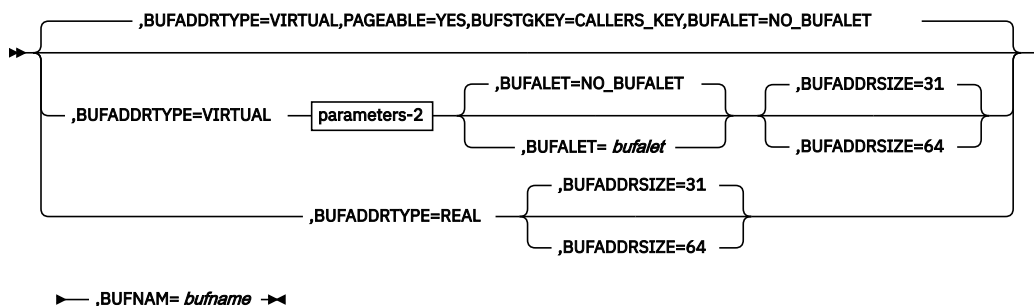
The syntax diagram for IXLCACHE REQUEST=CROSS\_INVALLIST is as follows:

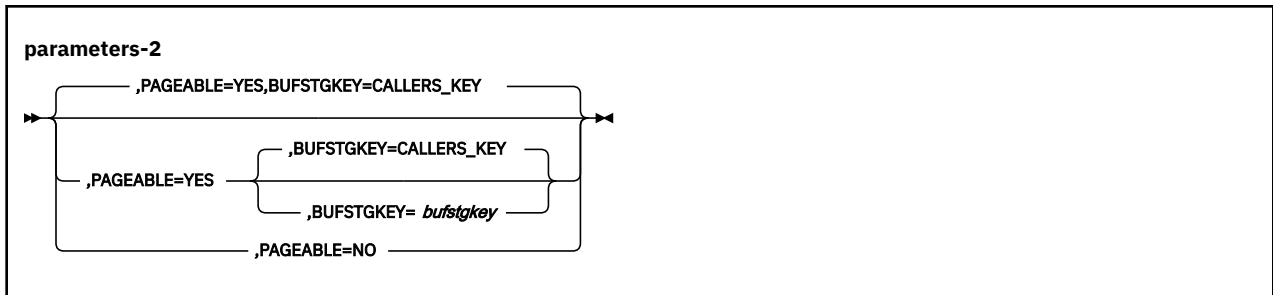
## main diagram

➔ IXLCACHE — REQUEST=CROSS\_INVALLIST — ,STARTINDEX= *startindex* — ,ENDINDEX= *endindex* ➔



## parameters-1





**Note:** You must specify `ANSAREA=ansarea`, `ANSLEN=anslen` if you:

- Specify `MODE=SYNCTOKEN` or `MODE=ASYNCTOKEN`, or
- Specify `AXIOVERRIDE=0` (or default to 0) and specified `ASYNCXI=1` on the `IXLCONN` invocation when connecting to the cache structure.

## Parameter Descriptions

The parameter descriptions for `REQUEST=CROSS_INVALLIST` are listed in alphabetical order. Default values are underlined:

### **REQUEST=CROSS\_INVALLIST**

Use this input parameter to specify that a cross-invalidate operation be performed against a given set of 1 to 4096 entries specified by a list of names in the `BUFFER` storage area or the buffers specified by `BUFLIST`. `STARTINDEX` and `ENDINDEX` determine the set of names provided by the user in the `BUFFER` or `BUFLIST` that will be processed by the request. Valid `STARTINDEX` and `ENDINDEX` values are 1-origin. The result of the operation is that with the exception of the connection specified by `CONTOKEN`, all connections that registered interest in the specified entries have interest deregistered and a cross-invalidate performed against their local caches.

#### **,ANSAREA=NO ANSAREA**

#### **,ANSAREA=ansarea**

Use this output parameter to specify an answer area to contain information returned from the request if the request does not complete successfully or if the request initiated asynchronous cross-invalidations. See the return and reason codes descriptions for this request to determine which fields in the answer area are valid for non-zero return codes.

An asynchronous cross-invalidation sequence number (`CAAASYNCXISEQNUM`) is returned from a request that initiates cross-invalidates of local caches asynchronously to the completion of the request. See the `IXLAXISN` service for a description of how to use the returned `CAAASYNCXISEQNUM` to determine when cross-invalidates of local caches associated with the request have completed.

The format of the answer area is described by the `IXLYCAA` mapping macro.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an area (with a length of `ANSLEN`) where the information returned from the request will be put.

#### **,ANSLEN=anslen**

Use this inppaameter to specify the size of the storage area specified by `ANSAREA`.

Use either `CAALEVELOLEN`, `CAALEVEL1LEN` or `CAALEVEL2LEN` of the `IXLYCAA` mapping macro to determine the minimum size of the answer area. The answer area length must be at least large enough to accommodate the level of the `IXLYCAA` mapping appropriate to the requested function.

- When the connection specified `ASYNCXI=1` on the `IXLCONN` invocation and `AXIOVERRIDE=0` was specified or defaulted to for the `IXLCACHE` request, the answer area length is a required parameter and must be a minimum value of `CAALEVEL2LEN` to contain a returned asynchronous cross-invalidation sequence number (`CAAASYNCXISEQNUM`).
- When the value of `PLISTVER` is 4 or above, the minimum answer area length is `CAALEVEL1LEN`.
- When the value of `PLISTVER` is 0 - 3, the minimum answer area length is `CAALEVELOLEN`.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12 2 to ) of a 2-byte field that contains the length of the answer area (ANSAREA).

**,AXIOVERRIDE=0**

**,AXIOVERRIDE=axioverride**

Use this input parameter to specify whether the asynchronous cross-invalidation control setting of IxlConnAsyncXiYes (1) for the connection identified by CONTOKEN should be overridden for this request. Valid values are 0 (IxlCacheAXiOverrideNo) or 1 (IxlCacheAXiOverrideYes).

The AXIOVERRIDE keyword is meaningful to processing only when the connection specified ASYNCXI=1 on the IXLCONN invocation when connecting to the cache structure and the cache structure is allocated in a CFLEVEL=23 or higher coupling facility.

A value of 0 (IxlCacheAXiOverrideNo) indicates that the asynchronous cross-invalidation control for the connection as specified on the IXLCONN invocation should be used for the request. Cross-invalidates against local caches for this request will preferentially be initiated asynchronously to the completion of the request when asynchronous cross-invalidations are supported by the coupling facility where the cache structure is allocated.

A value of 1 (IxlCacheAXiOverrideYes) indicates that the ASYNCXI specification of IxlConnAsyncXiYes (1) by the connector on the IXLCONN invocation should be overridden for this request only. Cross-invalidations generated by this request will be processed synchronously to the completion of the request.

Any value other than 0 or 1 for AXIOVERRIDE will have the same behavior as specifying a value of 0 (IxlCacheAXiOverrideNo).

If cross-invalidates against local caches for this request were initiated asynchronously to the completion of the request, an asynchronous cross-invalidation sequence number is returned in CAAASYNCXISEQNUM of the cache answer area (ANSAREA).

The asynchronous cross-invalidation sequence number can be used on a subsequent invocation of IXLAXISN to ensure that the asynchronous cross-invalidations associated with this request have completed.

When cross-invalidations are initiated synchronously to the completion of the request or no cross-invalidations occurred for the request, no asynchronous cross-invalidation sequence number is returned.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12 2 to ) of a one-byte input field that contains the value indicating whether the asynchronous cross-invalidation control setting of the connector should be overridden

**,BUFADDRSIZE=31**

**,BUFADDRSIZE=64**

Use this input parameter to specify whether a 31-bit or a 64-bit address is specified by a BUFLIST entry.

**31**

The entry in BUFLIST is 31 bits in size.

**64**

The entry in BUFLIST is 64 bits in size.

**,BUFADDRTYPE=VIRTUAL**

**,BUFADDRTYPE=REAL**

Use this input parameter to specify whether the buffer addresses specified in the BUFLIST list are virtual storage or real storage addresses.

**VIRTUAL**

The buffer addresses are virtual storage addresses. The virtual storage can be pageable or nonpageable. See the PAGEABLE parameter for information about managing storage binds when specifying virtual storage addresses.

**REAL**

The buffer addresses are real storage addresses.

It is the caller's responsibility to manage the binds between the data buffer virtual storage and the real storage addresses provided. The caller must ensure that the data buffer virtual storage remains bound to the real storage addresses provided until the request completes.

**,BUFALET=NO\_BUFALET**

**,BUFALET=*bufalet***

Use this input parameter to specify an access list entry token (ALET) to be used in referencing all of the buffers specified by BUFLIST.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 4-byte field that contains the ALET.

**,BUFFER=*buffer***

Use this input parameter to specify a buffer area to contain the set of 16-byte structure entry names on which a cross-invalidate operation is to be performed.

Only 31-bit addressable virtual storage areas (below 2GB) are supported by the BUFFER specification. High virtual storage areas (above 2GB) can only be specified via the BUFLIST specification.

Requirements for the buffer length are as follows:

- If you specify a buffer size of less than or equal to 4096 bytes, you must ensure that the buffer:
  - Is 256, 512, 1024, 2048, or 4096 bytes.
  - Starts on a 256-byte boundary.
  - Does not cross a 4096-byte boundary.
  - Does not start below storage address 512.
- If you specify a buffer size of greater than 4096 bytes, you must ensure that the buffer:
  - Is a multiple of 4096 bytes.
  - Is less than or equal to 65536 bytes.
  - Starts on a 4096-byte boundary.
  - Does not start below storage address 512.

See the BUFSIZE parameter description for defining the size of the buffer. See [z/OS MVS Programming: Sysplex Services Guide](#) for more information on buffers.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an area (with a length of BUFSIZE) to contain the structure entry names.

**,BUFINCRNUM=*bufincrnum***

Use this input parameter to specify the number of 256-byte segments comprising each buffer in the BUFLIST list.

Valid BUFINCRNUM values are 1, 2, 4, 8, or 16, which correspond to BUFLIST buffer sizes of 256, 512, 1024, 2048, and 4096 bytes respectively.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 1-byte field that contains 1, 2, 4, 8, or 16.

**,BUFLIST=*buflist***

Use this input parameter to specify a list of buffers to hold the structure entry names identifying the entries on which a cross-invalidate operation is to be performed. BUFLIST specifies a 128-byte storage area that consists of a list of 1 to 16 buffer address elements.

Either 31-bit addressable (below 2GB) or 64-bit addressable (above 2GB) real or virtual storage areas are supported for the BUFLIST specification, depending on the specification for the BUFADDRTYPE and BUFADDRSIZE keywords. However, pageable high shared virtual storage areas (above 2GB) may not be used.

The format of the list is a set of 8-byte elements. The BUFADDRSIZE keyword denotes whether four or eight bytes of the element are used.

- If BUFADDRSIZE=31 is specified, then the first four bytes of each element are reserved space and the last four bytes contain the address of the buffer.
- If BUFADDRSIZE=64 is specified, then the full eight bytes specify the address of the buffer.

**The BUFLIST buffers must:**

- Reside in the same address space or data space as defined by BUFALET.
- Be the same size: either 256, 512, 1024, 2048, or 4096 bytes as defined by BUFINCRNUM.
- Start on a 256-byte boundary and not cross a 4096-byte boundary.
- Not start below storage address 512.

**Note:** The buffers do not have to be contiguous in storage. XES treats BUFLIST buffers as a single buffer even if the buffers are not contiguous.

See the BUFNUM and BUFINCRNUM keyword descriptions for specifying the number and size of buffers.

See *z/OS MVS Programming: Sysplex Services Guide* for more information on buffers.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 128-byte field that contains a list of buffer addresses.

**,BUFNUM=*bufnum***

Use this input parameter to specify the number of buffers in the BUFLIST list. Valid BUFNUM values are from 1 to 16.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 1-byte field that contains the number of buffers (1 to 16) in the list (BUFLIST)

**,BUFSIZE=*bufsize***

Use this input parameter to specify the size of the BUFFER area. See the BUFFER parameter description for valid buffer sizes.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a fullword field that contains the size of the buffer (BUFFER) in bytes.

**,BUFSTGKEY=CALLERS\_KEY**

**,BUFSTGKEY=*bufstgkey***

Use this input parameter to specify a storage key that you define and use when referencing the buffer specified by BUFFER.

If you do not specify BUFSTGKEY, or if you specify BUFSTGKEY=CALLERS\_KEY, all references to the buffer are in the caller's PSW key.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 1-byte field that contains the storage key in the format kkkkxxxx, where kkkk is the key and xxxx is ignored.

**,CONTO=*conken***

Use this input parameter to specify the connect token that was returned by the IXLCONN service in the IXLCONN answer area, which is mapped which is by IXLYCONA. The connect token uniquely identifies your connection to the cache structure, and must be specified on each IXLCACHE invocation.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field that contains the connect token.

**,ENDINDEX=*endindex***

Use this input parameter to specify the index into the name elements in the buffer storage area or the buffers specified by BUFLIST of the last name element to be processed. The index value must be greater than or equal to the value specified for STARTINDEX.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a halfword field containing the ending index value.



**ERRORACTION=CONTINUE****ERRORACTION=TERMINATE**

Use this input parameter to specify whether processing is to continue with the next name element if an entry is not found.

**CONTINUE**

If an error occurs, processing is to continue with the next name element.

**TERMINATE**

If an error occurs, processing is to halt and the index of the name element that caused the error is returned in the answer area.

**,MF=S****,MF=(L,*mfctrl*)****,MF=(L,*mfctrl*,*mfattr*)****,MF=(L,*mfctrl*,0D)****,MF=(M,*mfctrl*)****,MF=(M,*mfctrl*,COMPLETE)****,MF=(M,*mfctrl*,NOCHECK)****,MF=(E,*mfctrl*)****,MF=(E,*mfctrl*,COMPLETE)****,MF=(E,*mfctrl*,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

**,*mfctrl***

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

**,*mfattr***

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

**,COMPLETE****,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

**COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if SMILE=*var* were an optional parameter and the default is SMILE=NO\_SMILE then it would not be documented. However, if the default was SMILE=-), then it would be documented because a value would be the default.

**NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

**,MODE=SYNCSUSPEND**

**,MODE=SYNCECB**

**,MODE=SYNCEXIT**

**,MODE=SYNCTOKEN**

**,MODE=ASYNCECB**

**,MODE=ASYNCEXIT**

**,MODE=ASYNCTOKEN**

Use this input parameter to specify:

- Whether the request is to be performed synchronously or asynchronously
- How you want to be notified of request completion if the request is processed asynchronously.

See *z/OS MVS Programming: Sysplex Services Guide* for more information on understanding synchronous and asynchronous cache operations.

**SYNCSUSPEND**

The request processes synchronously. If necessary, the request is suspended until it can complete synchronously. To use this option, your program must be enabled for I/O and external interrupts.

**SYNCECB**

The request processes synchronously if possible. If the request processes asynchronously, the ECB specified by REQECB is posted when the request completes.

**SYNCEXIT**

The request processes synchronously if possible. If the request processes asynchronously, your complete exit is given control when the request completes.

**SYNCTOKEN**

The request processes synchronously if possible. If the request processes asynchronously, an asynchronous request token is returned to the area specified by REQTOKEN. Use the returned request token on the IXLFCOMP macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=SYNCTOKEN is specified.

**ASYNCECB**

The request processes asynchronously. The ECB specified by REQECB is posted when the request completes.

**ASYNCEXIT**

The request processes asynchronously. Your complete exit is given control when the request completes.

**ASYNCTOKEN**

The request processes asynchronously. An asynchronous request token is returned to the area specified by REQTOKEN. Use the returned request token on the IXLFCOMP macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=ASYNCTOKEN is specified.

**,PLISTVER=IMPLIED\_VERSION**

**,PLISTVER=MAX**

**,PLISTVER=plistver**

Use this input parameter to specify the version of the macro. See “[Understanding IXLCACHE Version Support](#)” on page 461 for a description of the options available with PLISTVER.

**,REQDATA=NO\_REQDATA**

**,REQDATA=reqdata**

Use this input parameter with MODE=SYNCEXIT or MODE=ASYNCEXIT to pass any data you choose to the complete exit. The exit will get control only if the request is processed asynchronously.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the data to be passed to the complete exit.

**,REQECB=reqecb**

Use this output parameter with either MODE=SYNCECB or MODE=ASYNCECB to specify the address of an ECB, which is to be posted when the request completes if the request was processed asynchronously.

Before coding REQECB, you must ensure that:

- You initialize the ECB before you issue the request.
- The ECB resides in either common storage or the home address space where IXLCONN was issued.
- Any tasks that wait for the ECB to be posted reside in the home address space where IXLCONN was issued.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that contains the address of the ECB to be posted when the request completes. The ECB must be aligned on a fullword boundary.

**,REQID=NO\_REQID****,REQID=reqid**

Use this input parameter to specify a user-defined request identifier to be associated with the request. You can specify this request identifier on the IXPURGE macro to cancel a request that has not yet been processed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the user-defined request identifier.

**,REQTOKEN=reqtoken**

Use this output parameter with either MODE=SYNCTOKEN or MODE=ASYNCTOKEN to specify the address of a storage area to receive the request token that is returned when the request will be processed asynchronously. This token, which uniquely identifies the request, must be used as input to the IXLFCOMP macro, which you use to determine if the request has completed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field where the system will put the request token.

**,RETCODE=retcode**

Use this output parameter to specify a field to contain the return code. (The return code is also returned in register 15.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the return code when the request has completed.

**,RSNCODE=rsncode**

Use this output parameter to specify a field to contain the reason code returned, if applicable. (The reason code is also returned in register 0.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the reason code (if any) when the request has completed.

**,STARTINDEX=startindex**

Use this input parameter to specify the starting index for name element processing. Valid STARTINDEX values are from 1 to the value of ENDINDEX. The first name element has index number 1.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a halfword field containing the starting index value.

## ABEND Codes

---

Abend X'026' (See *z/OS MVS System Codes* for more information on this abend.)

## Return and Reason Codes

---

When the system returns control to your program:

- GPR 15 (and RETCODE, if specified) contains a return code.
- If the return code is not zero, GPR 0 (and RSNCODE, if specified) contains a reason code.

**Note:** The return and reason codes will also be put into the answer area (mapped by IXLYCAA), if there is one, when the request completes.

The IXLYCON macro provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

<b>0</b>	IXLRETCODEOK
<b>4</b>	IXLRETCODEWARNING
<b>8</b>	IXLRETCODEPARMERROR
<b>C</b>	IXLRETCODEENVERROR
<b>10</b>	IXLRETCODECOMPERROR

The following table contains hexadecimal return and reason codes, the equate symbols associated with each reason code, and the meaning and suggested action for each return and reason code.

Table 32. Return and Reason Codes for the IXLCACHE REQUEST=CROSS_INVALLIST Macro		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0	None.	<p><b>Meaning:</b> If you specified a MODE value of ASYNCECB, ASYNCEXIT, or ASYNCTOKEN, the request has been successfully initiated. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed. For any other MODE value, the request has successfully completed.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=ASYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=ASYNCEXIT, the connection's complete exit will be given control when the request has completed.</li> <li>• If you specified MODE=ASYNCTOKEN, use REQTOKEN when calling IXLFComp to determine when the request has completed.</li> </ul>

Table 32. Return and Reason Codes for the IXLCACHE REQUEST=CROSS\_INVALLIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	xxxx0402	<p><b>Equate Symbol:</b> IXLSRNCODEASYNCH</p> <p><b>Meaning:</b> The request could not be processed synchronously. It will be processed asynchronously. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed.</p> <ul style="list-style-type: none"> <li>• If you specified MODE=SYNCECB, the ECB specified in REQECB will be posted when the request has finished.</li> <li>• If you specified MODE=SYNCEXIT, the connection's complete exit will be given control when the request has finished.</li> <li>• If you specified MODE=SYNCTOKEN, the REQTOKEN contains an asynchronous request token that may be used on the IXLFCOMP macro to determine when the request has completed.</li> </ul> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=SYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=SYNCEXIT, the connection's completion exit will be given control when the request has completed.</li> <li>• If you specified MODE=SYNCTOKEN, use REQTOKEN when calling IXLFCOMP to determine when the request has completed.</li> </ul>
4	xxxx0409	<p><b>Equate Symbol:</b> IXLSRNCODETIMEOUT</p> <p><b>Meaning:</b> The request has completed prematurely because the model-dependent time-out criteria of the coupling facility has been exceeded. The index of the first unprocessed entry has been returned in the answer area (field CAACILINDEX). All prior entries have been processed.</p> <p><b>Action:</b> To restart the request, update STARTINDEX with the value of CAACILINDEX, the index of the next entry in the list of names to be processed in the BUFFER storage area or the buffers specified by BUFLIST. For more information about premature request completion, see <a href="#">z/OS MVS Programming: Sysplex Services Guide</a>.</p>

Table 32. Return and Reason Codes for the IXLCACHE REQUEST=CROSS\_INVALLIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0801	<p><b>Equate Symbol:</b> IXLRSNCODEBADPARMLIST</p> <p><b>Meaning:</b> The parameter list (mfctrl) for this request is not addressable. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the parameter list address.</li> <li>• The parameter list must be addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are issuing this macro while disabled, the parameter list must reside in either page-fixed or DREF storage.</li> <li>• If you are issuing this macro in AR-mode and you specified the parameter list address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are issuing this macro in AR-mode, SYSSTATE ASCENV=AR must be issued before this macro.</li> </ul>
8	xxxx0804	<p><b>Equate Symbol:</b> IXLRSNCODEBADVERSION#</p> <p><b>Meaning:</b> The version number in the macro parameter list is not compatible with the level of XES currently being used. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that your program did not overlay the parameter list storage.</li> <li>• Verify that your program was assembled with the correct macro library for the release of MVS on which your program is running.</li> <li>• Verify that your program is running on an MVS system that supports the version of the macro you are using.</li> </ul>

Table 32. Return and Reason Codes for the IXLCACHE REQUEST=CROSS\_INVALLIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx080A	<p><b>Equate Symbol:</b> IXLRNCODEBADCONTOKEN</p> <p><b>Meaning:</b> Program error. The specified connect token (CONTOKEN) was not valid for one of the following reasons:</p> <ol style="list-style-type: none"> <li>1. The user with the connection identifier represented by the token has disconnected from the structure.</li> <li>2. The connector's task (the task that issued IXLCONN) ended.</li> <li>3. The specified token is not the token that was returned from IXLCONN.</li> <li>4. The request was issued from an address space other than the address space in which IXLCONN was issued.</li> <li>5. The connect token was invalidated during rebuild.</li> <li>6. The connect token was invalidated by XES.</li> </ol> <p><b>Note:</b> The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Verify that the CONTOKEN value specified was valid and for the correct structure. The numbers on the following actions correspond to the numbers above.</p> <ol style="list-style-type: none"> <li>1. Discontinue use. Perform recovery and cleanup for the structure and check protocol for use of structure to determine why you did not know of the disconnection.</li> <li>2. Discontinue use. Perform recovery and cleanup for the structure and check protocol for use of structure to determine why you did not know that the task ended.</li> <li>3. Issue IXLCONN to connect to the structure. The contoken is returned in the CONACONTOKEN field of the answer area specified on the IXLCONN request.</li> <li>4. Issue your request from the same address space the IXLCONN was issued.</li> <li>5. Participate in the rebuild. When it is complete, try again.</li> <li>6. Discontinue use. Perform recovery and cleanup for the structure and check protocol for use of structure.</li> </ol> <p>You may want to issue IXCQUERY to get more information about the structure.</p>
8	xxxx0824	<p><b>Equate Symbol:</b> IXLRNCODEWRONGSTRTYPE</p> <p><b>Meaning:</b> The connect token specified by CONTOKEN is not to a cache structure.</p> <p><b>Action:</b> Verify the connect token for this cache structure.</p> <p><b>Note:</b> The connect token can be found in the IXLCONN answer area. Verify the value specified on the TYPE= parameter of the IXLCONN request for this structure.</p>

Table 32. Return and Reason Codes for the IXLCACHE REQUEST=CROSS\_INVALLIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0825	<p><b>Equate Symbol:</b> IXLRNCOENOENTRY</p> <p><b>Meaning:</b> Program error. The request failed because a name element specified an entry name that does not exist in the cache structure and ERRORACTION=TERMINATE was specified. The index of the failing name is returned in the answer area (field CAACILINDEX). All prior name elements were processed.</p> <p><b>Action:</b> Reissue the request specifying for STARTINDEX the index of the next valid name element to be processed.</p>
8	xxxx082B	<p><b>Equate Symbol:</b> IXLRNCOEBADIDINDEX</p> <p><b>Meaning:</b> Program error. The name index specified by either STARTINDEX or ENDINDEX is not valid. No elements have been processed.</p> <p><b>Action:</b> Ensure that STARTINDEX and ENDINDEX specify valid indexes into the name elements stored in BUFLIST or BUFFER. The value of ENDINDEX must be greater than or equal to STARTINDEX and ENDINDEX must be less than or equal to 4096.</p>
8	xxxx0833	<p><b>Equate Symbol:</b> IXLRNCOEBADPGBLATTR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list is specified as being pageable (PAGEABLE=YES) but is not.</p> <p><b>Action:</b> Change the buffer area(s) to pageable storage, or specify PAGEABLE=NO. See the PAGEABLE parameter description for specification instructions.</p>
8	xxxx0834	<p><b>Equate Symbol:</b> IXLRNCOEBADNONPGBLATTR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list is specified as being nonpageable (PAGEABLE=NO) but is either pageable or not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The type of storage in the buffer area corresponds to the value you specified on the PAGEABLE parameter. See the PAGEABLE parameter description for specification instructions.</li> <li>• The correct buffer address was used.</li> <li>• The buffer area(s) were not previously freed.</li> <li>• If BUFLIST was specified and your program is running in AR mode, ensure that: <ul style="list-style-type: none"> <li>– The BUFALET specification is correct.</li> <li>– You specified SYSSTATE ASCENV=AR before issuing the macro.</li> </ul> </li> <li>• If the caller is running in AR-mode and the BUFFER or BUFLIST parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> </ul>



Table 32. Return and Reason Codes for the IXLCACHE REQUEST=CROSS\_INVALLIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0835	<p><b>Equate Symbol:</b> IXLRNCODEBADDATAADDR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list, is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The correct buffer address was used for BUFFER or for a buffer within the BUFLIST.</li> <li>• The buffer area(s) were not previously freed.</li> <li>• The buffer area(s) were allocated in a storage key that matches the key specified using the BUFSTGKEY parameter, or, if BUFSTGKEY is omitted, the caller's PSW key.</li> <li>• If the caller is running in AR-mode and the BUFFER or BUFLIST parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If the caller is running in AR-mode, SYSSTATE ASCENV=AR must be specified before issuing this macro.</li> <li>• If BUFLIST was specified and your program is running in AR mode the BUFALET specification is correct.</li> </ul>
8	xxxx0836	<p><b>Equate Symbol:</b> IXLRNCODEBADREALADDR</p> <p><b>Meaning:</b> Program error. Real storage addresses were provided in a BUFLIST list, but one of the buffers is not addressable in central storage.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that BUFADDRTYPE was specified as you intended.</li> <li>• Ensure that the real buffer addresses specified by BUFLIST are valid.</li> </ul>
8	xxxx0838	<p><b>Equate Symbol:</b> IXLRNCODEBADANSAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by ANSAREA is not addressable.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the ANSAREA address.</li> <li>• ANSAREA must be addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are calling IXLCACHE while disabled, ANSAREA must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLCACHE in AR-mode and you specified the ANSAREA address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are calling IXLCACHE in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLCACHE macro.</li> </ul>

Table 32. Return and Reason Codes for the IXLCACHE REQUEST=CROSS\_INVALLIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0839	<p><b>Equate Symbol:</b> IXLRNCODEBADREQTOKENAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by REQTOKEN is not addressable.</p> <p><b>Action:</b> Ensure that the request token area specified by REQTOKEN is valid:</p> <ul style="list-style-type: none"> <li>• Verify the REQTOKEN address.</li> <li>• If you are calling IXLCACHE while disabled, REQTOKEN must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLCACHE in AR-mode and you specified the REQTOKEN address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are calling IXLCACHE in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLCACHE macro.</li> </ul>
8	xxxx083D	<p><b>Equate Symbol:</b> IXLRNCODEBADANSLEN</p> <p><b>Meaning:</b> Program error. The length of the answer area, as specified by ANSLEN, is not sufficient to contain answer area information. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Increase the size of the answer area provided for the request, and rerun your program. Depending on the macro version number, specify the length of the answer area as follows:</p> <ul style="list-style-type: none"> <li>• When the connection specified ASYNCXI=1 on the IXLCONN invocation and AXIOVERRIDE=0 was specified or defaulted to for the IXLCACHE request, the answer area length must be a minimum value of CAALEVEL2LEN</li> <li>• When the value of the macro version number (PLISTVER) is 4 or more, the minimum answer area length is CAALEVEL1LEN.</li> <li>• When the value of the macro version number (PLISTVER) is 0-3, the minimum answer area length is CAALEVEL0LEN.</li> </ul>
8	xxxx0851	<p><b>Equate Symbol:</b> IXLRNCODENOSUSPENDISABLE</p> <p><b>Meaning:</b> Program error. The request failed because MODE=SYNCSUSPEND was specified, but the caller is disabled and cannot be suspended.</p> <p><b>Action:</b> Either specify another MODE value or become enabled (release the CPU lock); then reissue the request.</p>
8	xxxx0865	<p><b>Equate Symbol:</b> IXLRNCODEBADBUFSPEC</p> <p><b>Meaning:</b> Program error. There is an error in the buffer specification.</p> <p><b>Action:</b> Check the following:</p> <ul style="list-style-type: none"> <li>• Requirements for BUFFER and BUFSIZE.</li> <li>• Buffer boundaries.</li> </ul>

Table 32. Return and Reason Codes for the IXLCACHE REQUEST=CROSS\_INVALLIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0866	<p><b>Equate Symbol:</b> IXLRNCODEBADBUFKEY</p> <p><b>Meaning:</b> Program error. The buffer storage key specified by BUFSTGKEY is incorrect or BUFSTGKEY was not specified and the PSW key does not match the key of the buffers.</p> <p>The data cannot be stored in the specified buffer area.</p> <p><b>Action:</b> Check the following:</p> <ul style="list-style-type: none"> <li>• Determine if the key of the storage being used for the buffers is different from the PSW key.</li> <li>• If BUFSTGKEY was specified, verify that the key was put in the correct bits (see the explanation of this parameter for more information).</li> <li>• If you are calling IXLCACHE in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLCACHE macro.</li> </ul>
8	xxxx0867	<p><b>Equate Symbol:</b> IXLRNCODEBADBUFLIST</p> <p><b>Meaning:</b> Program error. The 128-byte storage area specified by BUFLIST is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The correct BUFLIST address was used.</li> <li>• The BUFLIST area was not previously freed.</li> <li>• If the caller is running in AR-mode and the BUFLIST parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If the caller is disabled, then the BUFLIST must reside in either nonpageable or disabled reference storage.</li> <li>• If your program is running in AR mode, ensure that you specified SYSSTATE ASCENV=AR before issuing the macro.</li> </ul>
8	xxxx08AD	<p><b>Equate Symbol:</b> IXLRNCODEBADHIGHSHAREDVIRT</p> <p><b>Meaning:</b> Program error. The request specified a high shared virtual storage area (above 2GB).</p> <p><b>Action:</b> None required.</p>

Table 32. Return and Reason Codes for the IXLCACHE REQUEST=CROSS\_INVALLIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx08B9	<p><b>Equate Symbol:</b> IXLRSNCODENOANSAREA</p> <p><b>Meaning:</b> An answer area was not specified when one is required. The requested service determined that conditions exist that require an ANSAREA to complete the request.</p> <p><b>Action:</b> Provide an answer area (ANSAREA) and answer area length (ANSLEN) on the IXLCACHE macro invocation for the request. ANSAREA is required when the connection specified ASYNCCI=1 on the IXLCONN invocation when connecting to the cache structure, AXIOVERRIDE=0 was specified or defaulted to for the IXLCACHE request and the request is one of the following:</p> <ul style="list-style-type: none"> <li>• CROSS_INVALID</li> <li>• CROSS_INVALLIST</li> <li>• DELETE_NAME</li> <li>• DELETE_NAMELIST</li> <li>• READ_DATA</li> <li>• REG_NAMELIST</li> <li>• WRITE_DATA</li> <li>• WRITE_DATA_LIST</li> </ul>
C	xxxx0C06	<p><b>Equate Symbol:</b> IXLRSNCODENOCN</p> <p><b>Meaning:</b> Environmental error. No connectivity to the cache structure exists. This may occur due to operator commands such as VARY PATH,OFFLINE or CONFIG CHP,OFFLINE or hardware errors such as facility or path failures. The CONTOKEN is invalidated. The request fails.</p> <p><b>Action:</b> Either disconnect from the structure (using IXLDISC) or initiate a rebuild of the structure, if allowed (using IXLREBLD).</p>
C	xxxx0C13	<p><b>Equate Symbol:</b> IXLRSNCODEREQPURGED</p> <p><b>Meaning:</b> Environmental error. The request was purged prior to completion of the request. Possible reasons include:</p> <ul style="list-style-type: none"> <li>• The connector failed.</li> <li>• The connector disconnected.</li> <li>• The requestor failed.</li> <li>• The request was purged by IXLPURGE.</li> <li>• Requests were purged when the connector provided an IXLEERSP response for the Rebuild Quiesce, Rebuild Stop, or Rebuild Cleanup event.</li> <li>• The connector invoked IXLREBLD REQUEST=COMPLETE.</li> <li>• The secondary address space was no longer valid.</li> </ul> <p><b>Action:</b> None if this is expected. Otherwise, determine why the connector failed.</p>

Table 32. Return and Reason Codes for the IXLCACHE REQUEST=CROSS\_INVALLIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C14	<p><b>Equate Symbol:</b> IXLRSNCODESTATUSUNKNOWN</p> <p><b>Meaning:</b> Environmental error. The IXLCACHE request has completed, but the final disposition of the request cannot be determined. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Verify the validity of your data by comparing the expected results with what is in the coupling facility.</p>
C	xxxx0C25	<p><b>Equate Symbol:</b> IXLRSNCODESTRFAILURE</p> <p><b>Meaning:</b> Environmental error. The cache structure failed prior to completion of the request.</p> <p><b>Action:</b> Attempt to rebuild the structure using IXLREBLD, or disconnect from the structure using IXLDISC.</p>
C	xxxx0C68	<p><b>Equate Symbol:</b> IXLRSNCODEBADREQCFLEVEL</p> <p><b>Meaning:</b> Environmental error. The request type is not permitted for the level of coupling facility in which the target structure is allocated.</p> <p><b>Action:</b> Disconnect from the structure (using IXLDISC) and request that the installation provide a coupling facility of the correct CFLEVEL (CFLEVEL=12 or higher).</p>
C	xxxx0CA0	<p><b>Equate Symbol:</b> IXLRSNCODEQUIESCEDSUSPENDFAIL</p> <p><b>Meaning:</b> Environmental error. The request is failed because the structure is quiesced for a system-managed process and SUSPEND=FAIL is specified on the IXLCONN.</p> <p><b>Action:</b> None, if this is expected.</p>
C	xxxxFFFF	<p><b>Equate Symbol:</b> IXLRSNCODENOTAVAILABLE</p> <p><b>Meaning:</b> Environmental error. XES functions are not available. The coupling facility hardware might not be present.</p> <p><b>Action:</b> XES may not be used when XCF is running local mode. XES requires that the coupling facility be installed.</p>
10	xxxx10xx	<p><b>Equate Symbol:</b> IXLRSNCODECOMPERROR</p> <p><b>Meaning:</b> System error. XES processing failure. The state of the structure and the disposition of the request are unpredictable. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Save the reason code information, and contact the IBM support center.</p>



---

## Chapter 35. IXLCACHE REQUEST=DELETE\_NAME

### Description

---

A DELETE\_NAME request allows you to delete one or more data items identified by NAME and NAMEMASK from the cache structure. For cache structures allocated in a coupling facility of CFLEVEL=4 or lower, a DELETE\_NAME request frees directory and data-entry resources for reuse. All users with registered interest in the data items have their interest deregistered and the copies of the data items in their local cache buffers invalidated. The data items are also removed from storage and cast-out classes.

For cache structures allocated in a coupling facility of CFLEVEL=5 or higher, you can specify for each data entry what resources are to be deleted. The DELETETYPE keyword provides options that allow you to delete the directory entry and associated data, only the changed data but not the directory entry, only the unchanged data but not the directory data, or any data (changed or unchanged) but not the directory entry.

Optionally, with a structure allocated in a coupling facility of CFLEVEL=5 or higher, you can specify that a version comparison is to be made for each entry processed. If the comparison does not meet the condition specified for the comparison, the entry is not processed and processing continues with the next entry to be considered.

In a coupling facility of CFLEVEL=7 or higher, you can use NAMEMASK on an IXLCACHE DELETE\_NAME request in conjunction with NAMECLASSMASK (specified on the IXLCONN request to connect to the cache structure) to improve the efficiency of selecting data items for processing.

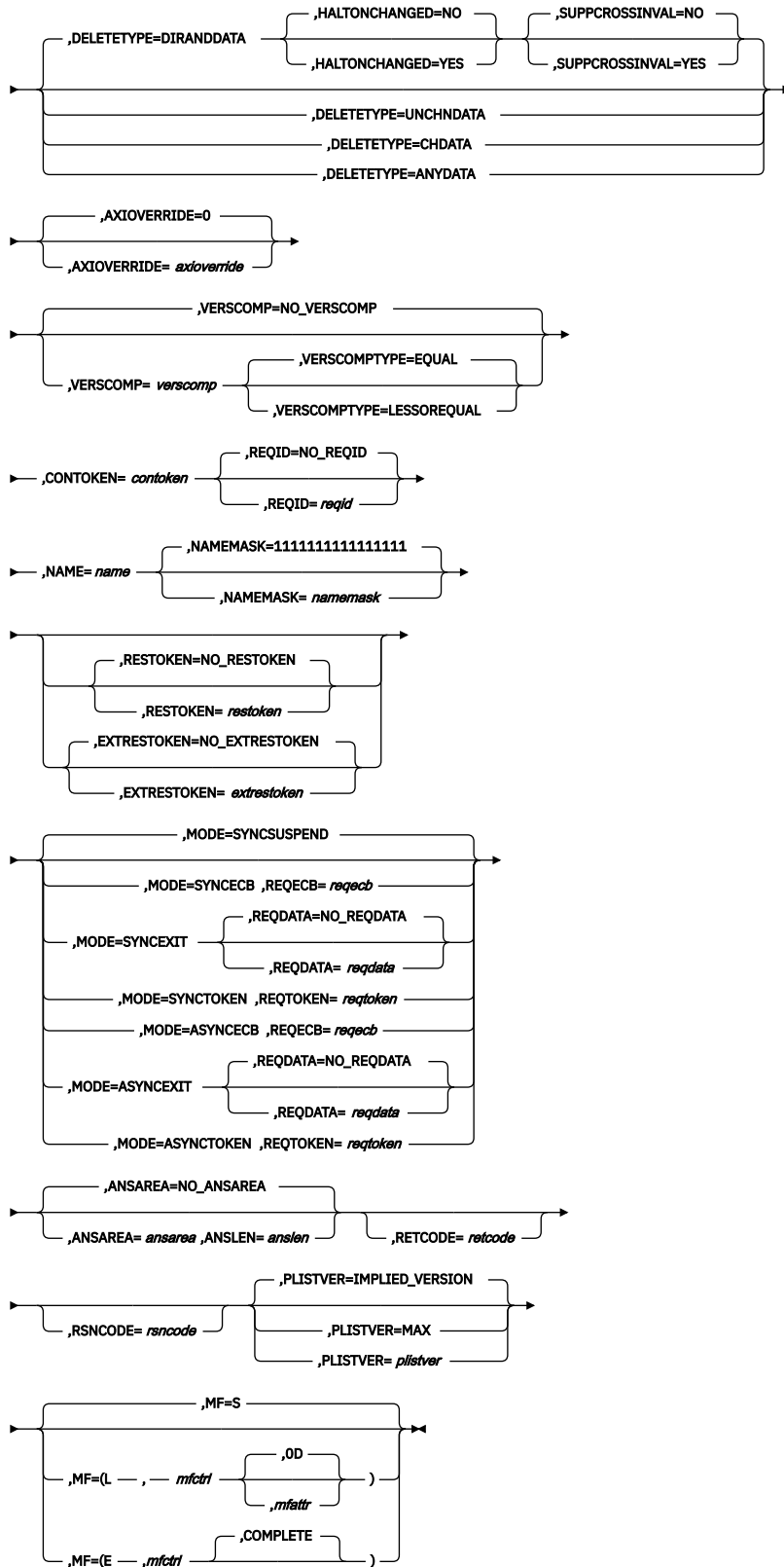
If the request exceeds the model-dependent time-out criteria before it completes or a halt condition is found as a result of specifying HALTONCHANGED=YES, a restart token (RESTOKEN) or an extended restart token (EXTRESTOKEN) is returned in the answer area (mapped by IXLYCAA). The token can be specified on the next DELETE\_NAME request to resume processing with the next data item (or in the case of when the request was halted because a structure entry met the HALTONCHANGED criteria, with the data item that caused the request to halt) to be deleted. Resumed requests are processed identically whether using the RESTOKEN or EXTRESTOKEN to specify the starting location.

### Syntax Diagram

---

The syntax diagram for the IXLCACHE REQUEST=DELETE\_NAME is as follows:

► IXLCACHE — REQUEST=DELETE\_NAME →



**Note:** You must specify ANSAREA=ansarea, ANSLEN=anslen if you:

- Specify MODE=SYNCTOKEN or MODE=ASYNCTOKEN, or



- Specify AXIOVERRIDE=0 (or default to 0) and specified ASYNCXI=1 on the IXLCONN invocation when connecting to the cache structure.

## Parameter Descriptions

---

The parameter descriptions for REQUEST=DELETE\_NAME are listed in alphabetical order. Default values are underlined:

### **REQUEST=DELETE\_NAME**

Use this input parameter to specify that one or more data items specified by NAME and NAMEMASK are to be deleted from the cache structure.

### **,ANSAREA=NO ANSAREA**

### **,ANSAREA=*ansarea***

Use this output parameter to specify an answer area to contain:

- A restart token (CAARESTOKEN) or extended restart token (CAAEXTRESTOKEN) that is returned from a request that exceeds the model-dependent time-out criteria. See the RESTOKEN and EXTRESTOKEN parameters for a description of the restart token.
- An asynchronous cross-invalidation sequence number (CAAASYNXISEQNUM) that is returned from a request that initiates cross-invalidates of local caches asynchronously to the completion of the request.

See the [Chapter 29, “IXLAXISN,” on page 449](#) service for a description of how to use the returned CAAASYNXISEQNUM to determine when cross-invalidates of local caches associated with the request have completed.

The format of the answer area is described by the IXLYCAA mapping macro.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an area (with a length of ANSLEN) where the information returned from the request will be put.

### **,ANSLEN=*anslen***

Use this inppaameter to specify the size of the storage area specified by ANSAREA.

Use either CAALEVELOLEN, CAALEVEL1LEN or CAALEVEL2LEN of the IXLYCAA mapping macro to determine the minimum size of the answer area. The answer area length must be at least large enough to accommodate the level of the IXLYCAA mapping appropriate to the requested function.

- When the connection specified ASYNCXI=1 on the IXLCONN invocation and AXIOVERRIDE=0 was specified or defaulted to for the IXLCACHE request, the answer area length is a required parameter and must be a minimum value of CAALEVEL2LEN to contain a returned asynchronous cross-invalidation sequence number (CAAASYNXISEQNUM).
- When the value of PLISTVER is 4 or above, the minimum answer area length is CAALEVEL1LEN.
- When the value of PLISTVER is 0 - 3, the minimum answer area length is CAALEVELOLEN.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12 2 to ) of a 2-byte field that contains the length of the answer area (ANSAREA).

### **,CONTO=*conken***

Use this input parameter to specify the connect token that was returned by the IXLCONN service in the IXLCONN answer area, which is mapped which is by IXLYCONA. The connect token uniquely identifies your connection to the cache structure, and must be specified on each IXLCACHE invocation.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field that contains the connect token.

**,AXIOVERRIDE=0****,AXIOVERRIDE=axioveride**

Use this input parameter to specify whether the asynchronous cross-invalidation control setting of IxlConnAsyncXiYes (1) for the connection identified by CONTOKEN should be overridden for this request. Valid values are 0 (IxlCacheAXiOverrideNo) or 1 (IxlCacheAXiOverrideYes).

The AXIOVERRIDE keyword is meaningful to processing only when the connection specified ASYNCXI=1 on the IXLCONN invocation when connecting to the cache structure and the cache structure is allocated in a CFLEVEL=23 or higher coupling facility.

A value of 0 (IxlCacheAXiOverrideNo) indicates that the asynchronous cross-invalidation control for the connection as specified on the IXLCONN invocation should be used for the request. Cross-invalidation against local caches for this request will preferentially be initiated asynchronously to the completion of the request when asynchronous cross-invalidations are supported by the coupling facility where the cache structure is allocated.

A value of 1 (IxlCacheAXiOverrideYes) indicates that the ASYNCXI specification of IxlConnAsyncXiYes (1) by the connector on the IXLCONN invocation should be overridden for this request only. Cross-invalidations generated by this request will be processed synchronously to the completion of the request.

Any value other than 0 or 1 for AXIOVERRIDE will have the same behavior as specifying a value of 0 (IxlCacheAXiOverrideNo).

If cross-invalidation against local caches for this request were initiated asynchronously to the completion of the request, an asynchronous cross-invalidation sequence number is returned in CAAASYNXISEQNUM of the cache answer area (ANSAREA).

The asynchronous cross-invalidation sequence number can be used on a subsequent invocation of IXLAXISN to ensure that the asynchronous cross-invalidations associated with this request have completed.

When cross-invalidations are initiated synchronously to the completion of the request or no cross-invalidations occurred for the request, no asynchronous cross-invalidation sequence number is returned.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12 2 to ) of a one-byte input field that contains the value indicating whether the asynchronous cross-invalidation control setting of the connector should be overridden

**,DELETETYPE=DIRANDDATA****,DELETETYPE=UNCHNDATA****,DELETETYPE=CHDATA****,DELETETYPE=ANYDATA**

Use this input parameter to specify the type of delete processing that is to be performed.

DELETETYPE is valid only for structures allocated in a coupling facility of CFLEVEL=5 or higher.

**DIRANDDATA**

For each applicable structure entry, invalidate the name and remove the name from storage and cast-out classes, and release all directory and data entry resources for the structure for reuse by the structure. The system deregisters interest in all connections with registered interest in the entry and, optionally, performs a cross-invalidate against their local caches.

Cross-invalidation against local caches may be suppressed to improve overall coupling facility performance and to quicken the completion of DELETE\_NAME requests.

**UNCHNDATA**

For each applicable structure entry, if the data is unchanged, release the data entry resources for the entry for reuse by the structure. The system does not delete the directory entry and does not perform a cross-invalidate.

**CHDATA**

For each applicable structure, if the data is changed, release the data entry resources for the entry for reuse by the structure, set the change bit, the castout lock, and the castout lock state to zero,

and remove the directory entry from the castout class. The system does not delete the directory entry and does not perform a cross-invalidate.

#### **ANYDATA**

For each applicable structure, whether the data is changed or unchanged, release the data entry resources for the entry for reuse by the structure. If the data is changed, set the change bit, the castout lock, and the castout lock state to zero, and remove the directory entry from the cast-out class. The system does not delete the directory entry and does not perform a cross-invalidate.

#### **,EXTRESTOKEN=NO\_EXTRESTOKEN**

#### **,EXTRESTOKEN=*extrestoken***

Use this input parameter to specify an extended restart token that can be used to resume processing of a DELETE\_NAME request that completed prematurely. The extended restart token is returned in the answer area (field CAAEXTRESTOKEN), and should be specified on the next DELETE\_NAME request to resume processing with the next data item to be processed.

If the request does not exceed the model-dependent time-out criteria or request processing was not halted (HALTONCHANGED=NO) or defaulted to NO), the extended restart token will not be provided.

#### **Note:**

1. Specifying an extended restart token of all zeros causes cache services to treat all of the entries as unprocessed.
2. Do not specify an extended restart token other than the one returned in the answer area or one set to all zeros, because results will be unpredictable.
3. Specifying an extended restart token requires that the length of the answer area be at least the length of CAALEVEL1LEN.

Requestors that specify IXLCONN ALLOWAUTO=YES must use the extended restart token. Requestors that specify or default to IXLCONN ALLOWAUTO=NO must use the standard restart token (RESTOKEN).

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-byte field that contains the extended restart token.

#### **HALTONCHANGED=NO**

#### **HALTONCHANGED=YES**

Use this input parameter to specify whether the DELETE\_NAME request should be halted when a structure entry is encountered that either contains changed data or for which the cast-out lock is currently held.

HALTONCHANGED is only meaningful when delete operations are issued to cache structures allocated in a coupling facility that supports request halting based on data changed status and cast-out lock state.

#### **HALTONCHANGED=NO**

The DELETE\_NAME request should continue processing a structure entry which contains changed data or for which the cast-out lock is held.

#### **HALTONCHANGED=YES**

The DELETE\_NAME request should be halted if a structure entry contains changed data or for which the cast-out lock is held. The structure entry is not deleted. Specifying HALTONCHANGED=YES can prevent the inadvertent loss of changed data that has not yet been cast out to permanent storage. When the request is halted due to the presence of a structure entry that meets the halt criteria, information identifying the structure entry is returned in the answer area specified by ANSAREA. A restart token is returned in the answer area for resuming the halted request.

HALTONCHANGED=YES is intended to prevent the inadvertent loss of changed data that has not yet been cast out to permanent storage. When the DELETE\_NAME request is halted, the application is expected to take some action to change the state of the indicated structure entry data to "unchanged" (i.e. the cast-out lock is not held and the status of the data is unchanged) before resuming the request with the restart token. One such action could be to read the entry

data for castout, write the data to permanent storage, then reset the cast-out lock to the not-held state.

Restarting the DELETE\_NAME request before taking such an action to change the state of the structure entry data to "unchanged" may result in the request halting on the same structure entry again thus preventing the request from proceeding further to process all intended structure entries.

When the request is halted due to the presence of a structure entry that meets the halt criteria, the request will complete with a return code IXLRETCODEWARNING, reason code IXLRSNCODEHALTCHANGEDDATA and the answer area specified by ANSAREA will contain specific information to identify the entry that was found to be in the "changed" state.

```
,MF=S
,MF=(L,mfctrl)
,MF=(L,mfctrl,mfattr)
,MF=(L,mfctrl,0D)
,MF=(M,mfctrl)
,MF=(M,mfctrl,COMPLETE)
,MF=(M,mfctrl,NOCHECK)
,MF=(E,mfctrl)
,MF=(E,mfctrl,COMPLETE)
,MF=(E,mfctrl,NOCHECK)
```

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

**,mfctrl**

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

**,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

**,COMPLETE**

**,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

**COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if SMILE=*var* were an optional parameter and the default is SMILE=NO\_SMILE then it would not be documented. However, if the default was SMILE=-), then it would be documented because a value would be the default.

**NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

**,MODE=SYNCSUSPEND**

**,MODE=SYNCECB**

**,MODE=SYNCEXIT**

**,MODE=SYNCTOKEN**

**,MODE=ASYNCECB**

**,MODE=ASYNCEXIT**

**,MODE=ASYNCTOKEN**

Use this input parameter to specify:

- Whether the request is to be performed synchronously or asynchronously
- How you want to be notified of request completion if the request is processed asynchronously.

See *z/OS MVS Programming: Sysplex Services Guide* for more information on understanding synchronous and asynchronous cache operations.

**SYNCSUSPEND**

The request processes synchronously. If necessary, the request is suspended until it can complete synchronously. To use this option, your program must be enabled for I/O and external interrupts.

**SYNCECB**

The request processes synchronously if possible. If the request processes asynchronously, the ECB specified by REQECB is posted when the request completes.

**SYNCEXIT**

The request processes synchronously if possible. If the request processes asynchronously, your complete exit is given control when the request completes.

**SYNCTOKEN**

The request processes synchronously if possible. If the request processes asynchronously, an asynchronous request token is returned to the area specified by REQTOKEN. Use the returned request token on the IXLFCOMP macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=SYNCTOKEN is specified.

**ASYNCECB**

The request processes asynchronously. The ECB specified by REQECB is posted when the request completes.

**ASYNCEXIT**

The request processes asynchronously. Your complete exit is given control when the request completes.

**ASYNCTOKEN**

The request processes asynchronously. An asynchronous request token is returned to the area specified by REQTOKEN. Use the returned request token on the IXLFCOMP macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=ASYNCTOKEN is specified.

**,NAME=name**

Use this input parameter to specify the name of the data item to be deleted from the cache structure.

You may delete multiple data items with one invocation of this macro, if they have similar names. See the description of the NAMEMASK parameter for how to do this.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-byte field that contains the name of the data item.

**,NAMEMASK=1111111111111111**

**,NAMEMASK=namemask**

Use this input parameter to specify which characters in the name specified by NAME are to be used in selecting data items for processing. This parameter allows you to select multiple data items based on common characters in NAME.

The position of each bit in NAMEMASK corresponds to the same relative character position in NAME. A one indicates that the corresponding letter should be used in selecting entries; a zero indicates that the corresponding letter should not be used.

Specifying a name mask with all zeros causes all names to be selected for processing. Specifying a name mask with all ones causes only the name specified by NAME to be selected.

For more information on how NAMEMASK may be used, see *z/OS MVS Programming: Sysplex Services Guide*.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12 2 to ) of a 2-byte field that contains the bit-mask for the name specified on the NAME keyword.

**,PLISTVER=IMPLIED\_VERSION**

**,PLISTVER=MAX**

**,PLISTVER=plistver**

Use this input parameter to specify the version of the macro. See “[Understanding IXLCACHE Version Support](#)” on page 461 for a description of the options available with PLISTVER.

**,REQDATA=NO\_REQDATA**

**,REQDATA=reqdata**

Use this input parameter with MODE=SYNCEXIT or MODE=ASYNCEXIT to pass any data you choose to the complete exit. The exit will get control only if the request is processed asynchronously.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the data to be passed to the complete exit.

**,REQECB=reqecb**

Use this output parameter with either MODE=SYNCECB or MODE=ASYNCECB to specify the address of an ECB, which is to be posted when the request completes if the request was processed asynchronously.

Before coding REQECB, you must ensure that:

- You initialize the ECB before you issue the request.
- The ECB resides in either common storage or the home address space where IXLCONN was issued.
- Any tasks that wait for the ECB to be posted reside in the home address space where IXLCONN was issued.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that contains the address of the ECB to be posted when the request completes. The ECB must be aligned on a fullword boundary.

**,REQID=NO\_REQID**

**,REQID=reqid**

Use this input parameter to specify a user-defined request identifier to be associated with the request. You can specify this request identifier on the IXPURGE macro to cancel a request that has not yet been processed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the user-defined request identifier.

**,REQTOKEN=reqtoken**

Use this output parameter with either MODE=SYNCTOKEN or MODE=ASYNCTOKEN to specify the address of a storage area to receive the request token that is returned when the request will be processed asynchronously. This token, which uniquely identifies the request, must be used as input to the IXLFComp macro, which you use to determine if the request has completed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field where the system will put the request token.

**,RESTOKEN=NO RESTOKEN**

**,RESTOKEN=restoken**

Use this input parameter to specify an extended restart token that can be used to resume processing of a DELETE\_NAME request that completed prematurely. The extended restart token is returned in the answer area (field CAARESTOKEN), and should be specified on the next DELETE\_NAME request to resume processing with the next data item to be deleted.

If the request does not exceed the model-dependent time-out criteria or request processing was not halted (HALTONCHANGED=NO or defaulted to NO), the extended restart token will not be provided.

**Note:**

1. Specifying a restart token of all zeros causes cache services to treat all of the entries as unprocessed.
2. Do not specify a restart token other than the one returned in the answer area or one set to all zeros, because results will be unpredictable.

Requestors that specify or default to IXLCONN ALLOWAUTO=NO must use the standard restart token. Requestors that specify IXLCONN ALLOWAUTO=YES must use the extended restart token (EXTRESTOKEN).

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-byte field that contains the restart token.

**,RETCODE=retcode**

Use this output parameter to specify a field to contain the return code. (The return code is also returned in register 15.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the return code when the request has completed.

**,RSNCODE=rsncode**

Use this output parameter to specify a field to contain the reason code returned, if applicable. (The reason code is also returned in register 0.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the reason code (if any) when the request has completed.

**SUPPCROSSINVAL=NO**

**SUPPCROSSINVAL=YES**

Use this input parameter to specify whether cross-invalidate processing associated with the DELETETYPE=DIRANDDATA request should be suppressed.

SUPPCROSSINVAL is only meaningful when delete operations are issued to cache structures allocated in a coupling facility that supports suppressing cross invalidation processing.

**SUPPCROSSINVAL=NO**

Cross-invalidation should be performed (not suppressed) against the local caches of all connections with registered interest in a successfully deleted structure entry.

**SUPPCROSSINVAL=YES**

Cross-invalidation should not be performed (suppressed) against the local caches of all connections with registered interest in a successfully deleted structure entry. Cross-invalidation signals associated with the cross-invalidation of local caches are not sent which improves overall performance by reducing coupling facility link traffic, saving CPU cycles and quickening the completion of the DELETE\_NAME request.

When requesting that the system suppress cross-invalidation of the local cache of a deleted data item, the IXLVECTR macro cannot be used to check the validity of that local cache data item using the vector index that had been associated with the data item. The vector index will be in an indeterminate state until it is subsequently reused for a new data item. The application must take

its own measures to ensure the data integrity of the local cache buffers and the subsequent reuse of the associated vector indexes.

**,VERSCOMP=NO\_VERSCOMP**

**,VERSCOMP=verscomp**

Use this input parameter to specify a value to be compared to the version number of the entry designated by NAME.

VERSCOMP is meaningful only for structures allocated in a coupling facility of CFLEVEL=5 or higher.

If the condition specified by VERSCOMPTYPE is not met, the entry is not processed and processing continues with the next entry to be considered.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-byte field that contains a value to be compared to each entry's version number.

**,VERSCOMPTYPE=EQUAL**

**,VERSCOMPTYPE=LESSOREQUAL**

Use this input parameter to specify how the structure entry version number comparison is to be performed.

VERSCOMPTYPE is meaningful only for structures allocated in a coupling facility of CFLEVEL=5 or higher and may be specified only when VERSCOMP also is specified.

**VERSCOMPTYPE=EQUAL**

The version number for the structure entry must be equal to the value specified for VERSCOMP.

**VERSCOMPTYPE=LESSOREQUAL**

The version number for the structure entry must be less than or equal to the value specified for VERSCOMP.

## ABEND Codes

---

Abend X'026' (See *z/OS MVS System Codes* for more information on this abend.)

## Return and Reason Codes

---

When the system returns control to your program:

- GPR 15 (and RETCODE, if specified) contains a return code.
- If the return code is not zero, GPR 0 (and RSNCODE, if specified) contains a reason code.

**Note:** The return and reason codes will also be put into the answer area (mapped by IXLYCAA), if there is one, when the request completes.

The IXLYCON macro provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

**0**

IXLRETCODEOK

**4**

IXLRETCODEWARNING

**8**

IXLRETCODEPARMERROR

**C**

IXLRETCODEENVERROR

**10**

IXLRETCODECOMPERROR

The following table contains hexadecimal return and reason codes, the equate symbols associated with each reason code, and the meaning and suggested action for each return and reason code.



Table 33. Return and Reason Codes for the IXLCACHE REQUEST=DELETE\_NAME Macro

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0	None.	<p><b>Meaning:</b> If you specified a MODE value of ASYNCECB, ASYNCEXIT, or ASYNCTOKEN, the request has been successfully initiated. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed. For any other MODE value, the request has successfully completed.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=ASYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=ASYNCEXIT, the connection's complete exit will be given control when the request has completed.</li> <li>• If you specified MODE=ASYNCTOKEN, use REQTOKEN when calling IXLFComp to determine when the request has completed.</li> </ul>
4	xxxx0402	<p><b>Equate Symbol:</b> IXLRNCODEASYNCH</p> <p><b>Meaning:</b> The request could not be processed synchronously. It will be processed asynchronously. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed.</p> <ul style="list-style-type: none"> <li>• If you specified MODE=SYNCECB, the ECB specified in REQECB will be posted when the request has finished.</li> <li>• If you specified MODE=SYNCEXIT, the connection's complete exit will be given control when the request has finished.</li> <li>• If you specified MODE=SYNCTOKEN, the REQTOKEN contains an asynchronous request token that may be used on the IXLFComp macro to determine when the request has completed.</li> </ul> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=SYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=SYNCEXIT, the connection's completion exit will be given control when the request has completed.</li> <li>• If you specified MODE=SYNCTOKEN, use REQTOKEN when calling IXLFComp to determine when the request has completed.</li> </ul>

Table 33. Return and Reason Codes for the IXLCACHE REQUEST=DELETE\_NAME Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	xxxx0409	<p><b>Equate Symbol:</b> IXLRNCODETIMEOUT</p> <p><b>Meaning:</b> The request has completed prematurely because the model-dependent time-out criteria of the coupling facility has been exceeded. A token for restarting the request has been returned in the answer area (field CAARESTOKEN or CAAEXTRESTOKEN).</p> <p><b>Action:</b> Reissue the request using the restart token (RESTOKEN or CAAEXTRESTOKEN). For more information about premature request completion, see <a href="#">z/OS MVS Programming: Sysplex Services Guide</a>.</p>
4	xxxx042A	<p><b>Equate Symbol:</b> IXLRNCODEHALTCHANGEDDATA</p> <p><b>Meaning:</b> A DELETE_NAME request with HALTONCHANGED=YES specified was halted due to a structure entry being found to have data in the changed state or having a cast-out lock held. The entry name that met the halt criteria as well as cast-out information for the entry has been returned in the answer area.</p> <p><b>Action:</b> Since delete processing was requested to be halted when changed data or data locked for cast-out was encountered, the application should treat the deletion attempt as an error condition and prior to restarting the request, write the data to permanent storage and change the state of the indicated structure entry data to "unchanged" (that is, the cast-out lock is not held and the status of the data is unchanged) before resuming the request with the restart token returned in the answer area (field CAARESTOKEN or CAAEXTRESTOKEN). For more information about premature request completion, see <a href="#">z/OS MVS Programming: Sysplex Services Guide</a>.</p>
8	xxxx0801	<p><b>Equate Symbol:</b> IXLRNCODEBADPARMLIST</p> <p><b>Meaning:</b> The parameter list (mfctrl) for this request is not addressable. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the parameter list address.</li> <li>• The parameter list must be addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are issuing this macro while disabled, the parameter list must reside in either page-fixed or DREF storage.</li> <li>• If you are issuing this macro in AR-mode and you specified the parameter list address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are issuing this macro in AR-mode, SYSSTATE ASCENV=AR must be issued before this macro.</li> </ul>

Table 33. Return and Reason Codes for the IXLCACHE REQUEST=DELETE\_NAME Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0804	<p><b>Equate Symbol:</b> IXLRSNCODEBADVERSION#</p> <p><b>Meaning:</b> The version number in the macro parameter list is not compatible with the level of XES currently being used. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that your program did not overlay the parameter list storage.</li> <li>• Verify that your program was assembled with the correct macro library for the release of MVS on which your program is running.</li> <li>• Verify that your program is running on an MVS system that supports the version of the macro you are using.</li> </ul>

Table 33. Return and Reason Codes for the IXLCACHE REQUEST=DELETE\_NAME Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx080A	<p><b>Equate Symbol:</b> IXLSRNCODEBADCONTOKEN</p> <p><b>Meaning:</b> Program error. The specified connect token (CONTOKEN) was not valid for one of the following reasons:</p> <ol style="list-style-type: none"> <li>1. The user with the connection identifier represented by the token has disconnected from the structure.</li> <li>2. The connector's task (the task that issued IXLCONN) ended.</li> <li>3. The specified token is not the token that was returned from IXLCONN.</li> <li>4. The request was issued from an address space other than the address space in which IXLCONN was issued.</li> <li>5. The connect token was invalidated during rebuild.</li> <li>6. The connect token was invalidated by XES.</li> </ol> <p><b>Note:</b> The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Verify that the CONTOKEN value specified was valid and for the correct structure. The numbers on the following actions correspond to the numbers above.</p> <ol style="list-style-type: none"> <li>1. Discontinue use. Perform recovery and cleanup for the structure and check protocol for use of structure to determine why you did not know of the disconnection.</li> <li>2. Discontinue use. Perform recovery and cleanup for the structure and check protocol for use of structure to determine why you did not know that the task ended.</li> <li>3. Issue IXLCONN to connect to the structure. The contoken is returned in the CONACONTOKEN field of the answer area specified on the IXLCONN request.</li> <li>4. Issue your request from the same address space the IXLCONN was issued.</li> <li>5. Participate in the rebuild. When it is complete, try again.</li> <li>6. Discontinue use. Perform recovery and cleanup for the structure and check protocol for use of structure.</li> </ol> <p>You may want to issue IXCQUERY to get more information about the structure.</p>
8	xxxx0824	<p><b>Equate Symbol:</b> IXLSRNCODEWRONGSTRTYPE</p> <p><b>Meaning:</b> The connect token specified by CONTOKEN is not to a cache structure.</p> <p><b>Action:</b> Verify the connect token for this cache structure.</p> <p><b>Note:</b> The connect token can be found in the IXLCONN answer area. Verify the value specified on the TYPE= parameter of the IXLCONN request for this structure.</p>

Table 33. Return and Reason Codes for the IXLCACHE REQUEST=DELETE\_NAME Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0838	<p><b>Equate Symbol:</b> IXLRNCODEBADANSAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by ANSAREA is not addressable.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the ANSAREA address.</li> <li>• ANSAREA must be addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are calling IXLCACHE while disabled, ANSAREA must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLCACHE in AR-mode and you specified the ANSAREA address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are calling IXLCACHE in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLCACHE macro.</li> </ul>
8	xxxx0839	<p><b>Equate Symbol:</b> IXLRNCODEBADREQTOKENAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by REQTOKEN is not addressable.</p> <p><b>Action:</b> Ensure that the request token area specified by REQTOKEN is valid:</p> <ul style="list-style-type: none"> <li>• Verify the REQTOKEN address.</li> <li>• If you are calling IXLCACHE while disabled, REQTOKEN must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLCACHE in AR-mode and you specified the REQTOKEN address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are calling IXLCACHE in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLCACHE macro.</li> </ul>
8	xxxx083D	<p><b>Equate Symbol:</b> IXLRNCODEBADANSLEN</p> <p><b>Meaning:</b> Program error. The length of the answer area, as specified by ANSLEN, is not sufficient to contain answer area information. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Increase the size of the answer area provided for the request, and rerun your program. Depending on the macro version number, specify the length of the answer area as follows:</p> <ul style="list-style-type: none"> <li>• When the connection specified ASYNCXI=1 on the IXLCONN invocation and AXIOVERRIDE=0 was specified or defaulted to for the IXLCACHE request, the answer area length must be a minimum value of CAALEVEL2LEN</li> <li>• When the value of the macro version number (PLISTVER) is 4 or more, the minimum answer area length is CAALEVEL1LEN.</li> <li>• When the value of the macro version number (PLISTVER) is 0-3, the minimum answer area length is CAALEVEL0LEN.</li> </ul>

Table 33. Return and Reason Codes for the IXLCACHE REQUEST=DELETE\_NAME Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0849	<p><b>Equate Symbol:</b> IXLRNCODEBADRESTOKEN</p> <p><b>Meaning:</b> Program error. The restart token specified by RESTOKEN is not valid.</p> <p><b>Action:</b> Determine why the restart token cannot be used to resume the request. Possible causes are:</p> <ul style="list-style-type: none"> <li>• The specified token does not correspond to the restart token returned in the answer area of the previous request.</li> <li>• The user specified RESTOKEN when EXTRESTOKEN was required.</li> <li>• The user specified EXTRESTOKEN when RESTOKEN was required.</li> </ul> <p>For more information about premature request completion, see <a href="#">z/OS MVS Programming: Sysplex Services Guide</a>.</p>
8	xxxx0851	<p><b>Equate Symbol:</b> IXLRNCODENOSUSPENDISABLE</p> <p><b>Meaning:</b> Program error. The request failed because MODE=SYNCSUSPEND was specified, but the caller is disabled and cannot be suspended.</p> <p><b>Action:</b> Either specify another MODE value or become enabled (release the CPU lock); then reissue the request.</p>
8	xxxx0887	<p><b>Equate Symbol:</b> IXLRNCODEBADEXTRESTOKEN</p> <p><b>Meaning:</b> Program error. The extended restart token specified by EXTRESTOKEN is not valid. The specified token refers to an older instance of the target structure. A system-managed process occurred between the time a request returned the extended restart token and the time the connector tried to continue the request using that token.</p> <p><b>Action:</b> Discard the results of the initial request and reissue the request with an EXTRESTOKEN value of zero. For more information about restarting requests, see <a href="#">z/OS MVS Programming: Sysplex Services Guide</a>.</p>
C	xxxx0C06	<p><b>Equate Symbol:</b> IXLRNCODENOCONN</p> <p><b>Meaning:</b> Environmental error. No connectivity to the cache structure exists. This may occur due to operator commands such as VARY PATH,OFFLINE or CONFIG CHP,OFFLINE or hardware errors such as facility or path failures. The CONTOKEN is invalidated. The request fails.</p> <p><b>Action:</b> Either disconnect from the structure (using IXLDISC) or initiate a rebuild of the structure, if allowed (using IXLREBLD).</p>

Table 33. Return and Reason Codes for the IXLCACHE REQUEST=DELETE\_NAME Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx08B9	<p><b>Equate Symbol:</b> IXLRNOCODENOANSAREA</p> <p><b>Meaning:</b> An answer area was not specified when one is required. The requested service determined that conditions exist that require an ANSAREA to complete the request.</p> <p><b>Action:</b> Provide an answer area (ANSAREA) and answer area length (ANSLEN) on the IXLCACHE macro invocation for the request. ANSAREA is required when the connection specified ASYNCCI=1 on the IXLCONN invocation when connecting to the cache structure, AXIOVERRIDE=0 was specified or defaulted to for the IXLCACHE request and the request is one of the following:</p> <ul style="list-style-type: none"> <li>• CROSS_INVALID</li> <li>• CROSS_INVALIDLIST</li> <li>• DELETE_NAME</li> <li>• DELETE_NAMELIST</li> <li>• READ_DATA</li> <li>• REG_NAMELIST</li> <li>• WRITE_DATA</li> <li>• WRITE_DATA_LIST</li> </ul>
C	xxxx0C13	<p><b>Equate Symbol:</b> IXLRNOCODEREQPURGED</p> <p><b>Meaning:</b> Environmental error. The request was purged prior to completion of the request. Possible reasons include:</p> <ul style="list-style-type: none"> <li>• The connector failed.</li> <li>• The connector disconnected.</li> <li>• The requestor failed.</li> <li>• The request was purged by IXLPURGE.</li> <li>• Requests were purged when the connector provided an IXLEERSP response for the Rebuild Quiesce, Rebuild Stop, or Rebuild Cleanup event.</li> <li>• The connector invoked IXLREBLD REQUEST=COMPLETE.</li> <li>• The secondary address space was no longer valid.</li> </ul> <p><b>Action:</b> None if this is expected. Otherwise, determine why the connector failed.</p>
C	xxxx0C14	<p><b>Equate Symbol:</b> IXLRNOCODESTATUSUNKNOWN</p> <p><b>Meaning:</b> Environmental error. The IXLCACHE request has completed, but the final disposition of the request cannot be determined. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Verify the validity of your data by comparing the expected results with what is in the coupling facility.</p>

Table 33. Return and Reason Codes for the IXLCACHE REQUEST=DELETE\_NAME Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C25	<b>Equate Symbol:</b> IXLRSNCODESTRFAILURE <b>Meaning:</b> Environmental error. The cache structure failed prior to completion of the request. <b>Action:</b> Attempt to rebuild the structure using IXLREBLD, or disconnect from the structure using IXLDISC.
C	xxxx0CA0	<b>Equate Symbol:</b> IXLRSNCODEQUIESCEDSUSPENDFAIL <b>Meaning:</b> Environmental error. The request is failed because the structure is quiesced for a system-managed process and SUSPEND=FAIL is specified on the IXLCONN. <b>Action:</b> None, if this is expected.
C	xxxxFFFF	<b>Equate Symbol:</b> IXLRSNCODENOTAVAILABLE <b>Meaning:</b> Environmental error. XES functions are not available. The coupling facility hardware might not be present. <b>Action:</b> XES may not be used when XCF is running local mode. XES requires that the coupling facility be installed.
10	xxxx10xx	<b>Equate Symbol:</b> IXLRSNCODECOMPERROR <b>Meaning:</b> System error. XES processing failure. The state of the structure and the disposition of the request are unpredictable. The answer area (ANSAREA) fields do not contain valid information. <b>Action:</b> Save the reason code information, and contact the IBM support center.



## Chapter 36. IXLCACHE REQUEST=DELETE\_NAMELIST

### Description

A DELETE\_NAMELIST request allows you to specify a set of data entries that you want to delete from the structure. Each entry is represented by a name block, mapped by the IXLYDNNB macro. The name block contains the name of the structure entry for which delete processing is to be performed and optionally, a version comparison value used when version comparison is requested. The name blocks are contained in the storage area specified by BUFFER or the buffers specified by BUFLIST.

The system references the name blocks by an index into the buffer and processes them sequentially beginning with the name block identified by the first index (STARTINDEX) and ending with the data item identified by the last index (ENDINDEX). The name blocks are numbered starting with one.

For each structure entry, the DELETETYPE keyword indicates what specifically will be deleted. The DELETETYPE keyword provides options that allow you to delete the directory entry and associated data, only the changed data but not the directory entry, only the unchanged data but not the directory entry, or any entry (changed or unchanged) but not the directory data.

If any name block fails to identify an existing structure entry, processing either continues or terminates, depending on the value specified for the ERRORACTION keyword. If ERRORACTION=TERMINATE, processing stops and the index value of the name block that caused the problem is returned in the CAADNLINDEX field in the IXLYCAA answer area. If ERRORACTION=CONTINUE, processing continues with the next name block.

Each name block can optionally contain a comparative version number that is to be compared with the version number associated with the cache structure entry identified in the name block. The VERSCOMPTYPE keyword is used to define how the comparison is to be performed. If a version comparison mismatch occurs, then processing will either terminate or continue depending on the value specified for the ERRORACTION keyword. If ERRORACTION=TERMINATE is specified, processing stops and the index value of the name block that caused the problem is returned in the CAADNLINDEX field in the IXLYCAA answer area. If ERRORACTION=CONTINUE is specified, processing continues with the next name block.

A DELETE\_NAMELIST can complete prematurely because the request exceeds the time-out criteria for the coupling facility (time-out criteria is model-dependent) or a halt condition is found as a result of specifying HALTONCHANGED=YES. When a request completes prematurely, the system returns an index value (CAADNLINDEX) in the answer area which you can use to restart the DELETE\_NAMELIST request. The index value returned when a DELETE\_NAMELIST request completes prematurely is the index of the first unprocessed name block (or in the case of when the request was halted because a structure entry met the HALTONCHANGED criteria, the index of the name block that caused the request to halt).

A DELETE\_NAMELIST request can be issued only for cache structures allocated in a coupling facility of CFLEVEL=5 or higher. DELETE\_NAMELIST requests issued for a cache structure allocated in a coupling facility of CFLEVEL=0 through 4 will fail.

For DELETETYPE=DIRANDDATA, a check of the changed status of data and cast-out lock state for a structure entry may be requested prior to processing the structure entry by specifying the HALTONCHANGED=YES keyword. If a structure entry is found to either contain changed data or for which the cast-out lock is currently held, then processing of the DELETE\_NAMELIST request is halted and the system returns the index value (CAADNLINDEX) of the name block meeting the halt criteria in the answer area. When the DELETE\_NAMELIST request is halted as per requested, the application is expected to take some action to change the state of the indicated structure entry data to "unchanged" (that is, the cast-out lock is not held and the status of the data is unchanged) before resuming the request. One such action could be to read the entry data for castout, write the data to permanent storage, then reset the cast-out lock to the not-held state. After taking such action, the DELETE\_NAMELIST request may be started at the

element in the list of name elements that originally caused the request to halt by setting input parameter STARTINDEX to CAADNLINDEX.

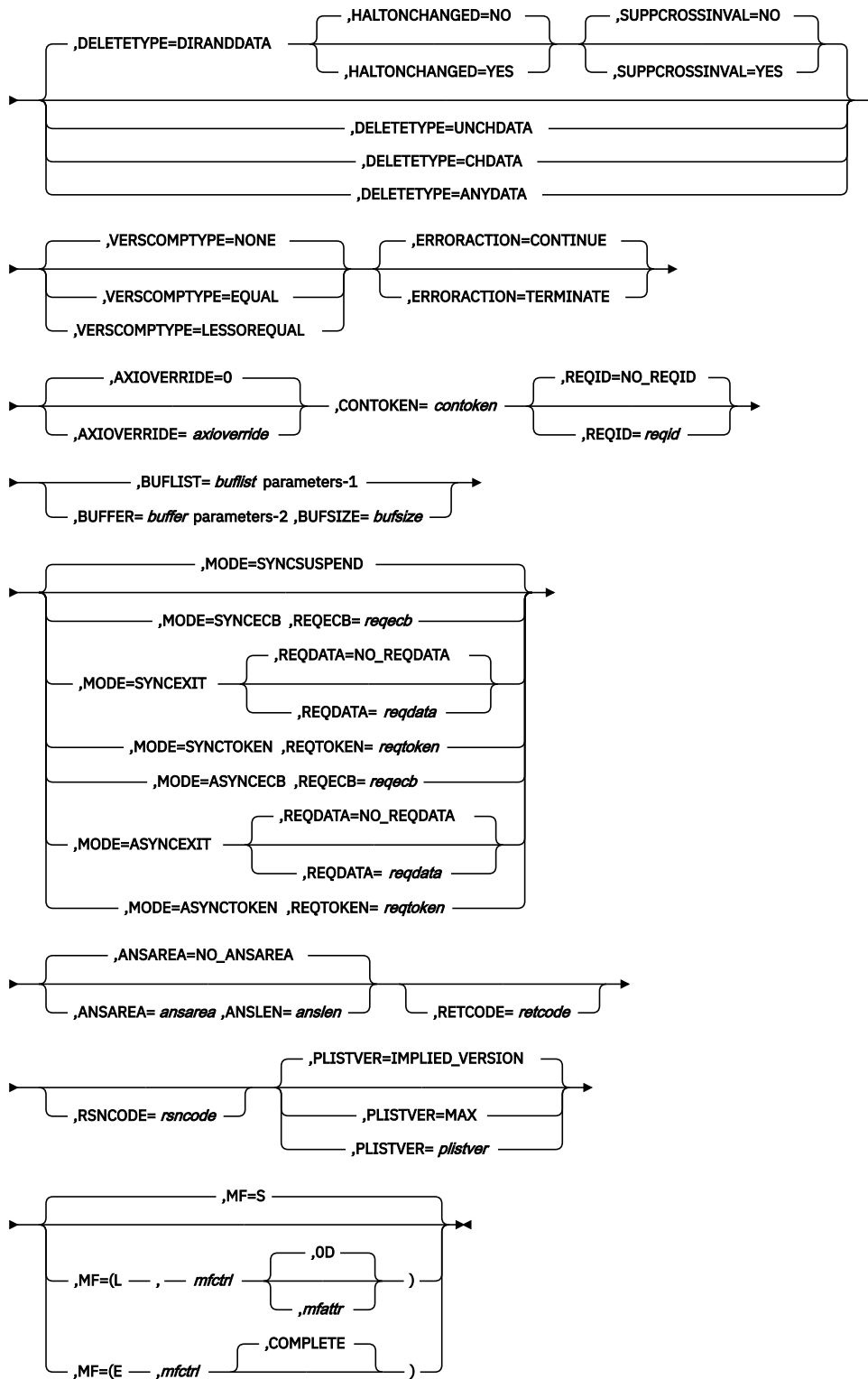
### Syntax Diagram

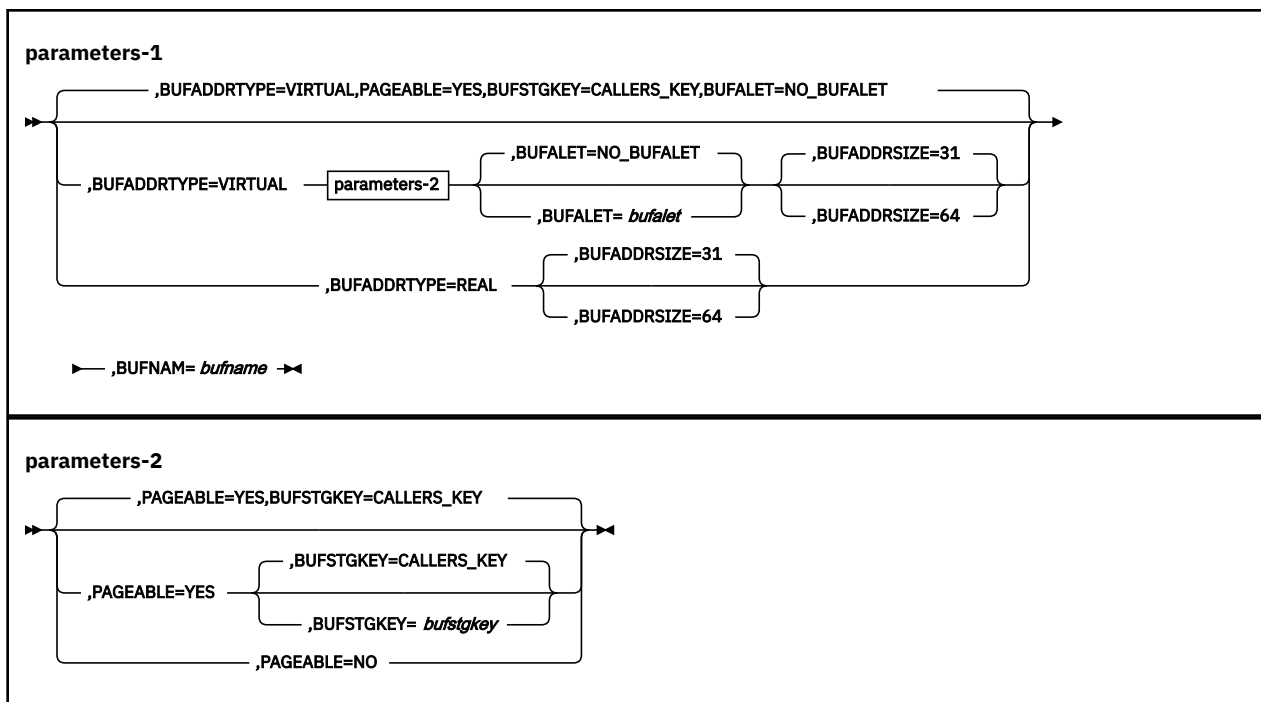
---

The syntax for the IXLCACHE REQUEST=DELETE\_NAMELIST is as follows:

## main diagram

➔ IXLCACHE — REQUEST=DELETE\_NAMELIST — ,STARTINDEX= *startindex* — ,ENDINDEX= *endindex* ➔





**Note:** You must specify **ANSAREA=ansarea**, **ANSLEN=anslen** if you:

- Specify **MODE=SYNCTOKEN** or **MODE=ASYNCTOKEN**, or
- Specify **AXIOVERRIDE=0** (or default to 0) and specified **ASYNCXI=1** on the **IXLCONN** invocation when connecting to the cache structure.

## Parameter Descriptions

The parameter descriptions for **REQUEST=DELETE\_NAMELIST** are listed in alphabetical order. Default values are underlined:

### **REQUEST=DELETE\_NAMELIST**

Use this input parameter to specify that you want to delete a set of entries from the structure. Each entry is represented by a name block, mapped by the **IXLYDNNB** macro, stored in the area specified by **BUFFER** or the buffers specified by **BUKLIST**.

#### **,ANSAREA=NO\_ANSAREA**

#### **,ANSAREA=ansarea**

Use this output parameter to specify an answer area to contain information returned from the request. The format of the answer area is described by the **IXLYCAA** mapping macro.

An asynchronous cross-invalidation sequence number (**CAAASYNXISEQNUM**) is returned from a request that initiates cross-invalidates of local caches asynchronously to the completion of the request.

See the Chapter 29, “IXLAXISN,” on page 449 service for a description of how to use the returned **CAAASYNXISEQNUM** to determine when cross-invalidates of local caches associated with the request have completed.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an area (with a length of **ANSLEN**) where the answer area information returned from the request will be stored.

#### **,ANSLEN=anslen**

Use this inppaameter to specify the size of the storage area specified by **ANSAREA**.

Use either **CAALEVEL0LEN**, **CAALEVEL1LEN** or **CAALEVEL2LEN** of the **IXLYCAA** mapping macro to determine the minimum size of the answer area. The answer area length must be at least large enough to accommodate the level of the **IXLYCAA** mapping appropriate to the requested function.

- When the connection specified ASYNXCI=1 on the IXLCONN invocation and AXIOVERRIDE=0 was specified or defaulted to for the IXLCACHE request, the answer area length is a required parameter and must be a minimum value of CAALEVEL2LEN to contain a returned asynchronous cross-invalidation sequence number (CAAASYNXISEQNUM).
- When the value of PLISTVER is 4 or above, the minimum answer area length is CAALEVEL1LEN.
- When the value of PLISTVER is 0 - 3, the minimum answer area length is CAALEVEL0LEN.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12 2 to ) of a 2-byte field that contains the length of the answer area (ANSAREA).

#### **,AXIOVERRIDE=0**

#### **,AXIOVERRIDE=***axiooverride*

Use this input parameter to specify whether the asynchronous cross-invalidation control setting of IxlConnAsyncXiYes (1) for the connection identified by CONTOKEN should be overridden for this request. Valid values are 0 (IxlCacheAXiOverrideNo) or 1 (IxlCacheAXiOverrideYes).

The AXIOVERRIDE keyword is meaningful to processing only when the connection specified ASYNXCI=1 on the IXLCONN invocation when connecting to the cache structure and the cache structure is allocated in a CFLEVEL=23 or higher coupling facility.

A value of 0 (IxlCacheAXiOverrideNo) indicates that the asynchronous cross-invalidation control for the connection as specified on the IXLCONN invocation should be used for the request. Cross-invalidation against local caches for this request will preferentially be initiated asynchronously to the completion of the request when asynchronous cross-invalidations are supported by the coupling facility where the cache structure is allocated.

A value of 1 (IxlCacheAXiOverrideYes) indicates that the ASYNXCI specification of IxlConnAsyncXiYes (1) by the connector on the IXLCONN invocation should be overridden for this request only. Cross-invalidations generated by this request will be processed synchronously to the completion of the request.

Any value other than 0 or 1 for AXIOVERRIDE will have the same behavior as specifying a value of 0 (IxlCacheAXiOverrideNo).

If cross-invalidation against local caches for this request were initiated asynchronously to the completion of the request, an asynchronous cross-invalidation sequence number is returned in CAAASYNXISEQNUM of the cache answer area (ANSAREA).

The asynchronous cross-invalidation sequence number can be used on a subsequent invocation of IXLAXISN to ensure that the asynchronous cross-invalidations associated with this request have completed.

When cross-invalidations are initiated synchronously to the completion of the request or no cross-invalidations occurred for the request, no asynchronous cross-invalidation sequence number is returned.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12 2 to ) of a one-byte input field that contains the value indicating whether the asynchronous cross-invalidation control setting of the connector should be overridden

#### **,BUFADDRSIZE=31**

#### **,BUFADDRSIZE=64**

Use this input parameter to specify whether a 31-bit or a 64-bit address is specified by a BUFLIST entry.

#### **31**

The entry in BUFLIST is 31 bits in size.

#### **64**

The entry in BUFLIST is 64 bits in size.

**,BUFADDRTYPE=VIRTUAL****,BUFADDRTYPE=REAL**

Use this input parameter to specify whether the buffer addresses specified in the BUFLIST list are virtual storage or real storage addresses.

**VIRTUAL**

The buffer addresses are virtual storage addresses. The virtual storage can be pageable or nonpageable. See the PAGEABLE parameter for information about managing storage binds when specifying virtual storage addresses.

**REAL**

The buffer addresses are real storage addresses.

It is the caller's responsibility to manage the binds between the data buffer virtual storage and the real storage addresses provided. The caller must ensure that the data buffer virtual storage remains bound to the real storage addresses provided until the request completes.

**,BUFALET=NO\_BUFALET****,BUFALET=bufalet**

Use this input parameter to specify an access list entry token (ALET) to be used in referencing all of the buffers specified by BUFLIST.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 4-byte field that contains the ALET.

**,BUFFER=buffer**

Use this input parameter to specify a buffer area to contain the returned name blocks that define the entries that are to be deleted.

Only 31-bit addressable virtual storage areas (below 2GB) are supported by the BUFFER specification. High virtual storage areas (above 2GB) can only be specified via the BUFLIST specification.

The format of the registration block is described by mapping macro IXLYDNNB.

You can define the buffer size to be a total of up to 65536 bytes, but it should not be larger than what you actually require to hold the maximum number of name blocks, subject to the other buffer length requirements stated as follows.

Other requirements depend on the size you select:

- If you specify a buffer size of less than or equal to 4096 bytes, you must ensure that the buffer:
  - Is 256, 512, 1024, 2048, or 4096 bytes.
  - Starts on a 256-byte boundary.
  - Does not cross a 4096-byte boundary.
  - Does not start below storage address 512.
- If you specify a buffer size of greater than 4096 bytes, you must ensure that the buffer:
  - Is a multiple of 4096 bytes.
  - Is less than or equal to 65536 bytes.
  - Starts on a 4096-byte boundary.
  - Does not start below storage address 512.

See the BUFSIZE parameter description for defining the size of the buffer. See [z/OS MVS Programming: Sysplex Services Guide](#) for more information on buffers.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an area (with a length of BUFSIZE) to contain the name blocks.

**,BUFINCRNUM=bufincrnum**

Use this input parameter to specify the number of 256-byte segments comprising each buffer in the BUFLIST list.

Valid BUFINCRNUM values are 1, 2, 4, 8, or 16, which correspond to BUFLIST buffer sizes of 256, 512, 1024, 2048, and 4096 bytes respectively.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 1-byte field that contains 1, 2, 4, 8, or 16.

#### **,BUFLIST=buflist**

Use this input parameter to specify a list of buffers to hold the name blocks identifying the entries to be deleted from the structure. BUFLIST specifies a 128-byte storage area that consists of a list of 1 to 16 buffer address elements.

The format of the name block is described by mapping macro IXLYDNNB.

Either 31-bit addressable (below 2GB) or 64-bit addressable (above 2GB) real or virtual storage areas are supported for the BUFLIST specification, depending on the specification for the BUFADDRTYPE and BUFFADDRSIZE keywords. However, pageable high shared virtual storage areas (above 2GB) may not be used.

The format of the list is a set of 8-byte elements. The BUFADDRSIZE keyword denotes whether four or eight bytes of the element are used.

- If BUFADDRSIZE=31 is specified, then the first four bytes of each element are reserved space and the last four bytes contain the address of the buffer.
- If BUFADDRSIZE=64 is specified, then the full eight bytes specify the address of the buffer.

#### **The BUFLIST buffers must:**

- Reside in the same address space or data space as defined by BUFALET.
- Be the same size: either 256, 512, 1024, 2048, or 4096 bytes as defined by BUFINCRNUM.
- Start on a 256-byte boundary and not cross a 4096-byte boundary.
- Not start below storage address 512.

**Note:** The buffers do not have to be contiguous in storage. XES treats BUFLIST buffers as a single buffer even if the buffers are not contiguous.

See the BUFNUM and BUFINCRNUM keyword descriptions for specifying the number and size of buffers.

See *z/OS MVS Programming: Sysplex Services Guide* for more information on buffers.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 128-byte field that contains a list of buffer addresses.

#### **,BUFNUM=bufnum**

Use this input parameter to specify the number of buffers in the BUFLIST list. Valid BUFNUM values are from 1 to 16.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 1-byte field that contains the number of buffers (1 to 16) in the list (BUFLIST)

#### **,BUFSIZE=bufsize**

Use this input parameter to specify the size of the BUFFER area. See the BUFFER parameter description for valid buffer sizes.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a fullword field that contains the size of the buffer (BUFFER) in bytes.

#### **,BUFSTGKEY=CALLERS\_KEY**

#### **,BUFSTGKEY=bufstgkey**

Use this input parameter to specify a storage key that you define and use when referencing the buffer specified by BUFFER.

If you do not specify BUFSTGKEY, or if you specify BUFSTGKEY=CALLERS\_KEY, all references to the buffer are in the caller's PSW key.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 1-byte field that contains the storage key in the format kkkkxxxx, where kkkk is the key and xxxx is ignored.

**,CONTO=conken**

Use this input parameter to specify the connect token that was returned by the IXLCONN service in the IXLCONN answer area, which is mapped which is by IXLYCONA. The connect token uniquely identifies your connection to the cache structure, and must be specified on each IXLCACHE invocation.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field that contains the connect token.

**DELETETYPE=DIRANDDATA**

**DELETETYPE=UNCHNDATA**

**DELETETYPE=CHDATA**

**DELETETYPE=ANYDATA**

Use this input parameter to specify the type of delete processing that is to be performed.

**DIRANDDATA**

For each applicable structure entry, invalidate the name and remove the name from storage and cast-out classes, and release all directory and data entry resources for the structure for reuse by the structure. The system deregisters interest in all connections with registered interest in the entry and, optionally, performs a cross-invalidate against their local caches.

Cross-invalidation against local caches may be suppressed to improve overall coupling facility performance and to quicken the completion of DELETE\_NAMELIST requests.

**UNCHNDATA**

For each applicable structure entry, if the data is unchanged, release the data entry resources for the entry for reuse by the structure. The system does not delete the directory entry and does not perform a cross-invalidate.

**CHDATA**

For each applicable structure, if the data is changed, release the data entry resources for the entry for reuse by the structure, set the change bit, the castout lock, and the castout lock state to zero, and remove the directory entry from the cast-out class. The system does not delete the directory entry and does not perform a cross-invalidate.

**ANYDATA**

For each applicable structure, whether the data is changed or unchanged, release the data entry resources for the entry for reuse by the structure. If the data is changed, set the change bit, the castout lock, and the castout lock state to zero, and remove the directory entry from the cast-out class. The system does not delete the directory entry and does not perform a cross-invalidate.

**,ENDINDEX=endindex**

Use this input parameter to specify the ending index for name block processing. The index value must be greater than or equal to the value specified for STARTINDEX.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 2-byte field containing the ending index value.

**ERRORACTION=CONTINUE**

**ERRORACTION=TERMINATE**

Use this input parameter to specify whether processing is to continue with the next name block if an entry is not found or a version mismatch occurs.

**CONTINUE**

If an error occurs, processing is to continue with the next name block.

**TERMINATE**

If an error occurs, processing is to halt and the index of the name block that caused the error is returned in the answer area.



**HALTONCHANGED=NO****HALTONCHANGED=YES**

Use this input parameter to specify whether the DELETE\_NAMELIST request should be halted when a structure entry in the set of entries to be deleted is encountered that either contains changed data or for which the cast-out lock is currently held.

HALTONCHANGED is only meaningful when delete operations are issued to cache structures allocated in a coupling facility that supports request halting based on data changed status and cast-out lock state.

**HALTONCHANGED=NO**

The DELETE\_NAMELIST request should continue processing a structure entry which contains changed data or for which the cast-out lock is held.

**HALTONCHANGED=YES**

The DELETE\_NAMELIST request should be halted if a structure entry contains changed data or for which the cast-out lock is held. The structure entry is not deleted. Specifying HALTONCHANGED=YES can prevent the inadvertent loss of changed data that has not yet been cast out to permanent storage. When the request is halted due to the presence of a structure entry that meets the halt criteria, information identifying the structure entry is returned in the answer area specified by ANSAREA.

HALTONCHANGED=YES is intended to prevent the inadvertent loss of changed data that has not yet been cast out to permanent storage. When the DELETE\_NAMELIST request is halted, the application is expected to take some action to change the state of the indicated structure entry data to "unchanged" (that is, the cast-out lock is not held and the status of the data is unchanged) before resuming the request by setting the STARTINDEX keyword to CaaDNLIndex. One such action could be to read the entry data for castout, write the data to permanent storage, then reset the cast-out lock to the not-held state.

Restarting the DELETE\_NAMELIST request before taking such an action to change the state of the structure entry data to "unchanged" may result in the request halting on the same structure entry again thus preventing the request from proceeding further to process all intended structure entries.

When the request is halted due to the presence of a structure entry that meets the halt criteria, the request will complete with a return code IXLRETCODEWARNING, reason code IXLRSNCODEHALTCHANGEDDATA and the answer area specified by ANSAREA will contain specific information to identify the entry that was found to be in the "changed" state.

**,MF=S**

**,MF=(L,mfctrl)**

**,MF=(L,mfctrl,mfattr)**

**,MF=(L,mfctrl,0D)**

**,MF=(M,mfctrl)**

**,MF=(M,mfctrl,COMPLETE)**

**,MF=(M,mfctrl,NOCHECK)**

**,MF=(E,mfctrl)**

**,MF=(E,mfctrl,COMPLETE)**

**,MF=(E,mfctrl,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into

the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

**,mfctrl**

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

**,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

**,COMPLETE**

**,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

**COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if *SMILE=var* were an optional parameter and the default is *SMILE=NO\_SMILE* then it would not be documented. However, if the default was *SMILE=-*), then it would be documented because a value would be the default.

**NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

**,MODE=SYNCSUSPEND**

**,MODE=SYNCECB**

**,MODE=SYNCEXIT**

**,MODE=SYNCTOKEN**

**,MODE=ASYNCECB**

**,MODE=ASYNCEXIT**

**,MODE=ASYNCTOKEN**

Use this input parameter to specify:

- Whether the request is to be performed synchronously or asynchronously
- How you want to be notified of request completion if the request is processed asynchronously.

See *z/OS MVS Programming: Sysplex Services Guide* for more information on understanding synchronous and asynchronous cache operations.

**SYNCSUSPEND**

The request processes synchronously. If necessary, the request is suspended until it can complete synchronously. To use this option, your program must be enabled for I/O and external interrupts.

**SYNCECB**

The request processes synchronously if possible. If the request processes asynchronously, the ECB specified by REQECB is posted when the request completes.

**SYNCEXIT**

The request processes synchronously if possible. If the request processes asynchronously, your complete exit is given control when the request completes.

**SYNCTOKEN**

The request processes synchronously if possible. If the request processes asynchronously, an asynchronous request token is returned to the area specified by REQTOKEN. Use the returned request token on the IXLFComp macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=SYNCTOKEN is specified.

**ASYNCECB**

The request processes asynchronously. The ECB specified by REQECB is posted when the request completes.

**ASYNCEXIT**

The request processes asynchronously. Your complete exit is given control when the request completes.

**ASYNCTOKEN**

The request processes asynchronously. An asynchronous request token is returned to the area specified by REQTOKEN. Use the returned request token on the IXLFCOMP macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=ASYNCTOKEN is specified.

**,PAGEABLE=YES****,PAGEABLE=NO**

Use this input parameter to identify whether the storage areas specified by BUFFER or BUFLIST are in pageable or potentially pageable storage.

**YES**

Specify this option to indicate that the BUFFER or BUFLIST buffers reside in pageable virtual storage. XES performs the required page fixing to fix the buffers in real storage while the cache or list request transfers data to or from the coupling facility.

This includes storage obtained from pageable subpools, disabled reference (DREF) storage, and may include storage that has the potential to become pageable during the processing of a request. (An example is address space storage owned by any swappable address space, for which a PGSER FIX has been successfully processed, but for which the owning address space gets swapped during processing of a cache or list request.) This does not include implicitly non-pageable storage (for example, storage obtained from non-pageable subpools).

The system takes responsibility for managing binds to central storage for the duration of the cache or list request, regardless of what address space owns the storage or whether the storage-owning address space is swappable or nonswappable. The storage can be owned by any address space.

High shared virtual storage areas (above 2GB) may not be used.

**NO**

Specify this option to indicate that the BUFFER or BUFLIST buffers reside in non-pageable virtual storage. XES does not page fix the buffers in real storage.

This includes implicitly non-pageable storage areas (for example, storage obtained from non-pageable subpools), and may include storage that has the potential to become pageable during the processing of a request (An example is address space storage owned by any swappable address space, for which a PGSER FIX has been successfully processed, but for which the owning address space gets swapped-out during processing of a cache or list request.)

The system takes responsibility for managing binds to central storage for the duration of the cache or list request, if and only if the non-pageable storage is owned by either the requester's address space or the connector's address space. If the storage is owned by any other address space, then the invoker is responsible for ensuring that the virtual storage remains non-pageable for the duration of the request (including the case in which the storage is owned by a swappable address space that is swapped during processing of an IXLCACHE or IXLLIST request). Subject to this consideration, the storage can be owned by any address space. See [z/OS MVS Programming: Sysplex Services Guide](#).

**,PLISTVER=IMPLIED\_VERSION****,PLISTVER=MAX****,PLISTVER=plistver**

Use this input parameter to specify the version of the macro. See [“Understanding IXLCACHE Version Support” on page 461](#) for a description of the options available with PLISTVER.

**,REQDATA=NO\_REQDATA****,REQDATA=reqdata**

Use this input parameter with MODE=SYNCEXIT or MODE=ASYNCEXIT to pass any data you choose to the complete exit. The exit will get control only if the request is processed asynchronously.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the data to be passed to the complete exit.

**,REQECB=reqecb**

Use this output parameter with either MODE=SYNCECB or MODE=ASYNCECB to specify the address of an ECB, which is to be posted when the request completes if the request was processed asynchronously.

Before coding REQECB, you must ensure that:

- You initialize the ECB before you issue the request.
- The ECB resides in either common storage or the home address space where IXLCONN was issued.
- Any tasks that wait for the ECB to be posted reside in the home address space where IXLCONN was issued.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that contains the address of the ECB to be posted when the request completes. The ECB must be aligned on a fullword boundary.

**,REQID=NO\_REQID****,REQID=reqid**

Use this input parameter to specify a user-defined request identifier to be associated with the request. You can specify this request identifier on the IXPURGE macro to cancel a request that has not yet been processed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the user-defined request identifier.

**,REQTOKEN=reqtoken**

Use this output parameter with either MODE=SYNCTOKEN or MODE=ASYNCTOKEN to specify the address of a storage area to receive the request token that is returned when the request will be processed asynchronously. This token, which uniquely identifies the request, must be used as input to the IXLFCOMP macro, which you use to determine if the request has completed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field where the system will put the request token.

**,RETCODE=retcode**

Use this output parameter to specify a field to contain the return code. (The return code is also returned in register 15.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the return code when the request has completed.

**,RSNCODE=rsncode**

Use this output parameter to specify a field to contain the reason code returned, if applicable. (The reason code is also returned in register 0.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the reason code (if any) when the request has completed.

**,STARTINDEX=startindex**

Use this input parameter to specify the starting index for name block processing. Valid STARTINDEX values are from 1 to the value of ENDINDEX. The first registration block has index number 1.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 2-byte field containing the starting index value.

**SUPPCROSSINVAL=NO****SUPPCROSSINVAL=YES**

Use this input parameter to specify whether cross-invalidate processing associated with the DELETETYPE=DIRANDDATA request should be suppressed.

SUPPCROSSINVAL is only meaningful when delete operations are issued to cache structures allocated in a coupling facility that supports suppressing cross invalidation processing.

**SUPPCROSSINVAL=NO**

Cross-invalidation should be performed (not suppressed) against the local caches of all connections with registered interest in a successfully deleted structure entry.

**SUPPCROSSINVAL=YES**

Cross-invalidation should not be performed (suppressed) against the local caches of all connections with registered interest in a successfully deleted structure entry. Cross-invalidation signals associated with the cross-invalidation of local caches are not sent which improves overall performance by reducing coupling facility link traffic, saving CPU cycles and quickening the completion of the DELETE\_NAMELIST request.

When requesting that the system suppress cross-invalidation of the local cache of a deleted data item, the IXLVECTR macro cannot be used to check the validity of that local cache data item using the vector index that had been associated with the data item. The vector index will be in an indeterminate state until it is subsequently reused for a new data item. The application must take its own measures to ensure the data integrity of the local cache buffers and the subsequent reuse of the associated vector indexes.

**VERSCOMPTYPE=NONE****VERSCOMPTYPE=EQUAL****VERSCOMPTYPE=LESSOREQUAL**

Use this input parameter to specify how the structure entry version number comparison is to be performed.

**VERSCOMPTYPE=NONE**

No version number is provided for comparison, therefore a version number comparison is not to be performed.

**VERSCOMPTYPE=EQUAL**

The version number for the structure entry must be equal to the value specified in the corresponding name block mapped by IXLYDNBB.

**VERSCOMPTYPE=LESSOREQUAL**

The version number for the structure entry must be less than or equal to the value specified in the corresponding name block mapped by IXLYDNBB.

## ABEND Codes

---

Abend X'026' (See [z/OS MVS Programming: Sysplex Services Guide](#) for more information on this abend.)

## Return and Reason Codes

---

When the system returns control to your program:

- GPR 15 (and RETCODE, if specified) contains a return code.
- If the return code is not zero, GPR 0 (and RSNCODE, if specified) contains a reason code.

**Note:** The return and reason codes will also be put into the answer area (mapped by IXLYCAA), if there is one, when the request completes.

The IXLYCON macro provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

**0**

IXLRETCODEOK

4

IXLRETCODEWARNING

8

IXLRETCODEPARMERROR

C

IXLRETCODEENVERROR

10

IXLRETCODECOMPERROR

The following table contains hexadecimal return and reason codes, the equate symbols associated with each reason code, and the meaning and suggested action for each return and reason code.

Table 34. Return and Reason Codes for the IXLCACHE REQUEST=DELETE_NAMELIST Macro		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0	None.	<p><b>Meaning:</b> If you specified a MODE value of ASYNCECB, ASYNCEXIT, or ASYNCTOKEN, the request has been successfully initiated. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed. For any other MODE value, the request has successfully completed.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=ASYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=ASYNCEXIT, the connection's complete exit will be given control when the request has completed.</li> <li>• If you specified MODE=ASYNCTOKEN, use REQTOKEN when calling IXLFCOMP to determine when the request has completed.</li> </ul>

Table 34. Return and Reason Codes for the IXLCACHE REQUEST=DELETE\_NAMELIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	xxxx0402	<p><b>Equate Symbol:</b> IXLRNCODEASYNCH</p> <p><b>Meaning:</b> The request could not be processed synchronously. It will be processed asynchronously. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed.</p> <ul style="list-style-type: none"> <li>• If you specified MODE=SYNCECB, the ECB specified in REQECB will be posted when the request has finished.</li> <li>• If you specified MODE=SYNCEXIT, the connection's complete exit will be given control when the request has finished.</li> <li>• If you specified MODE=SYNCTOKEN, the REQTOKEN contains an asynchronous request token that may be used on the IXLFCOMP macro to determine when the request has completed.</li> </ul> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=SYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=SYNCEXIT, the connection's completion exit will be given control when the request has completed.</li> <li>• If you specified MODE=SYNCTOKEN, use REQTOKEN when calling IXLFCOMP to determine when the request has completed.</li> </ul>
4	xxxx0409	<p><b>Equate Symbol:</b> IXLRNCODETIMEOUT</p> <p><b>Meaning:</b> The request has completed prematurely because the model-dependent time-out criteria of the coupling facility has been exceeded. The index of the next registration block to be processed has been returned in the answer area (field CAARNLINDEX).</p> <p><b>Action:</b> Reissue the request.</p> <p>Be sure to process the information returned from this request before reissuing the request. The data returned from this request will be overwritten if you specify the same buffer address. Continue to reissue the request until the return code indicates that all processing has completed.</p> <p>The entries that were processed are indexed from STARTINDEX to CAARNLINDEX-1.</p> <p>To restart the request, update STARTINDEX with the value of CAARNLINDEX, the index of the next name element to be processed. For more information about premature request completion, see <i>z/OS MVS Programming: Sysplex Services Guide</i>.</p>

Table 34. Return and Reason Codes for the IXLCACHE REQUEST=DELETE\_NAMELIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	xxxx042A	<p><b>Equate Symbol:</b> IXLRNCODEHALTCHANGEDDATA</p> <p><b>Meaning:</b> A DELETE_NAMELIST request with HALTONCHANGED=YES specified was halted due to a structure entry being found to have data in the changed state or having a cast-out lock held. The index of the name block in the element list that met the halt criteria as well as cast-out information for the entry has been returned in the answer area. All prior entries have been processed.</p> <p><b>Action:</b> Since delete processing was requested to be halted when changed data or data locked for cast-out was encountered, the application should treat the deletion attempt as an error condition and prior to resuming the request, write the data to permanent storage and change the state of the indicated structure entry data to "unchanged" (i.e. the cast-out lock is not held and the status of the data is unchanged) before resuming the request. The index value of the name within the specified list of name blocks that caused the request to be halted has been returned in the answer area. After taking action, the request may be started at the entry that caused the request to halt by setting the STARTINDEX parameter to CAADNLINDEX.</p>
8	xxxx0801	<p><b>Equate Symbol:</b> IXLRNCODEBADPARMLIST</p> <p><b>Meaning:</b> The parameter list (mfctrl) for this request is not addressable. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the parameter list address.</li> <li>• The parameter list must be addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are issuing this macro while disabled, the parameter list must reside in either page-fixed or DREF storage.</li> <li>• If you are issuing this macro in AR-mode and you specified the parameter list address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are issuing this macro in AR-mode, SYSSTATE ASCENV=AR must be issued before this macro.</li> </ul>



Table 34. Return and Reason Codes for the IXLCACHE REQUEST=DELETE\_NAMELIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0804	<p><b>Equate Symbol:</b> IXLRSNCODEBADVERSION#</p> <p><b>Meaning:</b> The version number in the macro parameter list is not compatible with the level of XES currently being used. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that your program did not overlay the parameter list storage.</li> <li>• Verify that your program was assembled with the correct macro library for the release of MVS on which your program is running.</li> <li>• Verify that your program is running on an MVS system that supports the version of the macro you are using.</li> </ul>

Table 34. Return and Reason Codes for the IXLCACHE REQUEST=DELETE\_NAMELIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx080A	<p><b>Equate Symbol:</b> IXLRNCODEBADCONTOKEN</p> <p><b>Meaning:</b> Program error. The specified connect token (CONTOKEN) was not valid for one of the following reasons:</p> <ol style="list-style-type: none"> <li>1. The user with the connection identifier represented by the token has disconnected from the structure.</li> <li>2. The connector's task (the task that issued IXLCONN) ended.</li> <li>3. The specified token is not the token that was returned from IXLCONN.</li> <li>4. The request was issued from an address space other than the address space in which IXLCONN was issued.</li> <li>5. The connect token was invalidated during rebuild.</li> <li>6. The connect token was invalidated by XES.</li> </ol> <p><b>Note:</b> The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Verify that the CONTOKEN value specified was valid and for the correct structure. The numbers on the following actions correspond to the numbers above.</p> <ol style="list-style-type: none"> <li>1. Discontinue use. Perform recovery and cleanup for the structure and check protocol for use of structure to determine why you did not know of the disconnection.</li> <li>2. Discontinue use. Perform recovery and cleanup for the structure and check protocol for use of structure to determine why you did not know that the task ended.</li> <li>3. Issue IXLCONN to connect to the structure. The contoken is returned in the CONACONTOKEN field of the answer area specified on the IXLCONN request.</li> <li>4. Issue your request from the same address space the IXLCONN was issued.</li> <li>5. Participate in the rebuild. When it is complete, try again.</li> <li>6. Discontinue use. Perform recovery and cleanup for the structure and check protocol for use of structure.</li> </ol> <p>You may want to issue IXCQUERY to get more information about the structure.</p>
8	xxxx0819	<p><b>Equate Symbol:</b> IXLRNCODEBADVECTOROP</p> <p><b>Meaning:</b> The vector index in the registration block indexed by CAARNLINDEX is not valid. None of the registration blocks have been processed. All vector indexes for all registration blocks in the buffer have been set to indicate that the local buffer is not valid.</p> <p><b>Action:</b> Correct the vector index value and reissue the REG_NAMELIST request with the same STARTINDEX and ENDINDEX values.</p>

Table 34. Return and Reason Codes for the IXLCACHE REQUEST=DELETE\_NAMELIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0824	<p><b>Equate Symbol:</b> IXLSRNCODEWRONGSTRTYPE</p> <p><b>Meaning:</b> The connect token specified by CONTOKEN is not to a cache structure.</p> <p><b>Action:</b> Verify the connect token for this cache structure.</p> <p><b>Note:</b> The connect token can be found in the IXLCONN answer area. Verify the value specified on the TYPE= parameter of the IXLCONN request for this structure.</p>
8	xxxx082D	<p><b>Equate Symbol:</b> IXLSRNCODEBADSTGCLASS</p> <p><b>Meaning:</b> Program error. A REG_NAMELIST registration block specified a storage class value that exceeded the maximum defined storage class for the structure. CAARNLINDEX contains the index of the registration block that contained the storage class value in error. All registration blocks preceding this block were processed.</p> <p><b>Action:</b> Correct the storage class value in the registration block indexed by CAARNLINDEX and reissue the REG_NAMELIST request starting with that registration block.</p>
8	xxxx0833	<p><b>Equate Symbol:</b> IXLSRNCODEBADPGBLATTR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER is specified as being pageable (PAGEABLE=YES) but is not.</p> <p><b>Action:</b> Change the buffer area to pageable storage, or specify PAGEABLE=NO. See the PAGEABLE parameter description for specification instructions.</p>
8	xxxx0834	<p><b>Equate Symbol:</b> IXLSRNCODEBADNONPGBLATTR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER is specified as being nonpageable (PAGEABLE=NO) but is either pageable or not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The type of storage in the buffer area corresponds to the value you specified on the PAGEABLE parameter. See the PAGEABLE parameter description for specification instructions.</li> <li>• The correct buffer address was used.</li> <li>• The buffer area was not previously freed.</li> <li>• If the caller is running in AR-mode and the BUFFER parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> </ul>

Table 34. Return and Reason Codes for the IXLCACHE REQUEST=DELETE\_NAMELIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0835	<p><b>Equate Symbol:</b> IXLRNCODEBADDATAADDR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER is not addressable. All bits in the local cache vector may be reset to overindicate that the data items are not valid.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The correct buffer address was used for BUFFER.</li> <li>• The buffer area was not previously freed.</li> <li>• The buffer area was allocated in a storage key that matches the key specified using the BUFSTGKEY parameter, or, if BUFSTGKEY is omitted, the caller's PSW key.</li> <li>• The buffer area is addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If the caller is running in AR-mode and the BUFFER parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If the caller is running in AR-mode, SYSSTATE ASCENV=AR must be specified before issuing this macro.</li> </ul>
8	xxxx0838	<p><b>Equate Symbol:</b> IXLRNCODEBADANSAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by ANSAREA is not addressable.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the ANSAREA address.</li> <li>• ANSAREA must be addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are calling IXLCACHE while disabled, ANSAREA must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLCACHE in AR-mode and you specified the ANSAREA address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are calling IXLCACHE in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLCACHE macro.</li> </ul>

Table 34. Return and Reason Codes for the IXLCACHE REQUEST=DELETE\_NAMELIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0839	<p><b>Equate Symbol:</b> IXLRNCODEBADREQTOKENAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by REQTOKEN is not addressable.</p> <p><b>Action:</b> Ensure that the request token area specified by REQTOKEN is valid:</p> <ul style="list-style-type: none"> <li>• Verify the REQTOKEN address.</li> <li>• If you are calling IXLCACHE while disabled, REQTOKEN must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLCACHE in AR-mode and you specified the REQTOKEN address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are calling IXLCACHE in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLCACHE macro.</li> </ul>
8	xxxx083D	<p><b>Equate Symbol:</b> IXLRNCODEBADANSLEN</p> <p><b>Meaning:</b> Program error. The length of the answer area, as specified by ANSLEN, is not sufficient to contain answer area information. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Increase the size of the answer area provided for the request, and rerun your program. Depending on the macro version number, specify the length of the answer area as follows:</p> <ul style="list-style-type: none"> <li>• When the connection specified ASYNCXI=1 on the IXLCONN invocation and AXIOVERRIDE=0 was specified or defaulted to for the IXLCACHE request, the answer area length must be a minimum value of CAALEVEL2LEN</li> <li>• When the value of the macro version number (PLISTVER) is 4 or more, the minimum answer area length is CAALEVEL1LEN.</li> <li>• When the value of the macro version number (PLISTVER) is 0-3, the minimum answer area length is CAALEVEL0LEN.</li> </ul>
8	xxxx0851	<p><b>Equate Symbol:</b> IXLRNCODENOSUSPENDISABLE</p> <p><b>Meaning:</b> Program error. The request failed because MODE=SYNCSUSPEND was specified, but the caller is disabled and cannot be suspended.</p> <p><b>Action:</b> Either specify another MODE value or become enabled (release the CPU lock); then reissue the request.</p>
8	xxxx0865	<p><b>Equate Symbol:</b> IXLRNCODEBADBUFSPEC</p> <p><b>Meaning:</b> Program error. There is an error in the buffer specification.</p> <p><b>Action:</b> Check the following:</p> <ul style="list-style-type: none"> <li>• Requirements for BUFFER and BUFSIZE.</li> <li>• Buffer boundaries.</li> </ul>

Table 34. Return and Reason Codes for the IXLCACHE REQUEST=DELETE\_NAMELIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0866	<p><b>Equate Symbol:</b> IXLRNCODEBADBUFKEY</p> <p><b>Meaning:</b> Program error. The buffer storage key specified by BUFSTGKEY is incorrect or BUFSTGKEY was not specified and the PSW key does not match the key of the buffers.</p> <p>The data cannot be stored in the specified buffer area.</p> <p><b>Action:</b> Check the following:</p> <ul style="list-style-type: none"> <li>• Determine if the key of the storage being used for the buffers is different from the PSW key.</li> <li>• If BUFSTGKEY was specified, verify that the key was put in the correct bits (see the explanation of this parameter for more information).</li> <li>• If you are calling IXLCACHE in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLCACHE macro.</li> </ul>
8	xxxx0874	<p><b>Equate Symbol:</b> IXLRNCODEBADRNINDEX</p> <p><b>Meaning:</b> Program error. Either the value specified for either STARTINDEX or ENDINDEX is not valid or the size of the buffer specified by BUFFER is smaller than required based on the value of ENDINDEX.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The values specified by STARTINDEX and ENDINDEX are in the range of 1 to 32.</li> <li>• The value specified by ENDINDEX is greater than or equal to the value of STARTINDEX.</li> <li>• The value specified by ENDINDEX does not imply a larger BUFFER size than was actually specified on the REG_NAMELIST request.</li> </ul>
8	xxxx0875	<p><b>Equate Symbol:</b> IXLRNCODEBADNSBAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by NSBAREA is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The address passed in NSBAREA is valid.</li> <li>• The NSBAREA area was not previously freed.</li> <li>• The NSBAREA area is addressable from the caller's primary address space or from the caller's PASN access list.</li> <li>• If the caller is running in AR-mode and NSBAREA was specified using explicit register notation, ensure that the corresponding access register was updated appropriately.</li> <li>• If the caller is disabled, then NSBAREA must reside in either nonpageable or disabled reference storage.</li> <li>• If your program is running in AR mode, ensure that you specified SYSSTATE ASCENV=AR before issuing the IXLCACHE macro.</li> </ul>

Table 34. Return and Reason Codes for the IXLCACHE REQUEST=DELETE\_NAMELIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx08AD	<p><b>Equate Symbol:</b> IXLRSNCODEBADHIGHSHAREDVIRT</p> <p><b>Meaning:</b> Program error. The request specified a high shared virtual storage area (above 2GB).</p> <p><b>Action:</b> None required.</p>
C	xxxx0C06	<p><b>Equate Symbol:</b> IXLRSNCODENOCOONN</p> <p><b>Meaning:</b> Environmental error. No connectivity to the cache structure exists. This may occur due to operator commands such as VARY PATH,OFFLINE or CONFIG CHP,OFFLINE or hardware errors such as facility or path failures. The CONTOKEN is invalidated. The request fails.</p> <p><b>Action:</b> Either disconnect from the structure (using IXLDISC) or initiate a rebuild of the structure, if allowed (using IXLREBLD).</p>
8	xxxx08B9	<p><b>Equate Symbol:</b> IXLRSNCODENOANSAREA</p> <p><b>Meaning:</b> An answer area was not specified when one is required. The requested service determined that conditions exist that require an ANSAREA to complete the request.</p> <p><b>Action:</b> Provide an answer area (ANSAREA) and answer area length (ANSLEN) on the IXLCACHE macro invocation for the request. ANSAREA is required when the connection specified ASYNCXI=1 on the IXLCONN invocation when connecting to the cache structure, AXIOVERRIDE=0 was specified or defaulted to for the IXLCACHE request and the request is one of the following:</p> <ul style="list-style-type: none"> <li>• CROSS_INVALID</li> <li>• CROSS_INVALLIST</li> <li>• DELETE_NAME</li> <li>• DELETE_NAMELIST</li> <li>• READ_DATA</li> <li>• REG_NAMELIST</li> <li>• WRITE_DATA</li> <li>• WRITE_DATA LIST</li> </ul>

Table 34. Return and Reason Codes for the IXLCACHE REQUEST=DELETE\_NAMELIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C13	<p><b>Equate Symbol:</b> IXLRSNCODEREQPURGED</p> <p><b>Meaning:</b> Environmental error. The request was purged prior to completion of the request. Possible reasons include:</p> <ul style="list-style-type: none"> <li>• The connector failed.</li> <li>• The connector disconnected.</li> <li>• The requestor failed.</li> <li>• The request was purged by IXLPURGE.</li> <li>• Requests were purged when the connector provided an IXLEERSP response for the Rebuild Quiesce, Rebuild Stop, or Rebuild Cleanup event.</li> <li>• The connector invoked IXLREBLD REQUEST=COMPLETE.</li> <li>• The secondary address space was no longer valid.</li> </ul> <p><b>Action:</b> None if this is expected. Otherwise, determine why the connector failed.</p>
C	xxxx0C14	<p><b>Equate Symbol:</b> IXLRSNCODESTATUSUNKNOWN</p> <p><b>Meaning:</b> Environmental error. The IXLCACHE request has completed, but the final disposition of the request cannot be determined. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Verify the validity of your data by comparing the expected results with what is in the coupling facility.</p>
C	xxxx0C17	<p><b>Equate Symbol:</b> IXLRSNCODESTRFULL</p> <p><b>Meaning:</b> Environmental error. Allocation of a directory entry was necessary, but was unavailable or could not be reclaimed. In the answer area, CAASTGCLFULL contains the storage class from which the reclaiming operation failed. CAARNLINDEX contains the index of the registration block that was being processed when the error occurred. All prior registration blocks were processed.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Determine if any data items may be cast-out to make room for this item.</li> <li>• Check your usage of storage classes to see if some data items can be moved to a different storage class (preferably with a lower priority) so some entries in the structure can be freed.</li> <li>• Determine if a rebuild or an alter of the structure is necessary to make room for more data entries/items.</li> </ul> <p>After correcting the error, restart the REG_NAMELIST request starting with the registration block indexed by CAARNLINDEX.</p>



Table 34. Return and Reason Codes for the IXLCACHE REQUEST=DELETE\_NAMELIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C25	<b>Equate Symbol:</b> IXLRSNCODESTRFAILURE <b>Meaning:</b> Environmental error. The cache structure failed prior to completion of the request. <b>Action:</b> Attempt to rebuild the structure using IXLREBLD, or disconnect from the structure using IXLDISC.
C	xxxx0C68	<b>Equate Symbol:</b> IXLRSNCODEBADREQCFLEVEL <b>Meaning:</b> Environmental error. The request type is not permitted for the level of coupling facility in which the target structure is allocated. <b>Action:</b> Either continue processing using single READ_DATA requests to perform the function of the REG_NAMELIST request or disconnect from the structure (using IXLDISC) and request that the installation provide a coupling facility of the correct CFLEVEL (CFLEVEL=2 or higher).
C	xxxx0CA0	<b>Equate Symbol:</b> IXLRSNCODEQUIESCEDSUSPENDFAIL <b>Meaning:</b> Environmental error. The request is failed because the structure is quiesced for a system-managed process and SUSPEND=FAIL is specified on the IXLCONN. <b>Action:</b> None, if this is expected.
C	xxxxFFFF	<b>Equate Symbol:</b> IXLRSNCODENOTAVAILABLE <b>Meaning:</b> Environmental error. XES functions are not available. The coupling facility hardware might not be present. <b>Action:</b> XES may not be used when XCF is running local mode. XES requires that the coupling facility be installed.
10	xxxx10xx	<b>Equate Symbol:</b> IXLRSNCODECOMPERROR <b>Meaning:</b> System error. XES processing failure. The state of the structure and the disposition of the request are unpredictable. The answer area (ANSAREA) fields do not contain valid information. <b>Action:</b> Save the reason code information, and contact the IBM support center.



---

## Chapter 37. IXLCACHE REQUEST=PROCESS\_REFLIST

---

### Description

A PROCESS\_REFLIST request allows you to indicate that a set of data items has been recently referenced, even in the absence of actual read or write activity against those data items. This causes the storage class queue to be re-ordered and the data items in the list to be considered recently referenced. The least recently referenced data items are the first to be considered for a reclaim. See *z/OS MVS Programming: Sysplex Services Guide* for more information on reclamation and referenced versus unreferenced.

The list of data items to be processed should be supplied in the storage area specified by BUFLIST or BUFFER. You must also specify the number of names (NUMNAMES) contained in the buffer(s). The data items are processed only if the connection has registered interest in the data items and the data items are associated with the specified storage class (STGCLASS).

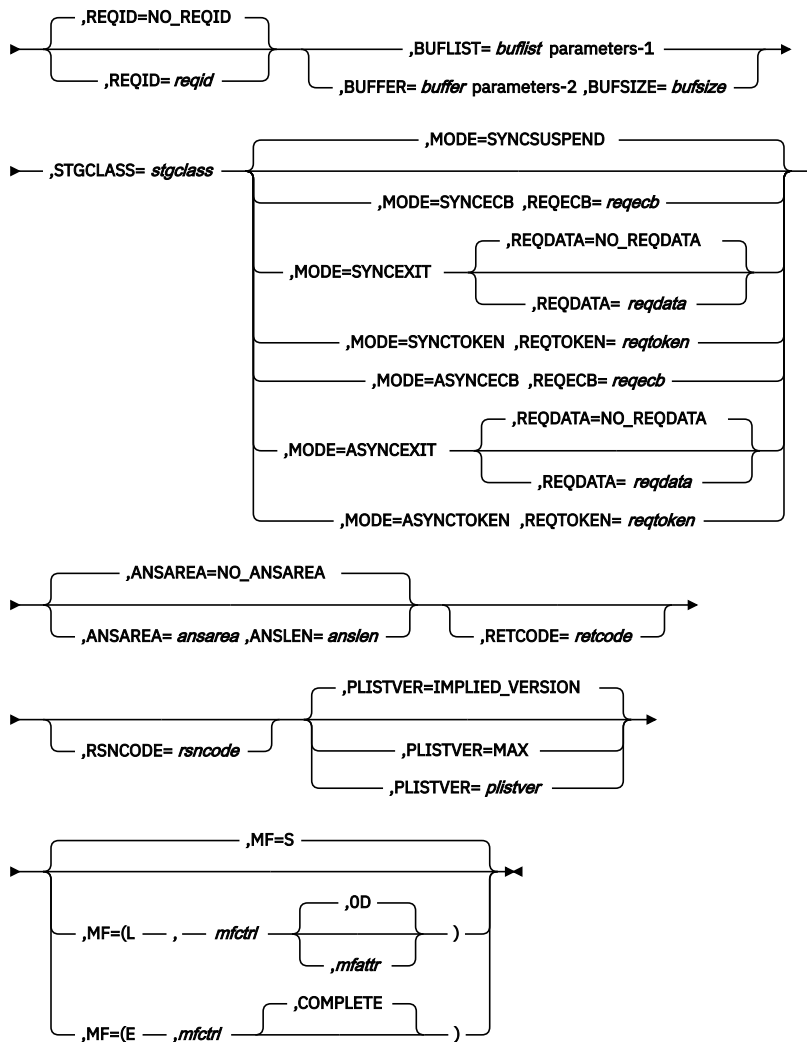
---

### Syntax Diagram

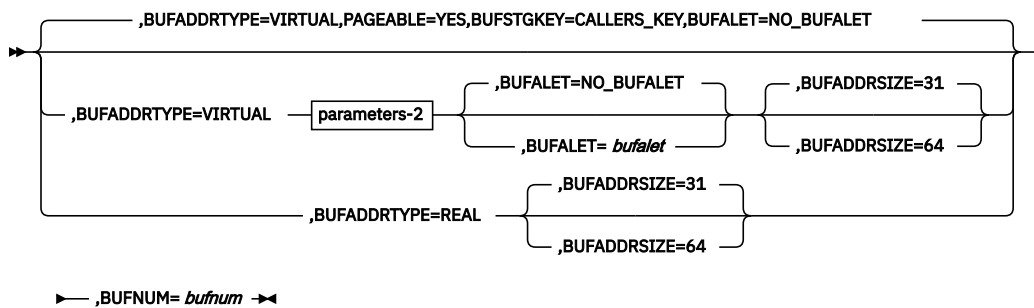
The syntax of the IXLCACHE REQUEST=PROCESS\_REFLIST macro is as follows:

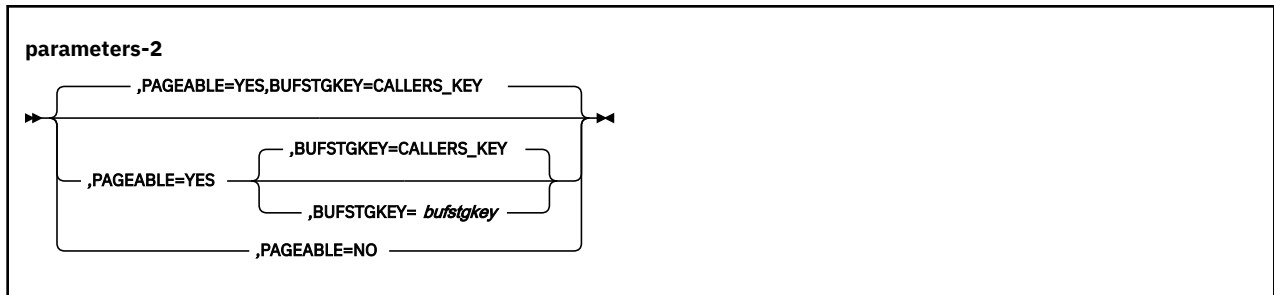
## main diagram

► IXLCACHE — REQUEST=PROCESS\_REFLIST — ,NUMNAMES= *numnames* — ,CONTOKEN= *contoken* →



## parameters-1





**Note:** When MODE=SYNCTOKEN or MODE=ASYNCTOKEN is specified, ANSAREA= *ansarea* and ANSLen=*anslen* are required.

## Parameter Descriptions

The parameter descriptions for REQUEST=PROCESS\_REFLIST are listed in alphabetical order. Default values are underlined:

### REQUEST=PROCESS\_REFLIST

Use this input parameter to specify that the data items listed by name in the storage areas specified by BUFLIST or BUFFER be processed as having been recently referenced.

#### **,ANSAREA=NO** ANSAREA

#### **,ANSAREA=ansarea**

Use this output parameter to specify an answer area to contain information returned from the request. The format of the answer area is described by the IXLYCAA mapping macro.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an area (with a length of ANSLen) where the information returned from the request will be put.

#### **,ANSLen=anslen**

Use this inppaameter to specify the size of the storage area specified by ANSAREA.

Use either CAALEVELOLEN, CAALEVEL1LEN or CAALEVEL2LEN of the IXLYCAA mapping macro to determine the minimum size of the answer area. The answer area length must be at least large enough to accommodate the level of the IXLYCAA mapping appropriate to the requested function.

- When the connection specified ASYNCXI=1 on the IXLCONN invocation and AXIOVERRIDE=0 was specified or defaulted to for the IXLCACHE request, the answer area length is a required parameter and must be a minimum value of CAALEVEL2LEN to contain a returned asynchronous cross-invalidation sequence number (CAAASYNXISEQNUM).
- When the value of PLISTVER is 4 or above, the minimum answer area length is CAALEVEL1LEN.
- When the value of PLISTVER is 0 - 3, the minimum answer area length is CAALEVELOLEN.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12 2 to ) of a 2-byte field that contains the length of the answer area (ANSAREA).

#### **,BUFADDRSIZE=31**

#### **,BUFADDRSIZE=64**

Use this input parameter to specify whether a 31-bit or a 64-bit address is specified by a BUFLIST entry.

#### **31**

The entry in BUFLIST is 31 bits in size.

#### **64**

The entry in BUFLIST is 64 bits in size.

#### **,BUFADDRTYPE=VIRTUAL**

#### **,BUFADDRTYPE=REAL**

Use this input parameter to specify whether the buffer addresses specified in the BUFLIST list are virtual storage or real storage addresses.

**VIRTUAL**

The buffer addresses are virtual storage addresses. The virtual storage can be pageable or nonpageable. See the PAGEABLE parameter for information about managing storage binds when specifying virtual storage addresses.

**REAL**

The buffer addresses are real storage addresses.

It is the caller's responsibility to manage the binds between the data buffer virtual storage and the real storage addresses provided. The caller must ensure that the data buffer virtual storage remains bound to the real storage addresses provided until the request completes.

**,BUFALET=NO\_BUFALET****,BUFALET=*bufalet***

Use this input parameter to specify an access list entry token (ALET) to be used in referencing all of the buffers specified by BUFLIST.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 4-byte field that contains the ALET.

**,BUFFER=*buffer***

Use this input parameter to specify a buffer area to contain a list of entry names to be processed.

You must ensure that the storage area specified by BUFFER:

- Is a multiple of 4096 bytes.
- Is less than or equal to 65536 bytes.
- Starts on a 4096-byte boundary.
- Does not start below storage address 512.

The buffers must be formatted to contain 16-byte elements, starting at offset zero. Each 16-byte element must contain a name to be processed.

See the NUMNAMES description to specify the number of data items to be processed, and the BUFSIZE description to specify the size of the buffer.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an area (with a length of BUFSIZE) that contains the list of names to be processed.

**,BUFLIST=*buflist***

Use this input parameter to specify a list of buffers to hold a list of names to be processed. BUFLIST specifies a 128-byte storage area that consists of a list of 0 to 16 buffer addresses.

The format of the list is a set of 8-byte elements. The BUFADDRSIZE keyword denotes whether four or eight bytes of the element are used.

- If BUFADDRSIZE=31 is specified, then the first four bytes of each element are reserved space and the last four bytes contain the address of the buffer.
- If BUFADDRSIZE=64 is specified, then the full eight bytes specify the address of the buffer.

**The BUFLIST buffers must:**

- Reside in the same address space or same data space.
- Be 4096 bytes.
- Start on a 4096-byte boundary.
- Not start below storage address 512.
- Consist of 16-byte elements, starting at offset zero. Each 16-byte element should contain a name to be processed. The named element must reside in the storage class specified by STGCLASS.

**Note:** The buffers do not have to be contiguous in storage. Cache services treat BUFLIST buffers as a single buffer, even if the buffers are not contiguous.

See the NUMNAMES description to specify the number of data items to be processed, and the BUFNUM parameter to specify the number of buffers in the buffer list.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 128-byte field that contains a list of buffer addresses.

**,BUFNUM=*bufnum***

Use this input parameter to specify the number of buffers in the BUFLIST list. Valid BUFNUM values are from 0 to 16. A value of zero indicates that no entries will be processed.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 1-byte field that contains the number of buffers in the buffer list.

**,BUFSIZE=*bufsize***

Use this input parameter to specify the size of the BUFFER area. See the BUFFER parameter description for valid buffer sizes.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a fullword field that contains the size of the buffer (BUFFER) in bytes.

**,BUFSTGKEY=CALLERS\_KEY**

**,BUFSTGKEY=*bufstgkey***

Use this input parameter to specify a storage key that you define and use when referencing the buffers specified by BUFLIST or the buffer that is specified by BUFFER.

If you do not specify BUFSTGKEY, or if you specify BUFSTGKEY=CALLERS\_KEY, all references to one or more buffers are performed by using the caller's PSW key.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of an 8-bit field that contains the storage key in the format B'kkkkxxxx', where kkkk is the key and xxxx is ignored.

**,CONTO=*conken***

Use this input parameter to specify the connect token that was returned by the IXLCONN service in the IXLCONN answer area, which is mapped which is by IXLYCONA. The connect token uniquely identifies your connection to the cache structure, and must be specified on each IXLCACHE invocation.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field that contains the connect token.

**,MF=S**

**,MF=(L,*mfctrl*)**

**,MF=(L,*mfctrl*,*mfattr*)**

**,MF=(L,*mfctrl*,0D)**

**,MF=(M,*mfctrl*)**

**,MF=(M,*mfctrl*,COMPLETE)**

**,MF=(M,*mfctrl*,NOCHECK)**

**,MF=(E,*mfctrl*)**

**,MF=(E,*mfctrl*,COMPLETE)**

**,MF=(E,*mfctrl*,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

**,*mfctrl***

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

**,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

**,COMPLETE**

**,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

**COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if *SMILE=var* were an optional parameter and the default is *SMILE=NO\_SMILE* then it would not be documented. However, if the default was *SMILE=-*), then it would be documented because a value would be the default.

**NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

**,MODE=SYNCSUSPEND**

**,MODE=SYNCECB**

**,MODE=SYNCEXIT**

**,MODE=SYNCTOKEN**

**,MODE=ASYNCECB**

**,MODE=ASYNCEXIT**

**,MODE=ASYNCTOKEN**

Use this input parameter to specify:

- Whether the request is to be performed synchronously or asynchronously
- How you want to be notified of request completion if the request is processed asynchronously.

See *z/OS MVS Programming: Sysplex Services Guide* for more information on understanding synchronous and asynchronous cache operations.

**SYNCSUSPEND**

The request processes synchronously. If necessary, the request is suspended until it can complete synchronously. To use this option, your program must be enabled for I/O and external interrupts.

**SYNCECB**

The request processes synchronously if possible. If the request processes asynchronously, the ECB specified by REQECB is posted when the request completes.

**SYNCEXIT**

The request processes synchronously if possible. If the request processes asynchronously, your complete exit is given control when the request completes.

**SYNCTOKEN**

The request processes synchronously if possible. If the request processes asynchronously, an asynchronous request token is returned to the area specified by REQTOKEN. Use the returned request token on the IXLFCOMP macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=SYNCTOKEN is specified.

**ASYNCECB**

The request processes asynchronously. The ECB specified by REQECB is posted when the request completes.



**ASYNCEXIT**

The request processes asynchronously. Your complete exit is given control when the request completes.

**ASYNCTOKEN**

The request processes asynchronously. An asynchronous request token is returned to the area specified by REQTOKEN. Use the returned request token on the IXLFCOMP macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=ASYNCTOKEN is specified.

**,NUMNAMES=numnames**

Use this input parameter to specify the number of names contained in the BUFFER area or BUFLIST buffers.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 2-byte field that contains the number of names.

**,PAGEABLE=YES****,PAGEABLE=NO**

Use this input parameter to identify whether the storage areas specified by BUFFER or BUFLIST are in pageable or potentially pageable storage.

**YES**

Specify this option to indicate that the BUFFER or BUFLIST buffers reside in pageable virtual storage. XES performs the required page fixing to fix the buffers in real storage while the cache or list request transfers data to or from the coupling facility.

This includes storage obtained from pageable subpools, disabled reference (DREF) storage, and may include storage that has the potential to become pageable during the processing of a request. (An example is address space storage owned by any swappable address space, for which a PGSER FIX has been successfully processed, but for which the owning address space gets swapped during processing of a cache or list request.) This does not include implicitly non-pageable storage (for example, storage obtained from non-pageable subpools).

The system takes responsibility for managing binds to central storage for the duration of the cache or list request, regardless of what address space owns the storage or whether the storage-owning address space is swappable or nonswappable. The storage can be owned by any address space.

High shared virtual storage areas (above 2GB) may not be used.

**NO**

Specify this option to indicate that the BUFFER or BUFLIST buffers reside in non-pageable virtual storage. XES does not page fix the buffers in real storage.

This includes implicitly non-pageable storage areas (for example, storage obtained from non-pageable subpools), and may include storage that has the potential to become pageable during the processing of a request (An example is address space storage owned by any swappable address space, for which a PGSER FIX has been successfully processed, but for which the owning address space gets swapped-out during processing of a cache or list request.)

The system takes responsibility for managing binds to central storage for the duration of the cache or list request, if and only if the non-pageable storage is owned by either the requester's address space or the connector's address space. If the storage is owned by any other address space, then the invoker is responsible for ensuring that the virtual storage remains non-pageable for the duration of the request (including the case in which the storage is owned by a swappable address space that is swapped during processing of an IXLCACHE or IXLLIST request). Subject to this consideration, the storage can be owned by any address space. See [z/OS MVS Programming: Sysplex Services Guide](#).

**,PLISTVER=IMPLIED\_VERSION****,PLISTVER=MAX****,PLISTVER=plistver**

Use this input parameter to specify the version of the macro. See “[Understanding IXLCACHE Version Support](#)” on page 461 for a description of the options available with PLISTVER.

**,REQDATA=NO\_REQDATA****,REQDATA=reqdata**

Use this input parameter with MODE=SYNCEXIT or MODE=ASYNCEXIT to pass any data you choose to the complete exit. The exit will get control only if the request is processed asynchronously.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the data to be passed to the complete exit.

**,REQECB=reqecb**

Use this output parameter with either MODE=SYNCECB or MODE=ASYNCECB to specify the address of an ECB, which is to be posted when the request completes if the request was processed asynchronously.

Before coding REQECB, you must ensure that:

- You initialize the ECB before you issue the request.
- The ECB resides in either common storage or the home address space where IXLCONN was issued.
- Any tasks that wait for the ECB to be posted reside in the home address space where IXLCONN was issued.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that contains the address of the ECB to be posted when the request completes. The ECB must be aligned on a fullword boundary.

**,REQID=NO\_REQID****,REQID=reqid**

Use this input parameter to specify a user-defined request identifier to be associated with the request. You can specify this request identifier on the IXPURGE macro to cancel a request that has not yet been processed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the user-defined request identifier.

**,REQTOKEN=reqtoken**

Use this output parameter with either MODE=SYNCTOKEN or MODE=ASYNCTOKEN to specify the address of a storage area to receive the request token that is returned when the request will be processed asynchronously. This token, which uniquely identifies the request, must be used as input to the IXLFCOMP macro, which you use to determine if the request has completed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field where the system will put the request token.

**,RETCODE=retcode**

Use this output parameter to specify a field to contain the return code. (The return code is also returned in register 15.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the return code when the request has completed.

**,RSNCODE=rsncode**

Use this output parameter to specify a field to contain the reason code returned, if applicable. (The reason code is also returned in register 0.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the reason code (if any) when the request has completed.

**,STGCLASS=stgclass**

Use this input parameter to specify the storage class to which the data items to be processed must be assigned.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 1-byte field that contains the storage class.

## ABEND Codes

Abend X'026' (See *z/OS MVS System Codes* for more information on this abend.)

## Return and Reason Codes

When the system returns control to your program:

- GPR 15 (and RETCODE, if specified) contains a return code.
- If the return code is not zero, GPR 0 (and RSNCODE, if specified) contains a reason code.

**Note:** The return and reason codes will also be put into the answer area (mapped by IXLYCAA), if there is one, when the request completes.

The IXLYCON macro provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

<b>0</b>	IXLRETCODEOK
<b>4</b>	IXLRETCODEWARNING
<b>8</b>	IXLRETCODEPARMERROR
<b>C</b>	IXLRETCODEENVERROR
<b>10</b>	IXLRETCODECOMPERROR

The following table contains hexadecimal return and reason codes, the equate symbols associated with each reason code, and the meaning and suggested action for each return and reason code.

Table 35. Return and Reason Codes for the IXLCACHE REQUEST=PROCESS_REFLIST Macro		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0	None.	<p><b>Meaning:</b> If you specified a MODE value of ASYNCECB, ASYNCEXIT, or ASYNCTOKEN, the request has been successfully initiated. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed. For any other MODE value, the request has successfully completed.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=ASYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=ASYNCEXIT, the connection's complete exit will be given control when the request has completed.</li> <li>• If you specified MODE=ASYNCTOKEN, use REQTOKEN when calling IXLFCOMP to determine when the request has completed.</li> </ul>

Table 35. Return and Reason Codes for the IXLCACHE REQUEST=PROCESS\_REFLIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	xxxx0402	<p><b>Equate Symbol:</b> IXLRNSNCODEASYNCH</p> <p><b>Meaning:</b> The request could not be processed synchronously. It will be processed asynchronously. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed.</p> <ul style="list-style-type: none"> <li>• If you specified MODE=SYNCECB, the ECB specified in REQECB will be posted when the request has finished.</li> <li>• If you specified MODE=SYNCEXIT, the connection's complete exit will be given control when the request has finished.</li> <li>• If you specified MODE=SYNCTOKEN, the REQTOKEN contains an asynchronous request token that may be used on the IXLFCOMP macro to determine when the request has completed.</li> </ul> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=SYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=SYNCEXIT, the connection's completion exit will be given control when the request has completed.</li> <li>• If you specified MODE=SYNCTOKEN, use REQTOKEN when calling IXLFCOMP to determine when the request has completed.</li> </ul>
8	xxxx0801	<p><b>Equate Symbol:</b> IXLRNSNCODEBADPARMLIST</p> <p><b>Meaning:</b> The parameter list (mfctrl) for this request is not addressable. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the parameter list address.</li> <li>• The parameter list must be addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are issuing this macro while disabled, the parameter list must reside in either page-fixed or DREF storage.</li> <li>• If you are issuing this macro in AR-mode and you specified the parameter list address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are issuing this macro in AR-mode, SYSSTATE ASCENV=AR must be issued before this macro.</li> </ul>

Table 35. Return and Reason Codes for the IXLCACHE REQUEST=PROCESS\_REFLIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0804	<p><b>Equate Symbol:</b> IXLRSNCODEBADVERSION#</p> <p><b>Meaning:</b> The version number in the macro parameter list is not compatible with the level of XES currently being used. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that your program did not overlay the parameter list storage.</li> <li>• Verify that your program was assembled with the correct macro library for the release of MVS on which your program is running.</li> <li>• Verify that your program is running on an MVS system that supports the version of the macro you are using.</li> </ul>

Table 35. Return and Reason Codes for the IXLCACHE REQUEST=PROCESS\_REFLIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx080A	<p><b>Equate Symbol:</b> IXLRNSCODEBADCONTOKEN</p> <p><b>Meaning:</b> Program error. The specified connect token (CONTOKEN) was not valid for one of the following reasons:</p> <ol style="list-style-type: none"> <li>1. The user with the connection identifier represented by the token has disconnected from the structure.</li> <li>2. The connector's task (the task that issued IXLCONN) ended.</li> <li>3. The specified token is not the token that was returned from IXLCONN.</li> <li>4. The request was issued from an address space other than the address space in which IXLCONN was issued.</li> <li>5. The connect token was invalidated during rebuild.</li> <li>6. The connect token was invalidated by XES.</li> </ol> <p><b>Note:</b> The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Verify that the CONTOKEN value specified was valid and for the correct structure. The numbers on the following actions correspond to the numbers above.</p> <ol style="list-style-type: none"> <li>1. Discontinue use. Perform recovery and cleanup for the structure and check protocol for use of structure to determine why you did not know of the disconnection.</li> <li>2. Discontinue use. Perform recovery and cleanup for the structure and check protocol for use of structure to determine why you did not know that the task ended.</li> <li>3. Issue IXLCONN to connect to the structure. The contoken is returned in the CONACONTOKEN field of the answer area specified on the IXLCONN request.</li> <li>4. Issue your request from the same address space the IXLCONN was issued.</li> <li>5. Participate in the rebuild. When it is complete, try again.</li> <li>6. Discontinue use. Perform recovery and cleanup for the structure and check protocol for use of structure.</li> </ol> <p>You may want to issue IXCQUERY to get more information about the structure.</p>
8	xxxx0824	<p><b>Equate Symbol:</b> IXLRNSCODEWRONGSTRTYPE</p> <p><b>Meaning:</b> The connect token specified by CONTOKEN is not to a cache structure.</p> <p><b>Action:</b> Verify the connect token for this cache structure.</p> <p><b>Note:</b> The connect token can be found in the IXLCONN answer area. Verify the value specified on the TYPE= parameter of the IXLCONN request for this structure.</p>

Table 35. Return and Reason Codes for the IXLCACHE REQUEST=PROCESS\_REFLIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx082D	<p><b>Equate Symbol:</b> IXLRNCODEBADSTGCLASS</p> <p><b>Meaning:</b> Program error. The storage class specified by STGCLASS exceeds the maximum number of storage classes defined for the cache structure.</p> <p><b>Action:</b> The number of storage classes allowed for a structure are defined on the IXLCONN macro by the NUMSTGCLASS parameter. Correct the STGCLASS parameter to specify a valid storage class.</p>
8	xxxx0830	<p><b>Equate Symbol:</b> IXLRNCODEBADNUMNAMES</p> <p><b>Meaning:</b> The NUMNAMES specification is not valid.</p> <p><b>Action:</b> Ensure that the number of names specified by NUMNAMES reflects the correct number of names stored in BUFLIST or BUFFER.</p>
8	xxxx0833	<p><b>Equate Symbol:</b> IXLRNCODEBADPGBLATTR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list is specified as being pageable (PAGEABLE=YES) but is not.</p> <p><b>Action:</b> Change the buffer area(s) to pageable storage, or specify PAGEABLE=NO. See the PAGEABLE parameter description for specification instructions.</p>
8	xxxx0834	<p><b>Equate Symbol:</b> IXLRNCODEBADNONPGBLATTR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list is specified as being nonpageable (PAGEABLE=NO) but is either pageable or not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The type of storage in the buffer area corresponds to the value you specified on the PAGEABLE parameter. See the PAGEABLE parameter description for specification instructions.</li> <li>• The correct buffer address was used.</li> <li>• The buffer area(s) were not previously freed.</li> <li>• If BUFLIST was specified and your program is running in AR mode, ensure that: <ul style="list-style-type: none"> <li>– The BUFALET specification is correct.</li> <li>– You specified SYSSTATE ASCENV=AR before issuing the macro.</li> </ul> </li> <li>• If the caller is running in AR-mode and the BUFFER or BUFLIST parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> </ul>

Table 35. Return and Reason Codes for the IXLCACHE REQUEST=PROCESS\_REFLIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0835	<p><b>Equate Symbol:</b> IXLRNCODEBADDATAADDR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list, is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The correct buffer address was used for BUFFER or for a buffer within the BUFLIST.</li> <li>• The buffer area(s) were not previously freed.</li> <li>• The buffer area(s) were allocated in a storage key that matches the key specified using the BUFSTGKEY parameter, or, if BUFSTGKEY is omitted, the caller's PSW key.</li> <li>• If the caller is running in AR-mode and the BUFFER or BUFLIST parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If the caller is running in AR-mode, SYSSTATE ASCENV=AR must be specified before issuing this macro.</li> <li>• If BUFLIST was specified and your program is running in AR mode the BUFALET specification is correct.</li> </ul>
8	xxxx0836	<p><b>Equate Symbol:</b> IXLRNCODEBADREALADDR</p> <p><b>Meaning:</b> Program error. Real storage addresses were provided in a BUFLIST list, but one of the buffers is not addressable in central storage.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that BUFADDRTYPE was specified as you intended.</li> <li>• Ensure that the real buffer addresses specified by BUFLIST are valid.</li> </ul>
8	xxxx0838	<p><b>Equate Symbol:</b> IXLRNCODEBADANSAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by ANSAREA is not addressable.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the ANSAREA address.</li> <li>• ANSAREA must be addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are calling IXLCACHE while disabled, ANSAREA must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLCACHE in AR-mode and you specified the ANSAREA address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are calling IXLCACHE in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLCACHE macro.</li> </ul>



Table 35. Return and Reason Codes for the IXLCACHE REQUEST=PROCESS\_REFLIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0839	<p><b>Equate Symbol:</b> IXLRNCODEBADREQTOKENAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by REQTOKEN is not addressable.</p> <p><b>Action:</b> Ensure that the request token area specified by REQTOKEN is valid:</p> <ul style="list-style-type: none"> <li>• Verify the REQTOKEN address.</li> <li>• If you are calling IXLCACHE while disabled, REQTOKEN must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLCACHE in AR-mode and you specified the REQTOKEN address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are calling IXLCACHE in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLCACHE macro.</li> </ul>
8	xxxx083D	<p><b>Equate Symbol:</b> IXLRNCODEBADANSLEN</p> <p><b>Meaning:</b> Program error. The length of the answer area, as specified by ANSLEN, is not sufficient to contain answer area information. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Increase the size of the answer area provided for the request, and rerun your program. Depending on the macro version number, specify the length of the answer area as follows:</p> <ul style="list-style-type: none"> <li>• When the connection specified ASYNCXI=1 on the IXLCONN invocation and AXIOVERRIDE=0 was specified or defaulted to for the IXLCACHE request, the answer area length must be a minimum value of CAALEVEL2LEN</li> <li>• When the value of the macro version number (PLISTVER) is 4 or more, the minimum answer area length is CAALEVEL1LEN.</li> <li>• When the value of the macro version number (PLISTVER) is 0-3, the minimum answer area length is CAALEVEL0LEN.</li> </ul>
8	xxxx0851	<p><b>Equate Symbol:</b> IXLRNCODENOSUSPENDISABLE</p> <p><b>Meaning:</b> Program error. The request failed because MODE=SYNCSUSPEND was specified, but the caller is disabled and cannot be suspended.</p> <p><b>Action:</b> Either specify another MODE value or become enabled (release the CPU lock); then reissue the request.</p>

Table 35. Return and Reason Codes for the IXLCACHE REQUEST=PROCESS\_REFLIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0865	<p><b>Equate Symbol:</b> IXLRNCODEBADBUFSPEC</p> <p><b>Meaning:</b> Program error. There is an error in the buffer specification.</p> <p><b>Action:</b> Check the following:</p> <ul style="list-style-type: none"> <li>• If BUFLIST was specified, check the requirements for BUFLIST and BUFNUM.</li> <li>• If BUFFER was specified, check the requirements for BUFFER and BUFSIZE.</li> <li>• Buffer pointer(s) in BUFLIST.</li> <li>• Buffer boundaries.</li> </ul>
8	xxxx0866	<p><b>Equate Symbol:</b> IXLRNCODEBADBUFKEY</p> <p><b>Meaning:</b> Program error. The buffer storage key specified by BUFSTGKEY is incorrect or BUFSTGKEY was not specified and the caller's PSW key does not match the key of the buffers.</p> <p>The data cannot be fetched from the specified buffer area.</p> <p><b>Action:</b> Check the following:</p> <ul style="list-style-type: none"> <li>• Determine if the key of the storage being used for the buffers is different from the PSW key.</li> <li>• If BUFSTGKEY was specified, verify that the key was put in the correct bits (see the explanation of this parameter for more information).</li> </ul>
8	xxxx0867	<p><b>Equate Symbol:</b> IXLRNCODEBADBUFLIST</p> <p><b>Meaning:</b> Program error. The 128-byte storage area specified by BUFLIST is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The correct BUFLIST address was used.</li> <li>• The BUFLIST area was not previously freed.</li> <li>• If the caller is running in AR-mode and the BUFLIST parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If the caller is disabled, then the BUFLIST must reside in either nonpageable or disabled reference storage.</li> <li>• If your program is running in AR mode, ensure that you specified SYSSTATE ASCENV=AR before issuing the macro.</li> </ul>

Table 35. Return and Reason Codes for the IXLCACHE REQUEST=PROCESS\_REFLIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C06	<p><b>Equate Symbol:</b> IXLRNOCODENOCOONN</p> <p><b>Meaning:</b> Environmental error. No connectivity to the cache structure exists. This may occur due to operator commands such as VARY PATH,OFFLINE or CONFIG CHP,OFFLINE or hardware errors such as facility or path failures. The CONTOKEN is invalidated. The request fails.</p> <p><b>Action:</b> Either disconnect from the structure (using IXLDISC) or initiate a rebuild of the structure, if allowed (using IXLREBLD).</p>
C	xxxx0C13	<p><b>Equate Symbol:</b> IXLRNOCODEREQPURGED</p> <p><b>Meaning:</b> Environmental error. The request was purged prior to completion of the request. Possible reasons include:</p> <ul style="list-style-type: none"> <li>• The connector failed.</li> <li>• The connector disconnected.</li> <li>• The requestor failed.</li> <li>• The request was purged by IXLPURGE.</li> <li>• Requests were purged when the connector provided an IXLEERSP response for the Rebuild Quiesce, Rebuild Stop, or Rebuild Cleanup event.</li> <li>• The connector invoked IXLREBLD REQUEST=COMPLETE.</li> <li>• The secondary address space was no longer valid.</li> </ul> <p><b>Action:</b> None if this is expected. Otherwise, determine why the connector failed.</p>
C	xxxx0C14	<p><b>Equate Symbol:</b> IXLRNOCODESTATUSUNKNOWN</p> <p><b>Meaning:</b> Environmental error. The IXLCACHE request has completed, but the final disposition of the request cannot be determined. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Verify the validity of your data by comparing the expected results with what is in the coupling facility.</p>
C	xxxx0C25	<p><b>Equate Symbol:</b> IXLRNOCODESTRFAILURE</p> <p><b>Meaning:</b> Environmental error. The cache structure failed prior to completion of the request.</p> <p><b>Action:</b> Attempt to rebuild the structure using IXLREBLD, or disconnect from the structure using IXLDISC.</p>
C	xxxx0CA0	<p><b>Equate Symbol:</b> IXLRNOCODEQUIESCEDSUSPENDFAIL</p> <p><b>Meaning:</b> Environmental error. The request is failed because the structure is quiesced for a system-managed process and SUSPEND=FAIL is specified on the IXLCONN.</p> <p><b>Action:</b> None, if this is expected.</p>

Table 35. Return and Reason Codes for the IXLCACHE REQUEST=PROCESS\_REFLIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxxFFFF	<p><b>Equate Symbol:</b> IXLRSNCODENOTAVAILABLE</p> <p><b>Meaning:</b> Environmental error. XES functions are not available. The coupling facility hardware might not be present.</p> <p><b>Action:</b> XES may not be used when XCF is running local mode. XES requires that the coupling facility be installed.</p>
10	xxxx10xx	<p><b>Equate Symbol:</b> IXLRSNCODECOMPERROR</p> <p><b>Meaning:</b> System error. XES processing failure. The state of the structure and the disposition of the request are unpredictable. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Save the reason code information, and contact the IBM support center.</p>

---

## Chapter 38. IXLCACHE REQUEST=READ\_COCLASS

---

### Description

A READ\_COCLASS request allows you to read directory entry names and directory entry user data entries associated with a specified cast-out class (COCLASS). You may filter the cast-out class information by specifying the NAME and NAMEMASK parameters. You may limit the amount of information returned with the DIRINFOFMT parameter. The retrieved information is stored in the storage areas specified by BUFLIST or BUFFER. For structures allocated in a coupling facility of CFLEVEL=4 or lower, the format of the returned information is described by mapping macro IXLYCANB.

For structures allocated in a coupling facility of CFLEVEL=5 or higher, the format of the returned information is described by either mapping macro IXLYCANB or mapping macro IXLYDEIB, depending on the value specified for the DIRINFOFMT keyword.

The number of directory entry name elements (names and user data) that are retrieved is provided in the answer area (ANSAREA). The answer area is described by the IXLYCAA mapping macro.

If the request completes prematurely because it exceeds the model-dependent time-out criteria or the specified buffer area (BUFLIST or BUFFER) is full, a restart token (RESTOKEN) or an extended restart token (EXTRESTOKEN) is returned in the answer area (fields CAARESTOKEN or CAAEXTRESTOKEN). The token can be specified on the next READ\_COCLASS request to resume processing with the next directory entry to be processed. Resumed requests are processed identically whether using the RESTOKEN or EXTRESTOKEN to specify the starting location.

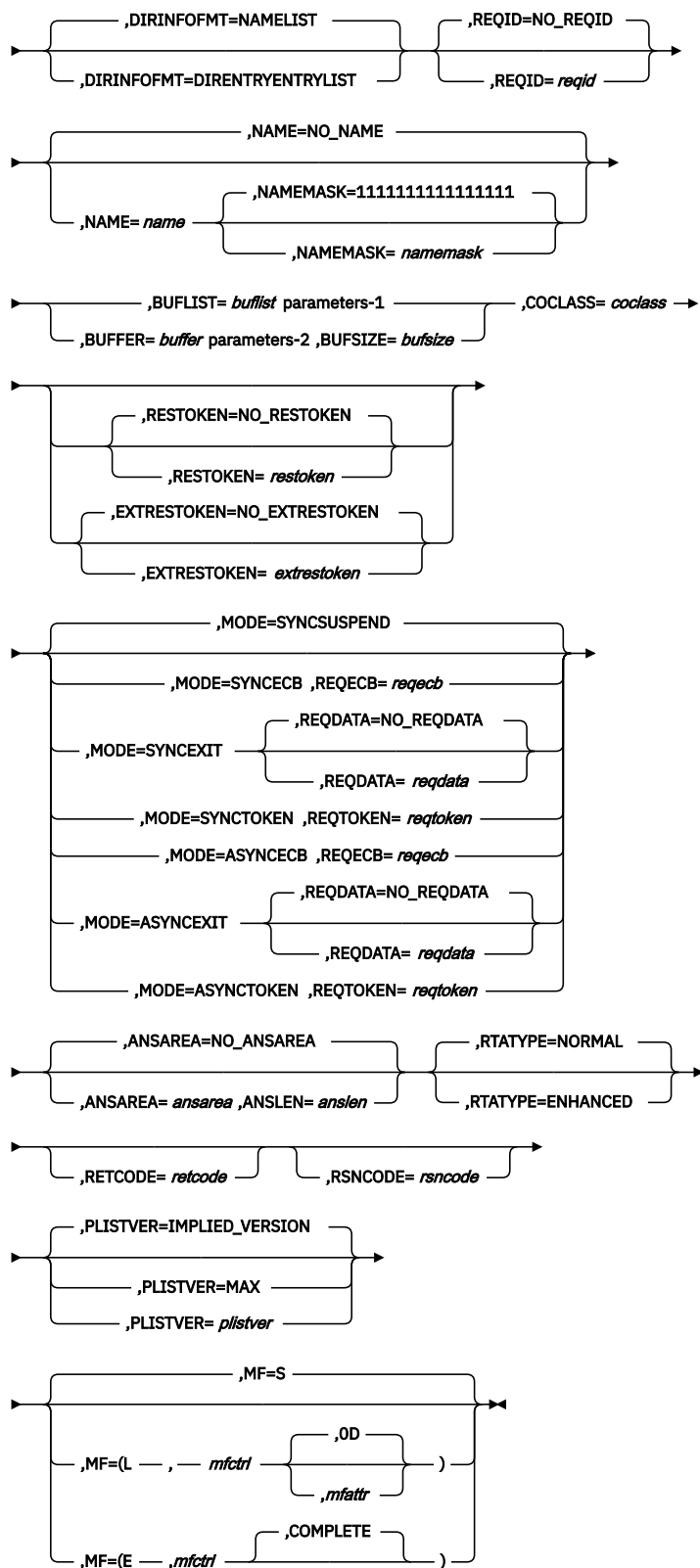
---

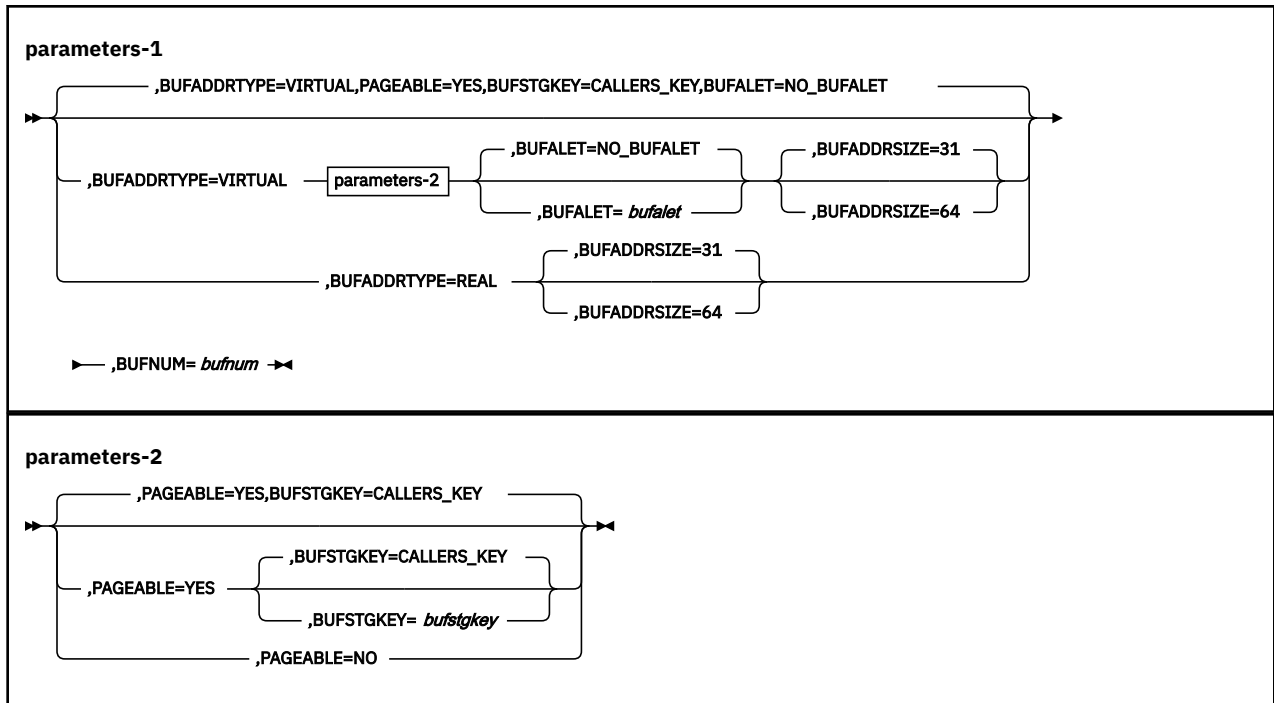
### Syntax Diagram

The syntax diagram for IXLCACHE REQUEST=READ\_COCLASS is as follows:

## main diagram

➔ IXLCACHE — REQUEST=READ\_COCLASS — ,CONTOKEN= *contoken* ➔





**Note:** When MODE=SYNCTOKEN or MODE=ASYNCTOKEN is specified, ANSAREA=*ansarea*,ANSLEN=*anslen* is required.

## Parameter Descriptions

The parameter descriptions for REQUEST=READ\_COCLASS are listed in alphabetical order. Default values are underlined:

### REQUEST=READ\_COCLASS

Use this input parameter to specify that the directory entry names and information for a specified cast-out class (COCLASS) be stored in the areas specified by BUFFER or BUFLIST. You may filter the amount of information returned by the request by using the NAME and NAMEMASK parameters.

### ,ANSAREA=NO\_ANSAREA

### ,ANSAREA=*ansarea*

Use this output parameter to specify an answer area to contain information returned from the request. The information can include a restart token or extended restart token when the request exceeds the model-dependent time-out criteria or the specified buffer area is full. The format of the answer area is described by the IXLYCAA mapping macro.

Upon successful completion of the request the answer area contains the number of directory entries processed by the request (field CAADIRCOUNT).

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an area (with a length of ANSLEN) where the information returned from the request will be put.

### ,ANSLEN=*anslen*

Use this inppaameter to specify the size of the storage area specified by ANSAREA.

Use either CAALEVELOLEN, CAALEVEL1LEN or CAALEVEL2LEN of the IXLYCAA mapping macro to determine the minimum size of the answer area. The answer area length must be at least large enough to accommodate the level of the IXLYCAA mapping appropriate to the requested function.

- When the connection specified ASYNCXI=1 on the IXLCONN invocation and AXIOVERRIDE=0 was specified or defaulted to for the IXLCACHE request, the answer area length is a required parameter and must be a minimum value of CAALEVEL2LEN to contain a returned asynchronous cross-invalidation sequence number (CAAASYNXISEQNUM).
- When the value of PLISTVER is 4 or above, the minimum answer area length is CAALEVEL1LEN.

- When the value of PLISTVER is 0 - 3, the minimum answer area length is CAALEVELLEN.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 2-byte field that contains the length of the answer area (ANSAREA).

**,BUFADDRSIZE=31**

**,BUFADDRSIZE=64**

Use this input parameter to specify whether a 31-bit or a 64-bit address is specified by a BUFLIST entry.

**31**

The entry in BUFLIST is 31 bits in size.

**64**

The entry in BUFLIST is 64 bits in size.

**,BUFADDRTYPE=VIRTUAL**

**,BUFADDRTYPE=REAL**

Use this input parameter to specify whether the buffer addresses specified in the BUFLIST list are virtual storage or real storage addresses.

**VIRTUAL**

The buffer addresses are virtual storage addresses. The virtual storage can be pageable or nonpageable. See the PAGEABLE parameter for information about managing storage binds when specifying virtual storage addresses.

**REAL**

The buffer addresses are real storage addresses.

It is the caller's responsibility to manage the binds between the data buffer virtual storage and the real storage addresses provided. The caller must ensure that the data buffer virtual storage remains bound to the real storage addresses provided until the request completes.

**,BUFALET=NO\_BUFALET**

**,BUFALET=*bufalet***

Use this input parameter to specify an access list entry token (ALET) to be used in referencing all of the buffers specified by BUFLIST.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 4-byte field that contains the ALET.

**,BUFFER=*buffer***

Use this output parameter to specify a buffer area to contain the data that is read from the specified cast-out class.

You must ensure that the storage area specified by BUFFER:

- Is a multiple of 4096 bytes.
- Is less than or equal to 65536 bytes.
- Starts on a 4096-byte boundary.
- Does not start below storage address 512.

Use the BUFSIZE description for specifying the size of the buffer.

Upon completion of the request, the buffer is arranged in 32-byte segments, starting at offset zero. The mapping of the buffers is described by the IXLYCANB mapping macro. For each named data item reported on, the following information is returned to the buffer:

- The data item name
- The user data in the directory entry
- The size of the data entry (that is, the number of elements in the data entry)

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an area (with a length of BUFSIZE) to receive the requested information.



**,BUFLIST=buflist**

Use this output parameter to specify a list of buffers to hold the data that is read from the specified cast-out class. BUFLIST specifies a 128-byte storage area that consists of a list of 0 to 16 buffer addresses.

The format of the list is a set of 8-byte elements. The BUFADDRSIZE keyword denotes whether four or eight bytes of the element are used.

- If BUFADDRSIZE=31 is specified, then the first four bytes of each element are reserved space and the last four bytes contain the address of the buffer.
- If BUFADDRSIZE=64 is specified, then the full eight bytes specify the address of the buffer.

**The BUFLIST buffers must:**

- Reside in the same address space or same data space.
- Be 4096 bytes.
- Start on a 4096-byte boundary.
- Not start below storage address 512.

**Note:** The buffers do not have to be contiguous in storage. Cache services treat BUFLIST buffers as a single buffer even if the buffers are not contiguous.

Use the BUFNUM parameter to specify the number of buffers in the buffer list.

See *z/OS MVS Programming: Sysplex Services Guide* for more information on buffers.

Upon completion of the request, the buffers are arranged in 32-byte segments, starting at offset zero. The mapping of the buffers is described by the IXLYCANB mapping macro. For each named data item reported on, the following information is returned to the buffers:

- The data item name
- The user data in the directory entry
- The size of the data entry

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 128-byte area that contains a list of buffer addresses.

**,BUFNUM=bufnum**

Use this input parameter to specify the number of buffers in the BUFLIST list. Valid BUFNUM values are 0 - 16. A value of zero indicates that no data is to be read into the buffers.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 1-byte field that contains the number of buffers (0 - 16) in the list (BUFLIST).

**,BUFSIZE=bufsize**

Use this input parameter to specify the size of the BUFFER area. See the BUFFER parameter description for valid buffer sizes.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a fullword field that contains the size of the buffer (BUFFER) in bytes.

**,BUFSTGKEY=CALLERS\_KEY****,BUFSTGKEY=bufstgkey**

Use this input parameter to specify a storage key that you define and use when referencing the buffers specified by BUFLIST or the buffer that is specified by BUFFER.

If you do not specify BUFSTGKEY, or if you specify BUFSTGKEY=CALLERS\_KEY, all references to one or more buffers are performed by using the caller's PSW key.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of an 8-bit field that contains the storage key in the format B'kkkkxxxx', where kkkk is the key and xxxx is ignored.

**,COCLASS=coclass**

Use this input parameter to specify the cast-out class for which directory names and directory entry user data should be retrieved.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 2-byte field that contains the cast-out class value.

**,CONTO=conken**

Use this input parameter to specify the connect token that was returned by the IXLCONN service in the IXLCONN answer area, which is mapped which is by IXLYCONA. The connect token uniquely identifies your connection to the cache structure, and must be specified on each IXLCACHE invocation.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field that contains the connect token.

**,DIRINFOFMT=NAMELIST**

**,DIRINFOFMT=DIRENTRYLIST**

Use this input parameter to specify the amount of information to be returned for each directory entry.

DIRINFOFMT is meaningful only for structures allocated in a coupling facility of CFLEVEL=5 or higher.

**NAMELIST**

Return a subset of the information for each directory. The information returned includes:

- Name
- User data
- Size of the data entry.

**DIRENTRYLIST**

Return all information for each directory. The information returned includes:

- Name
- User data
- Storage class assigned to
- Indication of whether data is currently cached
- Indication of whether data is changed (if cached)
- Castout class assigned to (if changed)
- Parity bits
- Value and state of the cast-out lock
- Indication of which connected users have registered interest
- Size of the data entry
- Version number.

See the IXLYDEIB macro for the format of the data returned in the BUFFER area or the BUFLIST buffers.

**,EXTRESTOKEN=NO\_EXTRESTOKEN**

**,EXTRESTOKEN=extrestoken**

Use this input parameter to specify an extended restart token that can be used to resume processing of a READ\_COCLASS request that completed prematurely because it exceeded the model-dependent time-out criteria or because the specified buffer area is full. The extended restart token is returned in the answer area (field CAAEXTRESTOKEN), and should be specified on the next READ\_COCLASS request to resume processing with the next data item to be processed.

If the request does not exceed the model-dependent time-out criteria or the buffer does not become full, the extended restart token will not be provided.

**Note:**

1. Specifying an extended restart token of all zeros causes cache services to treat all of the entries as unprocessed.
2. Do not specify an extended restart token other than the one returned in the answer area or one set to all zeros, because results will be unpredictable.

3. Specifying an extended restart token requires that the length of the answer area be at least the length of CAALEVEL1LEN.

Requestors that specify IXLCONN ALLOWAUTO=YES must use the extended restart token. Requestors that specify or default to IXLCONN ALLOWAUTO=NO must use the standard restart token (RESTOKEN).

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-byte field that contains the extended restart token.

```
,MF=S
,MF=(L,mfctrl)
,MF=(L,mfctrl,mfattr)
,MF=(L,mfctrl,0D)
,MF=(M,mfctrl)
,MF=(M,mfctrl,COMPLETE)
,MF=(M,mfctrl,NOCHECK)
,MF=(E,mfctrl)
,MF=(E,mfctrl,COMPLETE)
,MF=(E,mfctrl,NOCHECK)
```

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

**,mfctrl**

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

**,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

**,COMPLETE**

**,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

**COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if SMILE=*var* were an optional parameter and the default is SMILE=NO\_SMILE then it would not be documented. However, if the default was SMILE=-), then it would be documented because a value would be the default.

**NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

**,MODE=SYNCSUSPEND**

**,MODE=SYNCECB**

**,MODE=SYNCEXIT**

**,MODE=SYNCTOKEN**

**,MODE=ASYNCECB**

**,MODE=ASYNCEXIT**

**,MODE=ASYNCTOKEN**

Use this input parameter to specify:

- Whether the request is to be performed synchronously or asynchronously
- How you want to be notified of request completion if the request is processed asynchronously.

See *z/OS MVS Programming: Sysplex Services Guide* for more information on understanding synchronous and asynchronous cache operations.

**SYNCSUSPEND**

The request processes synchronously. If necessary, the request is suspended until it can complete synchronously. To use this option, your program must be enabled for I/O and external interrupts.

**SYNCECB**

The request processes synchronously if possible. If the request processes asynchronously, the ECB specified by REQECB is posted when the request completes.

**SYNCEXIT**

The request processes synchronously if possible. If the request processes asynchronously, your complete exit is given control when the request completes.

**SYNCTOKEN**

The request processes synchronously if possible. If the request processes asynchronously, an asynchronous request token is returned to the area specified by REQTOKEN. Use the returned request token on the IXLFCOMP macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=SYNCTOKEN is specified.

**ASYNCECB**

The request processes asynchronously. The ECB specified by REQECB is posted when the request completes.

**ASYNCEXIT**

The request processes asynchronously. Your complete exit is given control when the request completes.

**ASYNCTOKEN**

The request processes asynchronously. An asynchronous request token is returned to the area specified by REQTOKEN. Use the returned request token on the IXLFCOMP macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=ASYNCTOKEN is specified.

**,NAME=NO\_NAME**

**,NAME=name**

Use this input parameter to filter by name, the data item for which directory entry data will be read. You may use this parameter along with the NAMEMASK parameter to select certain data items. See the NAMEMASK parameter for more information on this option.

The entry name and user data associated with NAME, possibly NAMEMASK, and the specified cast-out class (COCLASS) are retrieved. If NAME is not specified, all the data items in the specified cast-out class are read.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-byte field that contains the name of the data item.

**,NAMEMASK=1111111111111111**

**,NAMEMASK=namemask**

Use this input parameter to specify which characters in the name specified by NAME are to be used in selecting data items for processing. This parameter allows you to select multiple data items based on common characters in NAME.

The position of each bit in NAMEMASK corresponds to the same relative character position in NAME. A one indicates that the corresponding letter should be used in selecting entries; a zero indicates that the corresponding letter should not be used.

Specifying a name mask with all zeros causes all names to be selected for processing. Specifying a name mask with all ones causes only the name specified by NAME to be selected.

For more information on how NAMEMASK may be used, see [z/OS MVS Programming: Sysplex Services Guide](#).

**To Code:** Specify the RS-type name or address (by using a register 2 - 12 2 to ) of a 2-byte field that contains the bit-mask for the name specified on the NAME keyword.

**,PAGEABLE=YES**

**,PAGEABLE=NO**

Use this input parameter to identify whether the storage areas specified by BUFFER or BUFLIST are in pageable or potentially pageable storage.

#### **YES**

Specify this option to indicate that the BUFFER or BUFLIST buffers reside in pageable virtual storage. XES performs the required page fixing to fix the buffers in real storage while the cache or list request transfers data to or from the coupling facility.

This includes storage obtained from pageable subpools, disabled reference (DREF) storage, and may include storage that has the potential to become pageable during the processing of a request. (An example is address space storage owned by any swappable address space, for which a PGSER FIX has been successfully processed, but for which the owning address space gets swapped during processing of a cache or list request.) This does not include implicitly non-pageable storage (for example, storage obtained from non-pageable subpools).

The system takes responsibility for managing binds to central storage for the duration of the cache or list request, regardless of what address space owns the storage or whether the storage-owning address space is swappable or nonswappable. The storage can be owned by any address space.

High shared virtual storage areas (above 2GB) may not be used.

#### **NO**

Specify this option to indicate that the BUFFER or BUFLIST buffers reside in non-pageable virtual storage. XES does not page fix the buffers in real storage.

This includes implicitly non-pageable storage areas (for example, storage obtained from non-pageable subpools), and may include storage that has the potential to become pageable during the processing of a request (An example is address space storage owned by any swappable address space, for which a PGSER FIX has been successfully processed, but for which the owning address space gets swapped-out during processing of a cache or list request.)

The system takes responsibility for managing binds to central storage for the duration of the cache or list request, if and only if the non-pageable storage is owned by either the requester's address space or the connector's address space. If the storage is owned by any other address space, then the invoker is responsible for ensuring that the virtual storage remains non-pageable for the duration of the request (including the case in which the storage is owned by a swappable address space that is swapped during processing of an IXLCACHE or IXLLIST request). Subject to this consideration, the storage can be owned by any address space. See [z/OS MVS Programming: Sysplex Services Guide](#).

**,PLISTVER=IMPLIED\_VERSION****,PLISTVER=MAX****,PLISTVER=*plistver***

Use this input parameter to specify the version of the macro. See “[Understanding IXLCACHE Version Support](#)” on page 461 for a description of the options available with PLISTVER.

**,REQDATA=NO\_REQDATA****,REQDATA=*reqdata***

Use this input parameter with MODE=SYNCEXIT or MODE=ASYNCEXIT to pass any data you choose to the complete exit. The exit will get control only if the request is processed asynchronously.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the data to be passed to the complete exit.

**,REQECB=*reqecb***

Use this output parameter with either MODE=SYNCECB or MODE=ASYNCECB to specify the address of an ECB, which is to be posted when the request completes if the request was processed asynchronously.

Before coding REQECB, you must ensure that:

- You initialize the ECB before you issue the request.
- The ECB resides in either common storage or the home address space where IXLCONN was issued.
- Any tasks that wait for the ECB to be posted reside in the home address space where IXLCONN was issued.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that contains the address of the ECB to be posted when the request completes. The ECB must be aligned on a fullword boundary.

**,REQID=NO\_REQID****,REQID=*reqid***

Use this input parameter to specify a user-defined request identifier to be associated with the request. You can specify this request identifier on the IXPURGE macro to cancel a request that has not yet been processed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the user-defined request identifier.

**,REQTOKEN=*reqtoken***

Use this output parameter with either MODE=SYNCTOKEN or MODE=ASYNCTOKEN to specify the address of a storage area to receive the request token that is returned when the request will be processed asynchronously. This token, which uniquely identifies the request, must be used as input to the IXLFComp macro, which you use to determine if the request has completed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field where the system will put the request token.

**,RESTOKEN=NO\_RESTOKEN****,RESTOKEN=*restoken***

Use this input parameter to specify a restart token that can be used to resume processing of a READ\_COCLASS request that completes prematurely because it exceeds the model-dependent time-out criteria or the specified buffer area (BUFLIST or BUFFER) is full. The restart token is returned in the answer area, and should be specified on the next READ\_COCLASS request to resume processing with the next directory entry to be processed.

If the request does not exceed the model-dependent time-out criteria or the buffer does not become full, the returned token will contain all binary zeros.

**Note:**

1. Specifying an extended restart token of all zeros causes cache services to treat all of the entries as unprocessed.

2. Do not specify a restart token other than the one returned in the answer area or one set to all zeros, because results will be unpredictable.

Requestors that specify or default to IXLCONN ALLOWAUTO=NO must use the standard restart token. Requestors that specify IXLCONN ALLOWAUTO=YES must use the extended restart token (EXTRESTOKEN).

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-byte field that contains the restart token.

**,RTATYPE=NORMAL**

**,RTATYPE=ENHANCED**

Use this input parameter to specify the restart token algorithm that can be used to resume processing of a READ\_COCLASS request that completed prematurely.

#### **NORMAL**

Use the normal restart token algorithm. The normal restart token algorithm might return duplicate entries that match the user's requested filtering criteria in some cases, and also miss entries that match the user's requested filtering criteria in some cases. When using this option, be prepared to handle any duplicate entries that might be returned.

#### **ENHANCED**

Use the enhanced restart token algorithm. The enhanced restart token algorithm might return duplicate entries that match the user's requested filtering criteria in some cases; however, unlike the normal algorithm, it will never miss any entries that match the requested filtering criteria. When using this option, be prepared to handle any duplicate entries that may be returned.

**Note:** The request to use the enhanced restart token algorithm might or might not be honored, depending on the presence or absence of the required support in the coupling facility that contains the cache structure. CaaEnhancedRtAlgPresent, the cache answer area field, will indicate whether the coupling facility support for the enhanced restart token algorithm is present and thus, whether it is possible to honor this request.

**,RETCODE=retcode**

Use this output parameter to specify a field to contain the return code. (The return code is also returned in register 15.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the return code when the request has completed.

**,RSNCODE=rsncode**

Use this output parameter to specify a field to contain the reason code returned, if applicable. (The reason code is also returned in register 0.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the reason code (if any) when the request has completed.

## ABEND Codes

---

Abend X'026' (See *z/OS MVS System Codes* for more information on this abend.)

## Return and Reason Codes

---

When the system returns control to your program:

- GPR 15 (and RETCODE, if specified) contains a return code.
- If the return code is not zero, GPR 0 (and RSNCODE, if specified) contains a reason code.

**Note:** The return and reason codes will also be put into the answer area (mapped by IXLYCAA), if there is one, when the request completes.

The IXLYCON macro provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

**0**

IXLRETCODEOK

**4**

IXLRETCODEWARNING

**8**

IXLRETCODEPARMERROR

**C**

IXLRETCODEENVERROR

**10**

IXLRETCODECOMPERROR

The following table contains hexadecimal return and reason codes, the equate symbols associated with each reason code, and the meaning and suggested action for each return and reason code.

Table 36. Return and Reason Codes for the IXLCACHE REQUEST=READ_COCLASS Macro		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0	None.	<p><b>Meaning:</b> If you specified a MODE value of ASYNCECB, ASYNCEXIT, or ASYNCTOKEN, the request has been successfully initiated. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed. For any other MODE value, the request has successfully completed.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=ASYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=ASYNCEXIT, the connection's complete exit will be given control when the request has completed.</li> <li>• If you specified MODE=ASYNCTOKEN, use REQTOKEN when calling IXLFCOMP to determine when the request has completed.</li> </ul>



Table 36. Return and Reason Codes for the IXLCACHE REQUEST=READ\_COCLASS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	xxxx0402	<p><b>Equate Symbol:</b> IXLRNCODEASYNCH</p> <p><b>Meaning:</b> The request could not be processed synchronously. It will be processed asynchronously. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed.</p> <ul style="list-style-type: none"> <li>• If you specified MODE=SYNCECB, the ECB specified in REQECB will be posted when the request has finished.</li> <li>• If you specified MODE=SYNCEXIT, the connection's complete exit will be given control when the request has finished.</li> <li>• If you specified MODE=SYNCTOKEN, the REQTOKEN contains an asynchronous request token that may be used on the IXLFCOMP macro to determine when the request has completed.</li> </ul> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=SYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=SYNCEXIT, the connection's completion exit will be given control when the request has completed.</li> <li>• If you specified MODE=SYNCTOKEN, use REQTOKEN when calling IXLFCOMP to determine when the request has completed.</li> </ul>
4	xxxx0409	<p><b>Equate Symbol:</b> IXLRNCODETIMEOUT</p> <p><b>Meaning:</b> The request has completed prematurely because the model-dependent time-out criteria of the coupling facility has been exceeded. The following information has been returned in the answer area:</p> <ul style="list-style-type: none"> <li>• The number of elements returned in the BUFFER or BUFLIST area (field CAADIRCOUNT)</li> <li>• A token for restarting the request (field CAARESTOKEN or CAAEXTRESTOKEN).</li> </ul> <p><b>Action:</b> Reissue the request using the restart token (RESTOKEN or CAAEXTRESTOKEN).</p> <p>Be sure to process the information returned from this request before reissuing the request. The data returned from this request will be overwritten if you specify the same buffer address. Continue to reissue the request until the return code indicates that all processing has completed.</p> <p>For more information about premature request completion, see <a href="#">z/OS MVS Programming: Sysplex Services Guide</a>.</p>

Table 36. Return and Reason Codes for the IXLCACHE REQUEST=READ\_COCLASS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	xxxx040F	<p><b>Equate Symbol:</b> IXLRNCODEBUFFERFULL</p> <p><b>Meaning:</b> The request completed prematurely because the buffer specified by BUFFER, or the buffers specified by BUFLIST, is full. The following information has been returned in the answer area:</p> <ul style="list-style-type: none"> <li>• The number of elements returned in the BUFFER or BUFLIST area (field CAADIRCOUNT)</li> <li>• A token for restarting the request (field CAARESTOKEN or CAAEXTRESTOKEN)</li> </ul> <p><b>Action:</b> Reissue the request, or increase the size of the buffer(s) and rerun your program. You can reissue the request using the restart token (RESTOKEN or EXTRESTOKEN).</p> <p>Be sure to process the information returned from this request before reissuing the request. The data returned from this request will be overwritten if you specify the same buffer address. Continue to reissue the request until the return code indicates that all processing has completed.</p> <p>For more information about premature request completion, see <a href="#">z/OS MVS Programming: Sysplex Services Guide</a>.</p>
8	xxxx0801	<p><b>Equate Symbol:</b> IXLRNCODEBADPARMLIST</p> <p><b>Meaning:</b> The parameter list (mfctrl) for this request is not addressable. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the parameter list address.</li> <li>• The parameter list must be addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are issuing this macro while disabled, the parameter list must reside in either page-fixed or DREF storage.</li> <li>• If you are issuing this macro in AR-mode and you specified the parameter list address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are issuing this macro in AR-mode, SYSSTATE ASCENV=AR must be issued before this macro.</li> </ul>

Table 36. Return and Reason Codes for the IXLCACHE REQUEST=READ\_COCLASS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0804	<p><b>Equate Symbol:</b> IXLRSNCODEBADVERSION#</p> <p><b>Meaning:</b> The version number in the macro parameter list is not compatible with the level of XES currently being used. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that your program did not overlay the parameter list storage.</li> <li>• Verify that your program was assembled with the correct macro library for the release of MVS on which your program is running.</li> <li>• Verify that your program is running on an MVS system that supports the version of the macro you are using.</li> </ul>

Table 36. Return and Reason Codes for the IXLCACHE REQUEST=READ\_COCLASS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx080A	<p><b>Equate Symbol:</b> IXLSRNCODEBADCONTOKEN</p> <p><b>Meaning:</b> Program error. The specified connect token (CONTOKEN) was not valid for one of the following reasons:</p> <ol style="list-style-type: none"> <li>1. The user with the connection identifier represented by the token has disconnected from the structure.</li> <li>2. The connector's task (the task that issued IXLCONN) ended.</li> <li>3. The specified token is not the token that was returned from IXLCONN.</li> <li>4. The request was issued from an address space other than the address space in which IXLCONN was issued.</li> <li>5. The connect token was invalidated during rebuild.</li> <li>6. The connect token was invalidated by XES.</li> </ol> <p><b>Note:</b> The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Verify that the CONTOKEN value specified was valid and for the correct structure. The numbers on the following actions correspond to the numbers above.</p> <ol style="list-style-type: none"> <li>1. Discontinue use. Perform recovery and cleanup for the structure and check protocol for use of structure to determine why you did not know of the disconnection.</li> <li>2. Discontinue use. Perform recovery and cleanup for the structure and check protocol for use of structure to determine why you did not know that the task ended.</li> <li>3. Issue IXLCONN to connect to the structure. The contoken is returned in the CONACONTOKEN field of the answer area specified on the IXLCONN request.</li> <li>4. Issue your request from the same address space the IXLCONN was issued.</li> <li>5. Participate in the rebuild. When it is complete, try again.</li> <li>6. Discontinue use. Perform recovery and cleanup for the structure and check protocol for use of structure.</li> </ol> <p>You may want to issue IXCQUERY to get more information about the structure.</p>
8	xxxx0824	<p><b>Equate Symbol:</b> IXLSRNCODEWRONGSTRTYPE</p> <p><b>Meaning:</b> The connect token specified by CONTOKEN is not to a cache structure.</p> <p><b>Action:</b> Verify the connect token for this cache structure.</p> <p><b>Note:</b> The connect token can be found in the IXLCONN answer area. Verify the value specified on the TYPE= parameter of the IXLCONN request for this structure.</p>

Table 36. Return and Reason Codes for the IXLCACHE REQUEST=READ\_COCLASS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx082E	<p><b>Equate Symbol:</b> IXLRNCODEBADCOCLASS</p> <p><b>Meaning:</b> Program error. The cast-out class specified by COCLASS exceeds the maximum number of cast-out classes defined for the cache structure.</p> <p><b>Action:</b> Correct the COCLASS parameter to specify a valid cast-out class. The valid range for COCLASS is from 1 to the maximum value specified by the NUMCOCLASS parameter on the IXLCONN macro.</p>
8	xxxx0833	<p><b>Equate Symbol:</b> IXLRNCODEBADPGBLATTR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list is specified as being pageable (PAGEABLE=YES) but is not.</p> <p><b>Action:</b> Change the buffer area(s) to pageable storage, or specify PAGEABLE=NO. See the PAGEABLE parameter description for specification instructions.</p>
8	xxxx0834	<p><b>Equate Symbol:</b> IXLRNCODEBADNONPGBLATTR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list is specified as being nonpageable (PAGEABLE=NO) but is either pageable or not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The type of storage in the buffer area corresponds to the value you specified on the PAGEABLE parameter. See the PAGEABLE parameter description for specification instructions.</li> <li>• The correct buffer address was used.</li> <li>• The buffer area(s) were not previously freed.</li> <li>• If BUFLIST was specified and your program is running in AR mode, ensure that: <ul style="list-style-type: none"> <li>– The BUFALET specification is correct.</li> <li>– You specified SYSSTATE ASCENV=AR before issuing the macro.</li> </ul> </li> <li>• If the caller is running in AR-mode and the BUFFER or BUFLIST parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> </ul>

Table 36. Return and Reason Codes for the IXLCACHE REQUEST=READ\_COCLASS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0835	<p><b>Equate Symbol:</b> IXLRNCODEBADDATAADDR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list, is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The correct buffer address was used for BUFFER or for a buffer within the BUFLIST.</li> <li>• The buffer area(s) were not previously freed.</li> <li>• The buffer area(s) were allocated in a storage key that matches the key specified using the BUFSTGKEY parameter, or, if BUFSTGKEY is omitted, the caller's PSW key.</li> <li>• If the caller is running in AR-mode and the BUFFER or BUFLIST parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If the caller is running in AR-mode, SYSSTATE ASCENV=AR must be specified before issuing this macro.</li> <li>• If BUFLIST was specified and your program is running in AR mode the BUFALET specification is correct.</li> </ul>
8	xxxx0836	<p><b>Equate Symbol:</b> IXLRNCODEBADREALADDR</p> <p><b>Meaning:</b> Program error. Real storage addresses were provided in a BUFLIST list, but one of the buffers is not addressable in central storage.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that BUFADDRTYPE was specified as you intended.</li> <li>• Ensure that the real buffer addresses specified by BUFLIST are valid.</li> </ul>
8	xxxx0838	<p><b>Equate Symbol:</b> IXLRNCODEBADANSAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by ANSAREA is not addressable.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the ANSAREA address.</li> <li>• ANSAREA must be addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are calling IXLCACHE while disabled, ANSAREA must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLCACHE in AR-mode and you specified the ANSAREA address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are calling IXLCACHE in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLCACHE macro.</li> </ul>

Table 36. Return and Reason Codes for the IXLCACHE REQUEST=READ\_COCLASS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0839	<p><b>Equate Symbol:</b> IXLRNCODEBADREQTOKENAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by REQTOKEN is not addressable.</p> <p><b>Action:</b> Ensure that the request token area specified by REQTOKEN is valid:</p> <ul style="list-style-type: none"> <li>• Verify the REQTOKEN address.</li> <li>• If you are calling IXLCACHE while disabled, REQTOKEN must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLCACHE in AR-mode and you specified the REQTOKEN address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are calling IXLCACHE in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLCACHE macro.</li> </ul>
8	xxxx083D	<p><b>Equate Symbol:</b> IXLRNCODEBADANSLEN</p> <p><b>Meaning:</b> Program error. The length of the answer area, as specified by ANSLEN, is not sufficient to contain answer area information. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Increase the size of the answer area provided for the request, and rerun your program. Depending on the macro version number, specify the length of the answer area as follows:</p> <ul style="list-style-type: none"> <li>• When the connection specified ASYNCXI=1 on the IXLCONN invocation and AXIOVERRIDE=0 was specified or defaulted to for the IXLCACHE request, the answer area length must be a minimum value of CAALEVEL2LEN</li> <li>• When the value of the macro version number (PLISTVER) is 4 or more, the minimum answer area length is CAALEVEL1LEN.</li> <li>• When the value of the macro version number (PLISTVER) is 0-3, the minimum answer area length is CAALEVEL0LEN.</li> </ul>

Table 36. Return and Reason Codes for the IXLCACHE REQUEST=READ\_COCLASS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0849	<p><b>Equate Symbol:</b> IXLRNCODEBADRESTOKEN</p> <p><b>Meaning:</b> Program error. The restart token specified by RESTOKEN is not valid.</p> <p><b>Action:</b> Determine why the restart token cannot be used to resume the request. Possible causes are:</p> <ul style="list-style-type: none"> <li>• The entry to be resumed no longer resides in the cast-out class specified by COCLASS.</li> <li>• The specified token does not correspond to the restart token returned in the answer area of the previous request.</li> <li>• The user specified RESTOKEN when EXTRESTOKEN was required.</li> <li>• The user specified EXTRESTOKEN when RESTOKEN was required.</li> </ul> <p>For more information about premature request completion, see <a href="#">z/OS MVS Programming: Sysplex Services Guide</a>.</p>
8	xxxx0851	<p><b>Equate Symbol:</b> IXLRNCODENOSUSPENDISABLE</p> <p><b>Meaning:</b> Program error. The request failed because MODE=SYNCSUSPEND was specified, but the caller is disabled and cannot be suspended.</p> <p><b>Action:</b> Either specify another MODE value or become enabled (release the CPU lock); then reissue the request.</p>
8	xxxx0864	<p><b>Equate Symbol:</b> IXLRNCODEBADBUFSIZE</p> <p><b>Meaning:</b> Program error. The size of the BUFLIST areas or BUFFER area is not large enough to contain the data being read. No data is returned.</p> <p><b>Action:</b> If more space is available, specify a larger buffer size and reissue the request. Consider using the BUFLIST parameter if BUFFER was specified.</p>
8	xxxx0865	<p><b>Equate Symbol:</b> IXLRNCODEBADBUFSPEC</p> <p><b>Meaning:</b> Program error. There is an error in the buffer specification.</p> <p><b>Action:</b> Check the following:</p> <ul style="list-style-type: none"> <li>• If BUFLIST was specified, check the requirements for BUFLIST and BUFNUM.</li> <li>• If BUFFER was specified, check the requirements for BUFFER and BUFSIZE.</li> <li>• Buffer pointer(s) in BUFLIST.</li> <li>• Buffer boundaries.</li> </ul>



Table 36. Return and Reason Codes for the IXLCACHE REQUEST=READ\_COCLASS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0866	<p><b>Equate Symbol:</b> IXLRNCODEBADBUFKEY</p> <p><b>Meaning:</b> Program error. The buffer storage key specified by BUFSTGKEY is incorrect or BUFSTGKEY was not specified and the PSW key does not match the key of the buffers.</p> <p>The data cannot be stored in the specified buffer area.</p> <p><b>Action:</b> Check the following:</p> <ul style="list-style-type: none"> <li>• Determine if the key of the storage being used for the buffers is different from the PSW key.</li> <li>• If BUFSTGKEY was specified, verify that the key was put in the correct bits (see the explanation of this parameter for more information).</li> <li>• If you are calling IXLCACHE in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLCACHE macro.</li> </ul>
8	xxxx0867	<p><b>Equate Symbol:</b> IXLRNCODEBADBUFLIST</p> <p><b>Meaning:</b> Program error. The 128-byte storage area specified by BUFLIST is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The correct BUFLIST address was used.</li> <li>• The BUFLIST area was not previously freed.</li> <li>• If the caller is running in AR-mode and the BUFLIST parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If the caller is disabled, then the BUFLIST must reside in either nonpageable or disabled reference storage.</li> <li>• If your program is running in AR mode, ensure that you specified SYSSTATE ASCENV=AR before issuing the macro.</li> </ul>
8	xxxx0887	<p><b>Equate Symbol:</b> IXLRNCODEBADEXTRESTOKEN</p> <p><b>Meaning:</b> Program error. The extended restart token specified by EXTRESTOKEN is not valid. The specified token refers to an older instance of the target structure. A system-managed process occurred between the time a request returned the extended restart token and the time the connector tried to continue the request using that token.</p> <p><b>Action:</b> Discard the results of the initial request and reissue the request with an EXTRESTOKEN value of zero. For more information about restarting requests, see <a href="#">z/OS MVS Programming: Sysplex Services Guide</a>.</p>

Table 36. Return and Reason Codes for the IXLCACHE REQUEST=READ\_COCLASS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C06	<p><b>Equate Symbol:</b> IXLRNOCODENOCONE</p> <p><b>Meaning:</b> Environmental error. No connectivity to the cache structure exists. This may occur due to operator commands such as VARY PATH,OFFLINE or CONFIG CHP,OFFLINE or hardware errors such as facility or path failures. The CONTOKEN is invalidated. The request fails.</p> <p><b>Action:</b> Either disconnect from the structure (using IXLDISC) or initiate a rebuild of the structure, if allowed (using IXLREBLD).</p>
C	xxxx0C13	<p><b>Equate Symbol:</b> IXLRNOCODEREQPURGED</p> <p><b>Meaning:</b> Environmental error. The request was purged prior to completion of the request. Possible reasons include:</p> <ul style="list-style-type: none"> <li>• The connector failed.</li> <li>• The connector disconnected.</li> <li>• The requestor failed.</li> <li>• The request was purged by IXLPURGE.</li> <li>• Requests were purged when the connector provided an IXLEERSP response for the Rebuild Quiesce, Rebuild Stop, or Rebuild Cleanup event.</li> <li>• The connector invoked IXLREBLD REQUEST=COMPLETE.</li> <li>• The secondary address space was no longer valid.</li> </ul> <p><b>Action:</b> None if this is expected. Otherwise, determine why the connector failed.</p>
C	xxxx0C14	<p><b>Equate Symbol:</b> IXLRNOCODESTATUSUNKNOWN</p> <p><b>Meaning:</b> Environmental error. The IXLCACHE request has completed, but the final disposition of the request cannot be determined. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Verify the validity of your data by comparing the expected results with what is in the coupling facility.</p>
C	xxxx0C25	<p><b>Equate Symbol:</b> IXLRNOCODESTRFAILURE</p> <p><b>Meaning:</b> Environmental error. The cache structure failed prior to completion of the request.</p> <p><b>Action:</b> Attempt to rebuild the structure using IXLREBLD, or disconnect from the structure using IXLDISC.</p>
C	xxxx0CA0	<p><b>Equate Symbol:</b> IXLRNOCODEQUIESCEDSUSPENDFAIL</p> <p><b>Meaning:</b> Environmental error. The request is failed because the structure is quiesced for a system-managed process and SUSPEND=FAIL is specified on the IXLCONN.</p> <p><b>Action:</b> None, if this is expected.</p>

Table 36. Return and Reason Codes for the IXLCACHE REQUEST=READ\_COCLASS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxxFFFF	<p><b>Equate Symbol:</b> IXLRSNCODENOTAVAILABLE</p> <p><b>Meaning:</b> Environmental error. XES functions are not available. The coupling facility hardware might not be present.</p> <p><b>Action:</b> XES may not be used when XCF is running local mode. XES requires that the coupling facility be installed.</p>
10	xxxx10xx	<p><b>Equate Symbol:</b> IXLRSNCODECOMPERROR</p> <p><b>Meaning:</b> System error. XES processing failure. The state of the structure and the disposition of the request are unpredictable. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Save the reason code information, and contact the IBM support center.</p>



## Chapter 39. IXLCACHE REQUEST=READ\_COSTATS

### Description

A READ\_COSTATS request allows you to retrieve statistical information for the specified cast-out classes in the range beginning with COCLASSB and ending with COCLASSE. The information is returned in the storage areas specified by BUFLIST or BUFFER. The format of the information returned depends on the level of the coupling facility in which the structure is allocated and on the COSTATSFMT specification. For cache structures allocated in a coupling facility of CFLEVEL=5 or higher, you can use the COSTATSFMT keyword to specify the level of detailed information that is to be returned.

For structures allocated in a coupling facility of CFLEVEL=4 or lower, or if COSTATSFMT=COCOUNTSLIST is specified, the format of the information returned is described by the CCIH and CCIHCOUNTS mappings of the IXLYCCIH mapping macro and consists of:

- The first and last cast-out classes for which information was returned.
- For each cast-out class for which information was returned, the number of data elements associated with data entries in the indicated cast-out class.

For structures allocated in a coupling facility of CFLEVEL=5 or higher, when COSTATSFMT=COSTATSLIST, the format of the information returned is described by the CCIH, CCIHCOSTATSLIST, and CCIHCCIBS mappings of the IXLYCCIH mapping macro. The information returned consists of:

- The first and last cast-out classes for which information was returned.
- For each cast-out class for which information was returned, the number of data elements associated with data entries in the indicated cast-out class and the user data that was written to the directory entry when the IXLCACHE WRITE\_DATA request was invoked. If the structure was allocated with a UDF order queue, the field contains the user data of the first entry on the UDF order queue. If the structure was allocated without UDF order queues, the field contains the user date of the first entry on the cast-out class queue.

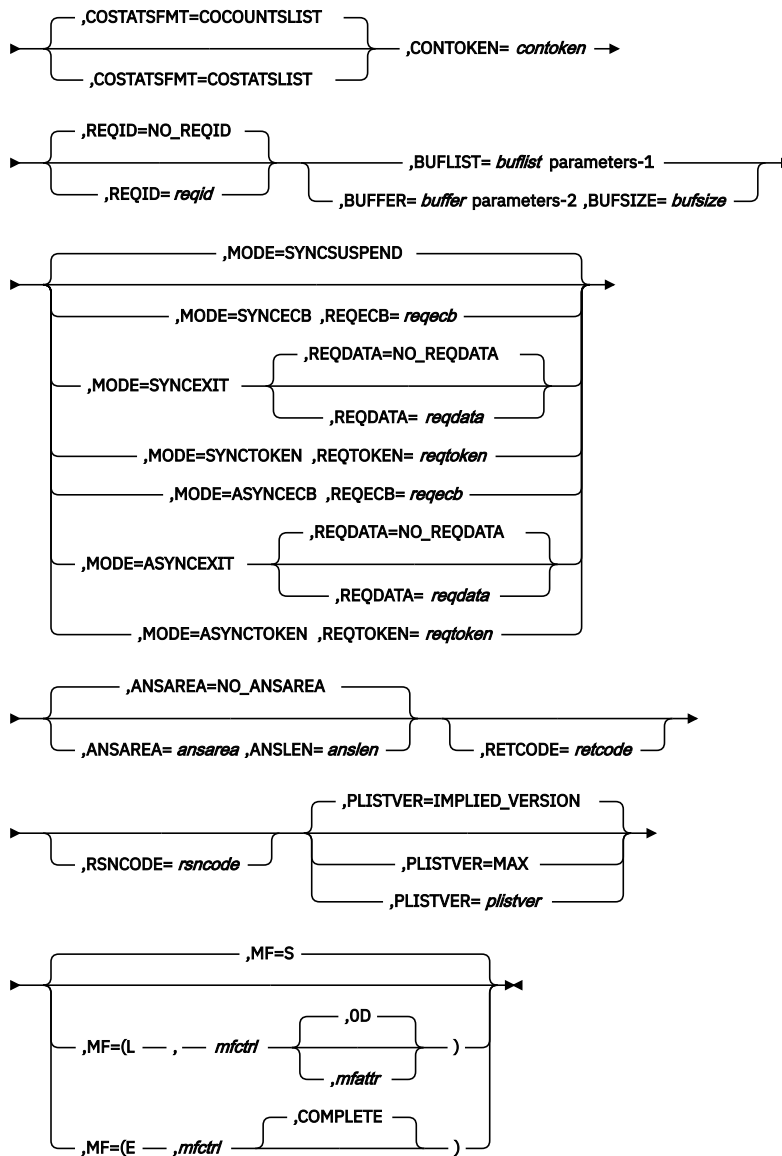
(The UDFORDER keyword of IXLCONN is used to allocate a cache structure with UDF order queues.)

### Syntax Diagram

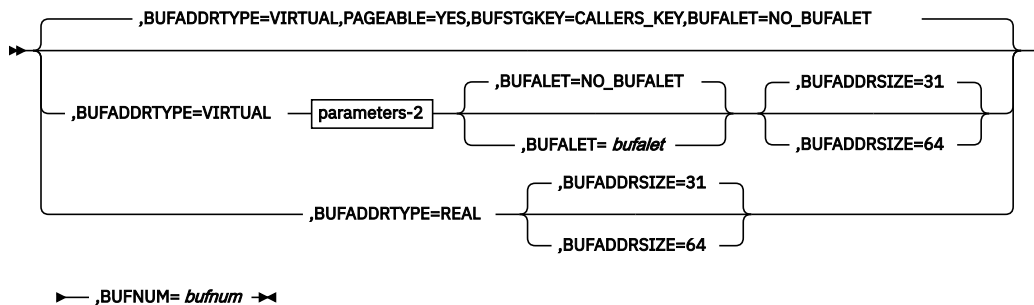
The syntax diagram for IXLCACHE REQUEST=READ\_COSTATS is as follows:

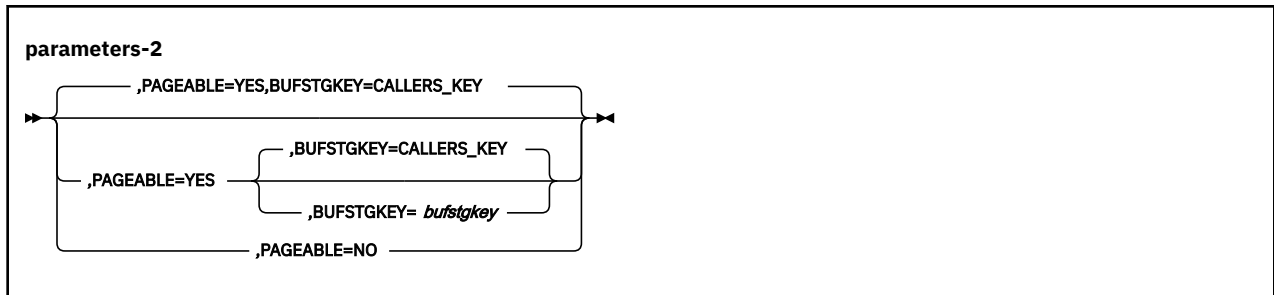
## main diagram

► IXLCACHE — REQUEST=READ\_COSTATS — ,COCLASSB= *coclassb* — ,COCLASSE= *coclasse* ►



## parameters-1





**Note:** When MODE=SYNCTOKEN or MODE=ASYNCTOKEN is specified, ANSAREA= *ansarea* and ANSLen=*anslen* are required.

## Parameter Descriptions

The parameter descriptions for REQUEST=READ\_COSTATS are listed in alphabetical order. Default values are underlined:

### REQUEST=READ\_COSTATS

Use this input parameter to specify that statistical information for the cast-out classes in the range COCLASSB to COCLASSE be retrieved and placed in the storage areas specified by BUFLIST or BUFFER.

#### **,ANSAREA=NO\_ANSAREA**

#### **,ANSAREA=*ansarea***

Use this output parameter to specify an answer area to contain information returned from the request. The format of the answer area is described by the IXLYCAA mapping macro.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an area (with a length of ANSLen) where the information returned from the request will be put.

#### **,ANSLen=*anslen***

Use this inppaameter to specify the size of the storage area specified by ANSAREA.

Use either CAALEVELOLEN, CAALEVEL1LEN or CAALEVEL2LEN of the IXLYCAA mapping macro to determine the minimum size of the answer area. The answer area length must be at least large enough to accommodate the level of the IXLYCAA mapping appropriate to the requested function.

- When the connection specified ASYNCXI=1 on the IXLCONN invocation and AXIOVERRIDE=0 was specified or defaulted to for the IXLCACHE request, the answer area length is a required parameter and must be a minimum value of CAALEVEL2LEN to contain a returned asynchronous cross-invalidation sequence number (CAAASYNXISEQNUM).
- When the value of PLISTVER is 4 or above, the minimum answer area length is CAALEVEL1LEN.
- When the value of PLISTVER is 0 - 3, the minimum answer area length is CAALEVELOLEN.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12 2 to ) of a 2-byte field that contains the length of the answer area (ANSAREA).

#### **,BUFADDRSIZE=31**

#### **,BUFADDRSIZE=64**

Use this input parameter to specify whether a 31-bit or a 64-bit address is specified by a BUFLIST entry.

**31**

The entry in BUFLIST is 31 bits in size.

**64**

The entry in BUFLIST is 64 bits in size.

#### **,BUFADDRTYPE=VIRTUAL**

#### **,BUFADDRTYPE=REAL**

Use this input parameter to specify whether the buffer addresses specified in the BUFLIST list are virtual storage or real storage addresses.

**VIRTUAL**

The buffer addresses are virtual storage addresses. The virtual storage can be pageable or nonpageable. See the PAGEABLE parameter for information about managing storage binds when specifying virtual storage addresses.

**REAL**

The buffer addresses are real storage addresses.

It is the caller's responsibility to manage the binds between the data buffer virtual storage and the real storage addresses provided. The caller must ensure that the data buffer virtual storage remains bound to the real storage addresses provided until the request completes.

**,BUFALET=NO\_BUFALET****,BUFALET=*bufalet***

Use this input parameter to specify an access list entry token (ALET) to be used in referencing all of the buffers specified by BUFLIST.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 4-byte field that contains the ALET.

**,BUFFER=*buffer***

Use this output parameter to specify a buffer area to hold the returned cast-out class statistics.

You must ensure that the storage area specified by BUFFER:

- Is a multiple of 4096 bytes.
- Is less than or equal to 65536 bytes.
- Starts on a 4096-byte boundary.
- Does not start below storage address 512.

See the BUFSIZE description for specifying the size of the buffer.

Upon a successful completion of the request:

- The first four bytes of the buffer contains the start and end cast-out classes for which information was returned. (The format of these four bytes is described by mapping macro IXLYCCIH.)
- Beginning at offset four, the buffer contains an array of fullwords, each corresponding to a cast-out class in the specified range (CCIHCOCLASSBEG to CCIHCOCLASSEND). (Each fullword corresponds to a cast-out class by relative position, such that, for example, the first fullword in the array corresponds to the first cast-out class in the range, and so forth.) Each fullword contains the number of data elements associated with the data entries assigned to this cast-out class.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an area (with a length of BUFSIZE) (the buffer) to contain the data returned from the request.

**,BUFLIST=*buflist***

Use this output parameter to specify a list of buffers to hold the returned cast-out class statistics. BUFLIST specifies a 128-byte storage area that consists of a list of 0 to 16 buffer addresses.

The format of the list is a set of 8-byte elements. The BUFADDRSIZE keyword denotes whether four or eight bytes of the element are used.

- If BUFADDRSIZE=31 is specified, then the first four bytes of each element are reserved space and the last four bytes contain the address of the buffer.
- If BUFADDRSIZE=64 is specified, then the full eight bytes specify the address of the buffer.

**The BUFLIST buffers must:**

- Reside in the same address space or same data space.
- Be 4096 bytes.
- Start on a 4096-byte boundary.
- Not start below storage address 512.



**Note:** The buffers do not have to be contiguous in storage. Cache services treat BUFLIST buffers as a single buffer, even if the buffers are not contiguous.

See the BUFNUM parameter to specify the number of buffers in the buffer list.

Upon successful completion of the request:

- The first four bytes of the buffers contain the start and end cast-out classes for which information was returned. (The format of these four bytes is described by mapping macro IXLYCCIH.)
- Beginning at offset four in the first buffer, the buffers contain an array of fullwords, each corresponding to a cast-out class in the specified range (CCIHCOCLASSBEG to CCIHCOCLASSEND). (Each fullword corresponds to a cast-out class by relative position, such that, for example, the first fullword in the array corresponds to the first cast-out class in the range, and so forth.) Each fullword contains the number of data elements associated with the data entries assigned to this cast-out class.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 128-byte area that contains a list of buffer addresses.

#### **,BUFNUM=*bufnum***

Use this input parameter to specify the number of buffers in the BUFLIST list. Valid BUFNUM values are from 0 to 16. A value of zero indicates that no data is to be read from the buffers.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 1-byte field that contains the number of buffers in the buffer list.

#### **,BUFSIZE=*bufsize***

Use this input parameter to specify the size of the BUFFER area. See the BUFFER parameter description for valid buffer sizes.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a fullword field that contains the size of the buffer (BUFFER) in bytes.

#### **,BUFSTGKEY=CALLERS\_KEY**

#### **,BUFSTGKEY=*bufstgkey***

Use this input parameter to specify a storage key that you define and use when referencing the buffers specified by BUFLIST or the buffer that is specified by BUFFER.

If you do not specify BUFSTGKEY, or if you specify BUFSTGKEY=CALLERS\_KEY, all references to one or more buffers are performed by using the caller's PSW key.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of an 8-bit field that contains the storage key in the format B'kkkkxxxx', where kkkk is the key and xxxx is ignored.

#### **,COCLASSB=*coclassb***

Use this input parameter to specify the first cast-out class in the range of cast-out classes for which statistics will be reported. The specified value must fall within the range 1 to the maximum number of cast-out classes defined for the structure. The value cannot be larger than that specified for COCLASSE.

**Note:** The maximum number of cast-out classes is specified on the IXLCONN macro by the NUMCOCLASS parameter.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 2-byte field that contains the first cast-out class.

#### **,COCLASSE=*coclasse***

Use this input parameter to specify the last cast-out class in the range of cast-out classes for which statistics will be reported. The specified value must fall within the range 1 to the maximum number of cast-out classes defined for the structure. The value cannot be smaller than that specified for COCLASSB.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 2-byte field that contains the last cast-out class.

**,CONTO=*conken***

Use this input parameter to specify the connect token that was returned by the IXLCONN service in the IXLCONN answer area, which is mapped which is by IXLYCONA. The connect token uniquely identifies your connection to the cache structure, and must be specified on each IXLCACHE invocation.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field that contains the connect token.

**,COSTATSFMT=COCOUNTSLIST****,COSTATSFMT=COSTATSLIST**

Use this input parameter to specify the format of the data that is returned for each cast-out class.

COSTATSFMT is meaningful only for structures allocated in a coupling facility with CFLEVEL=5 or higher.

**COCOUNTSLIST**

Return the data in the BUFFER area or the BUFLIST buffers in the format defined by the CCIH and CCIHCOUNTS mappings of the IXLYCCIH macro.

**COSTATSLIST**

Return the data in the BUFFER area or the BUFLIST buffers in the format defined by the CCIH, CCIHCOSTATSLIST, and CCIHCCIBS mappings of the IXLYCCIH macro.

**,MF=S****,MF=(L,*mfctrl*)****,MF=(L,*mfctrl*,*mfattr*)****,MF=(L,*mfctrl*,0D)****,MF=(M,*mfctrl*)****,MF=(M,*mfctrl*,COMPLETE)****,MF=(M,*mfctrl*,NOCHECK)****,MF=(E,*mfctrl*)****,MF=(E,*mfctrl*,COMPLETE)****,MF=(E,*mfctrl*,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

**,*mfctrl***

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

**,*mfattr***

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

**,COMPLETE****,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

**COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if `SMILE=var` were an optional parameter and the default is `SMILE=NO_SMILE` then it would not be documented. However, if the default was `SMILE=-`, then it would be documented because a value would be the default.

**NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

**,MODE=SYNCSUSPEND**

**,MODE=SYNCECB**

**,MODE=SYNCEXIT**

**,MODE=SYNCTOKEN**

**,MODE=ASYNCECB**

**,MODE=ASYNCEXIT**

**,MODE=ASYNCTOKEN**

Use this input parameter to specify:

- Whether the request is to be performed synchronously or asynchronously
- How you want to be notified of request completion if the request is processed asynchronously.

See *z/OS MVS Programming: Sysplex Services Guide* for more information on understanding synchronous and asynchronous cache operations.

**SYNCSUSPEND**

The request processes synchronously. If necessary, the request is suspended until it can complete synchronously. To use this option, your program must be enabled for I/O and external interrupts.

**SYNCECB**

The request processes synchronously if possible. If the request processes asynchronously, the ECB specified by REQECB is posted when the request completes.

**SYNCEXIT**

The request processes synchronously if possible. If the request processes asynchronously, your complete exit is given control when the request completes.

**SYNCTOKEN**

The request processes synchronously if possible. If the request processes asynchronously, an asynchronous request token is returned to the area specified by REQTOKEN. Use the returned request token on the IXLFCOMP macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when `MODE=SYNCTOKEN` is specified.

**ASYNCECB**

The request processes asynchronously. The ECB specified by REQECB is posted when the request completes.

**ASYNCEXIT**

The request processes asynchronously. Your complete exit is given control when the request completes.

**ASYNCTOKEN**

The request processes asynchronously. An asynchronous request token is returned to the area specified by REQTOKEN. Use the returned request token on the IXLFCOMP macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when `MODE=ASYNCTOKEN` is specified.

**,PAGEABLE=YES****,PAGEABLE=NO**

Use this input parameter to identify whether the storage areas specified by BUFFER or BUFLIST are in pageable or potentially pageable storage.

**YES**

Specify this option to indicate that the BUFFER or BUFLIST buffers reside in pageable virtual storage. XES performs the required page fixing to fix the buffers in real storage while the cache or list request transfers data to or from the coupling facility.

This includes storage obtained from pageable subpools, disabled reference (DREF) storage, and may include storage that has the potential to become pageable during the processing of a request. (An example is address space storage owned by any swappable address space, for which a PGSER FIX has been successfully processed, but for which the owning address space gets swapped during processing of a cache or list request.) This does not include implicitly non-pageable storage (for example, storage obtained from non-pageable subpools).

The system takes responsibility for managing binds to central storage for the duration of the cache or list request, regardless of what address space owns the storage or whether the storage-owning address space is swappable or nonswappable. The storage can be owned by any address space.

High shared virtual storage areas (above 2GB) may not be used.

**NO**

Specify this option to indicate that the BUFFER or BUFLIST buffers reside in non-pageable virtual storage. XES does not page fix the buffers in real storage.

This includes implicitly non-pageable storage areas (for example, storage obtained from non-pageable subpools), and may include storage that has the potential to become pageable during the processing of a request (An example is address space storage owned by any swappable address space, for which a PGSER FIX has been successfully processed, but for which the owning address space gets swapped-out during processing of a cache or list request.)

The system takes responsibility for managing binds to central storage for the duration of the cache or list request, if and only if the non-pageable storage is owned by either the requester's address space or the connector's address space. If the storage is owned by any other address space, then the invoker is responsible for ensuring that the virtual storage remains non-pageable for the duration of the request (including the case in which the storage is owned by a swappable address space that is swapped during processing of an IXLCACHE or IXLLIST request). Subject to this consideration, the storage can be owned by any address space. See [z/OS MVS Programming: Sysplex Services Guide](#).

**,PLISTVER=IMPLIED\_VERSION****,PLISTVER=MAX****,PLISTVER=plistver**

Use this input parameter to specify the version of the macro. See [“Understanding IXLCACHE Version Support” on page 461](#) for a description of the options available with PLISTVER.

**,REQDATA=NO\_REQDATA****,REQDATA=reqdata**

Use this input parameter with MODE=SYNCEXIT or MODE=ASYNCEXIT to pass any data you choose to the complete exit. The exit will get control only if the request is processed asynchronously.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the data to be passed to the complete exit.

**,REQECB=reqecb**

Use this output parameter with either MODE=SYNCECB or MODE=ASYNCECB to specify the address of an ECB, which is to be posted when the request completes if the request was processed asynchronously.

Before coding REQECB, you must ensure that:

- You initialize the ECB before you issue the request.

- The ECB resides in either common storage or the home address space where IXLCONN was issued.
- Any tasks that wait for the ECB to be posted reside in the home address space where IXLCONN was issued.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that contains the address of the ECB to be posted when the request completes. The ECB must be aligned on a fullword boundary.

**,REQID=NO\_REQID**

**,REQID=reqid**

Use this input parameter to specify a user-defined request identifier to be associated with the request. You can specify this request identifier on the IXPURGE macro to cancel a request that has not yet been processed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the user-defined request identifier.

**,REQTOKEN=reqtoken**

Use this output parameter with either MODE=SYNCTOKEN or MODE=ASYNCTOKEN to specify the address of a storage area to receive the request token that is returned when the request will be processed asynchronously. This token, which uniquely identifies the request, must be used as input to the IXLFCOMP macro, which you use to determine if the request has completed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field where the system will put the request token.

**,RETCODE=retcode**

Use this output parameter to specify a field to contain the return code. (The return code is also returned in register 15.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the return code when the request has completed.

**,RSNCODE=rsncode**

Use this output parameter to specify a field to contain the reason code returned, if applicable. (The reason code is also returned in register 0.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the reason code (if any) when the request has completed.

## ABEND Codes

---

Abend X'026' (See *z/OS MVS System Codes* for more information on this abend.)

## Return and Reason Codes

---

When the system returns control to your program:

- GPR 15 (and RETCODE, if specified) contains a return code.
- If the return code is not zero, GPR 0 (and RSNCODE, if specified) contains a reason code.

**Note:** The return and reason codes will also be put into the answer area (mapped by IXLYCAA), if there is one, when the request completes.

The IXLYCON macro provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

**0**

IXLRETCODEOK

**4**

IXLRETCODEWARNING

8

IXLRETCODEPARMERROR

C

IXLRETCODEENVERROR

10

IXLRETCODECOMPERROR

The following table contains hexadecimal return and reason codes, the equate symbols associated with each reason code, and the meaning and suggested action for each return and reason code.

Table 37. Return and Reason Codes for the IXLCACHE REQUEST=READ_COSTATS Macro		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0	None.	<p><b>Meaning:</b> If you specified a MODE value of ASYNCECB, ASYNCEXIT, or ASYNCTOKEN, the request has been successfully initiated. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed. For any other MODE value, the request has successfully completed.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=ASYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=ASYNCEXIT, the connection's complete exit will be given control when the request has completed.</li> <li>• If you specified MODE=ASYNCTOKEN, use REQTOKEN when calling IXLFCOMP to determine when the request has completed.</li> </ul>
4	xxxx0402	<p><b>Equate Symbol:</b> IXLRSNCODEASYNCH</p> <p><b>Meaning:</b> The request could not be processed synchronously. It will be processed asynchronously. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed.</p> <ul style="list-style-type: none"> <li>• If you specified MODE=SYNCECB, the ECB specified in REQECB will be posted when the request has finished.</li> <li>• If you specified MODE=SYNCEXIT, the connection's complete exit will be given control when the request has finished.</li> <li>• If you specified MODE=SYNCTOKEN, the REQTOKEN contains an asynchronous request token that may be used on the IXLFCOMP macro to determine when the request has completed.</li> </ul> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=SYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=SYNCEXIT, the connection's completion exit will be given control when the request has completed.</li> <li>• If you specified MODE=SYNCTOKEN, use REQTOKEN when calling IXLFCOMP to determine when the request has completed.</li> </ul>

Table 37. Return and Reason Codes for the IXLCACHE REQUEST=READ\_COSTATS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	xxxx040B	<p><b>Equate Symbol:</b> IXLRSNCODEHIGHCOEND</p> <p><b>Meaning:</b> The specified COCLASSE exceeds the maximum defined cast-out class for the structure. The range of classes from that specified by COCLASSB through the maximum defined class was reported.</p> <p><b>Action:</b> None. However, if you did not intend for the range to end with the maximum defined cast-out class, specify a value for COCLASSE that is within the maximum limit. The number of cast-out classes is defined on the IXLCONN macro with the keyword NUMCOCLASS.</p>
4	xxxx040F	<p><b>Equate Symbol:</b> IXLRSNCODEBUFFERFULL</p> <p><b>Meaning:</b> The request completed prematurely because the buffer specified by BUFFER, or the buffers specified by BUFLIST, is full. A subrange of the requested cast-out classes was reported on, as described by the IXLYCCIH mapping macro.</p> <p><b>Action:</b> Reissue the request specifying the same value for the end cast-out class (COCLASSE), but a start cast-out class (COCLASSB) equal to the end cast-out class plus one (CCIHCOCLASSEND+1).  Be sure to process the information returned from this request before reissuing the request. The data returned from this request will be overwritten if you specify the same buffer address. Continue to reissue the request until the return code indicates that all processing has completed.</p> <p>For more information about premature request completion, see <a href="#">z/OS MVS Programming: Sysplex Services Guide</a>.</p>
8	xxxx0801	<p><b>Equate Symbol:</b> IXLRSNCODEBADPARMLIST</p> <p><b>Meaning:</b> The parameter list (mfctrl) for this request is not addressable. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the parameter list address.</li> <li>• The parameter list must be addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are issuing this macro while disabled, the parameter list must reside in either page-fixed or DREF storage.</li> <li>• If you are issuing this macro in AR-mode and you specified the parameter list address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are issuing this macro in AR-mode, SYSSTATE ASCENV=AR must be issued before this macro.</li> </ul>

Table 37. Return and Reason Codes for the IXLCACHE REQUEST=READ\_COSTATS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0804	<p><b>Equate Symbol:</b> IXLRSNCODEBADVERSION#</p> <p><b>Meaning:</b> The version number in the macro parameter list is not compatible with the level of XES currently being used. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that your program did not overlay the parameter list storage.</li> <li>• Verify that your program was assembled with the correct macro library for the release of MVS on which your program is running.</li> <li>• Verify that your program is running on an MVS system that supports the version of the macro you are using.</li> </ul>



Table 37. Return and Reason Codes for the IXLCACHE REQUEST=READ\_COSTATS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx080A	<p><b>Equate Symbol:</b> IXLSRNCODEBADCONTOKEN</p> <p><b>Meaning:</b> Program error. The specified connect token (CONTOKEN) was not valid for one of the following reasons:</p> <ol style="list-style-type: none"> <li>1. The user with the connection identifier represented by the token has disconnected from the structure.</li> <li>2. The connector's task (the task that issued IXLCONN) ended.</li> <li>3. The specified token is not the token that was returned from IXLCONN.</li> <li>4. The request was issued from an address space other than the address space in which IXLCONN was issued.</li> <li>5. The connect token was invalidated during rebuild.</li> <li>6. The connect token was invalidated by XES.</li> </ol> <p><b>Note:</b> The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Verify that the CONTOKEN value specified was valid and for the correct structure. The numbers on the following actions correspond to the numbers above.</p> <ol style="list-style-type: none"> <li>1. Discontinue use. Perform recovery and cleanup for the structure and check protocol for use of structure to determine why you did not know of the disconnection.</li> <li>2. Discontinue use. Perform recovery and cleanup for the structure and check protocol for use of structure to determine why you did not know that the task ended.</li> <li>3. Issue IXLCONN to connect to the structure. The contoken is returned in the CONACONTOKEN field of the answer area specified on the IXLCONN request.</li> <li>4. Issue your request from the same address space the IXLCONN was issued.</li> <li>5. Participate in the rebuild. When it is complete, try again.</li> <li>6. Discontinue use. Perform recovery and cleanup for the structure and check protocol for use of structure.</li> </ol> <p>You may want to issue IXCQUERY to get more information about the structure.</p>
8	xxxx0824	<p><b>Equate Symbol:</b> IXLSRNCODEWRONGSTRTYPE</p> <p><b>Meaning:</b> The connect token specified by CONTOKEN is not to a cache structure.</p> <p><b>Action:</b> Verify the connect token for this cache structure.</p> <p><b>Note:</b> The connect token can be found in the IXLCONN answer area. Verify the value specified on the TYPE= parameter of the IXLCONN request for this structure.</p>

Table 37. Return and Reason Codes for the IXLCACHE REQUEST=READ\_COSTATS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx082A	<p><b>Equate Symbol:</b> IXLRSNCODEBADCOBEG</p> <p><b>Meaning:</b> Program error. The specified COCLASSB exceeds the maximum number of cast-out classes defined for the structure.</p> <p><b>Action:</b> Specify for COCLASSB a valid index within the range 1 to the maximum cast-out classes defined for the structure. The maximum number of cast-out classes is defined on the IXLCONN macro by the NUMCOCLASS keyword.</p>
8	xxxx0833	<p><b>Equate Symbol:</b> IXLRSNCODEBADPGBLATTR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list is specified as being pageable (PAGEABLE=YES) but is not.</p> <p><b>Action:</b> Change the buffer area(s) to pageable storage, or specify PAGEABLE=NO. See the PAGEABLE parameter description for specification instructions.</p>
8	xxxx0834	<p><b>Equate Symbol:</b> IXLRSNCODEBADNONPGBLATTR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list is specified as being nonpageable (PAGEABLE=NO) but is either pageable or not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The type of storage in the buffer area corresponds to the value you specified on the PAGEABLE parameter. See the PAGEABLE parameter description for specification instructions.</li> <li>• The correct buffer address was used.</li> <li>• The buffer area(s) were not previously freed.</li> <li>• If BUFLIST was specified and your program is running in AR mode, ensure that: <ul style="list-style-type: none"> <li>– The BUFALET specification is correct.</li> <li>– You specified SYSSTATE ASCENV=AR before issuing the macro.</li> </ul> </li> <li>• If the caller is running in AR-mode and the BUFFER or BUFLIST parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> </ul>

Table 37. Return and Reason Codes for the IXLCACHE REQUEST=READ\_COSTATS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0835	<p><b>Equate Symbol:</b> IXLRNCODEBADDATAADDR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list, is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The correct buffer address was used for BUFFER or for a buffer within the BUFLIST.</li> <li>• The buffer area(s) were not previously freed.</li> <li>• The buffer area(s) were allocated in a storage key that matches the key specified using the BUFSTGKEY parameter, or, if BUFSTGKEY is omitted, the caller's PSW key.</li> <li>• If the caller is running in AR-mode and the BUFFER or BUFLIST parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If the caller is running in AR-mode, SYSSTATE ASCENV=AR must be specified before issuing this macro.</li> <li>• If BUFLIST was specified and your program is running in AR mode the BUFALET specification is correct.</li> </ul>
8	xxxx0836	<p><b>Equate Symbol:</b> IXLRNCODEBADREALADDR</p> <p><b>Meaning:</b> Program error. Real storage addresses were provided in a BUFLIST list, but one of the buffers is not addressable in central storage.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that BUFADDRTYPE was specified as you intended.</li> <li>• Ensure that the real buffer addresses specified by BUFLIST are valid.</li> </ul>
8	xxxx0838	<p><b>Equate Symbol:</b> IXLRNCODEBADANSAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by ANSAREA is not addressable.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the ANSAREA address.</li> <li>• ANSAREA must be addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are calling IXLCACHE while disabled, ANSAREA must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLCACHE in AR-mode and you specified the ANSAREA address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are calling IXLCACHE in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLCACHE macro.</li> </ul>

Table 37. Return and Reason Codes for the IXLCACHE REQUEST=READ\_COSTATS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0839	<p><b>Equate Symbol:</b> IXLRNCODEBADREQTOKENAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by REQTOKEN is not addressable.</p> <p><b>Action:</b> Ensure that the request token area specified by REQTOKEN is valid:</p> <ul style="list-style-type: none"> <li>• Verify the REQTOKEN address.</li> <li>• If you are calling IXLCACHE while disabled, REQTOKEN must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLCACHE in AR-mode and you specified the REQTOKEN address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are calling IXLCACHE in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLCACHE macro.</li> </ul>
8	xxxx083D	<p><b>Equate Symbol:</b> IXLRNCODEBADANSLEN</p> <p><b>Meaning:</b> Program error. The length of the answer area, as specified by ANSLEN, is not sufficient to contain answer area information. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Increase the size of the answer area provided for the request, and rerun your program. Depending on the macro version number, specify the length of the answer area as follows:</p> <ul style="list-style-type: none"> <li>• When the connection specified ASYNCXI=1 on the IXLCONN invocation and AXIOVERRIDE=0 was specified or defaulted to for the IXLCACHE request, the answer area length must be a minimum value of CAALEVEL2LEN</li> <li>• When the value of the macro version number (PLISTVER) is 4 or more, the minimum answer area length is CAALEVEL1LEN.</li> <li>• When the value of the macro version number (PLISTVER) is 0-3, the minimum answer area length is CAALEVEL0LEN.</li> </ul>
8	xxxx0851	<p><b>Equate Symbol:</b> IXLRNCODENOSUSPENDISABLE</p> <p><b>Meaning:</b> Program error. The request failed because MODE=SYNCSUSPEND was specified, but the caller is disabled and cannot be suspended.</p> <p><b>Action:</b> Either specify another MODE value or become enabled (release the CPU lock); then reissue the request.</p>
8	xxxx0864	<p><b>Equate Symbol:</b> IXLRNCODEBADBUFSIZE</p> <p><b>Meaning:</b> Program error. The size of the BUFLIST areas or BUFFER area is not large enough to contain the data being read. No data is returned.</p> <p><b>Action:</b> If more space is available, specify a larger buffer size and reissue the request. Consider using the BUFLIST parameter if BUFFER was specified.</p>

Table 37. Return and Reason Codes for the IXLCACHE REQUEST=READ\_COSTATS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0865	<p><b>Equate Symbol:</b> IXLRSNCODEBADBUFSPEC</p> <p><b>Meaning:</b> Program error. There is an error in the buffer specification.</p> <p><b>Action:</b> Check the following:</p> <ul style="list-style-type: none"> <li>• If BUFLIST was specified, check the requirements for BUFLIST and BUFNUM.</li> <li>• If BUFFER was specified, check the requirements for BUFFER and BUFSIZE.</li> <li>• Buffer pointer(s) in BUFLIST.</li> <li>• Buffer boundaries.</li> </ul>
8	xxxx0866	<p><b>Equate Symbol:</b> IXLRSNCODEBADBUFKEY</p> <p><b>Meaning:</b> Program error. The buffer storage key specified by BUFSTGKEY is incorrect or BUFSTGKEY was not specified and the PSW key does not match the key of the buffers.</p> <p>The data cannot be stored in the specified buffer area.</p> <p><b>Action:</b> Check the following:</p> <ul style="list-style-type: none"> <li>• Determine if the key of the storage being used for the buffers is different from the PSW key.</li> <li>• If BUFSTGKEY was specified, verify that the key was put in the correct bits (see the explanation of this parameter for more information).</li> <li>• If you are calling IXLCACHE in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLCACHE macro.</li> </ul>
8	xxxx0867	<p><b>Equate Symbol:</b> IXLRSNCODEBADBUFLIST</p> <p><b>Meaning:</b> Program error. The 128-byte storage area specified by BUFLIST is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The correct BUFLIST address was used.</li> <li>• The BUFLIST area was not previously freed.</li> <li>• If the caller is running in AR-mode and the BUFLIST parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If the caller is disabled, then the BUFLIST must reside in either nonpageable or disabled reference storage.</li> <li>• If your program is running in AR mode, ensure that you specified SYSSTATE ASCENV=AR before issuing the macro.</li> </ul>

Table 37. Return and Reason Codes for the IXLCACHE REQUEST=READ\_COSTATS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C06	<p><b>Equate Symbol:</b> IXLRNOCODENOCNN</p> <p><b>Meaning:</b> Environmental error. No connectivity to the cache structure exists. This may occur due to operator commands such as VARY PATH,OFFLINE or CONFIG CHP,OFFLINE or hardware errors such as facility or path failures. The CONTOKEN is invalidated. The request fails.</p> <p><b>Action:</b> Either disconnect from the structure (using IXLDISC) or initiate a rebuild of the structure, if allowed (using IXLREBLD).</p>
C	xxxx0C13	<p><b>Equate Symbol:</b> IXLRNOCODEREQPURGED</p> <p><b>Meaning:</b> Environmental error. The request was purged prior to completion of the request. Possible reasons include:</p> <ul style="list-style-type: none"> <li>• The connector failed.</li> <li>• The connector disconnected.</li> <li>• The requestor failed.</li> <li>• The request was purged by IXLPURGE.</li> <li>• Requests were purged when the connector provided an IXLEERSP response for the Rebuild Quiesce, Rebuild Stop, or Rebuild Cleanup event.</li> <li>• The connector invoked IXLREBLD REQUEST=COMPLETE.</li> <li>• The secondary address space was no longer valid.</li> </ul> <p><b>Action:</b> None if this is expected. Otherwise, determine why the connector failed.</p>
C	xxxx0C14	<p><b>Equate Symbol:</b> IXLRNOCODESTATUSUNKNOWN</p> <p><b>Meaning:</b> Environmental error. The IXLCACHE request has completed, but the final disposition of the request cannot be determined. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Verify the validity of your data by comparing the expected results with what is in the coupling facility.</p>
C	xxxx0C25	<p><b>Equate Symbol:</b> IXLRNOCODESTRFAILURE</p> <p><b>Meaning:</b> Environmental error. The cache structure failed prior to completion of the request.</p> <p><b>Action:</b> Attempt to rebuild the structure using IXLREBLD, or disconnect from the structure using IXLDISC.</p>
C	xxxx0CA0	<p><b>Equate Symbol:</b> IXLRNOCODEQUIESCEDSUSPENDFAIL</p> <p><b>Meaning:</b> Environmental error. The request is failed because the structure is quiesced for a system-managed process and SUSPEND=FAIL is specified on the IXLCONN.</p> <p><b>Action:</b> None, if this is expected.</p>

Table 37. Return and Reason Codes for the IXLCACHE REQUEST=READ\_COSTATS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxxFFFF	<b>Equate Symbol:</b> IXLRSNCODENOTAVAILABLE <b>Meaning:</b> Environmental error. XES functions are not available. The coupling facility hardware might not be present. <b>Action:</b> XES may not be used when XCF is running local mode. XES requires that the coupling facility be installed.
10	xxxx10xx	<b>Equate Symbol:</b> IXLRSNCODECOMPERROR <b>Meaning:</b> System error. XES processing failure. The state of the structure and the disposition of the request are unpredictable. The answer area (ANSAREA) fields do not contain valid information. <b>Action:</b> Save the reason code information, and contact the IBM support center.





## Chapter 40. IXLCACHE REQUEST=READ\_DATA

### Description

A READ\_DATA request allows you to read a data item from a data entry in the cache structure and optionally place it in the storage areas specified by BUFFER or BUFLIST and/or ADJAREA.

A data item you attempt to read might not reside in the cache structure at the time the read is requested. (For instance, only a directory entry exists, or neither a directory entry nor a data entry exists.) If this is the case, no data is read. However, you can use a READ\_DATA request to allocate a directory entry to the data item. To do this, use the ASSIGN=YES option. This is the method directory-only users employ to allocate a coupling facility cache structure directory entry for a data item.

The NAME parameter identifies the data item that will be, or has already been, introduced to the structure. Once introduced to the structure, NAME also references the structure resources (the directory entry and data entry) allocated for that data item.

With a structure allocated in a coupling facility of CFLEVEL=5 or higher, when issuing a READ\_DATA request you can choose whether to register interest in the data item. The default, if you do not specify an option, is to register interest in the data item.

- You can register or update your interest in the data item by specifying REGUSER=YES and then specifying an index into your local cache vector (VECTORINDEX) to be associated with the data item in the structure. This index identifies a vector entry that cache services uses to indicate both your interest in the data item and the validity of your locally cached copy of the data item. Having registered interest in a data item **means** that you have a valid copy of the data item in your local cache buffer. When another user updates the data item in the structure, cache services will update the associated vector entry, thereby deregistering your interest in the data item and invalidating your locally cached copy.

Cache services is responsible for deregistering your interest in a data item whenever another user updates the data item in the structure. You are responsible for maintaining the association between the entry in your local cache vector and the copy of the data item in your local cache buffer.

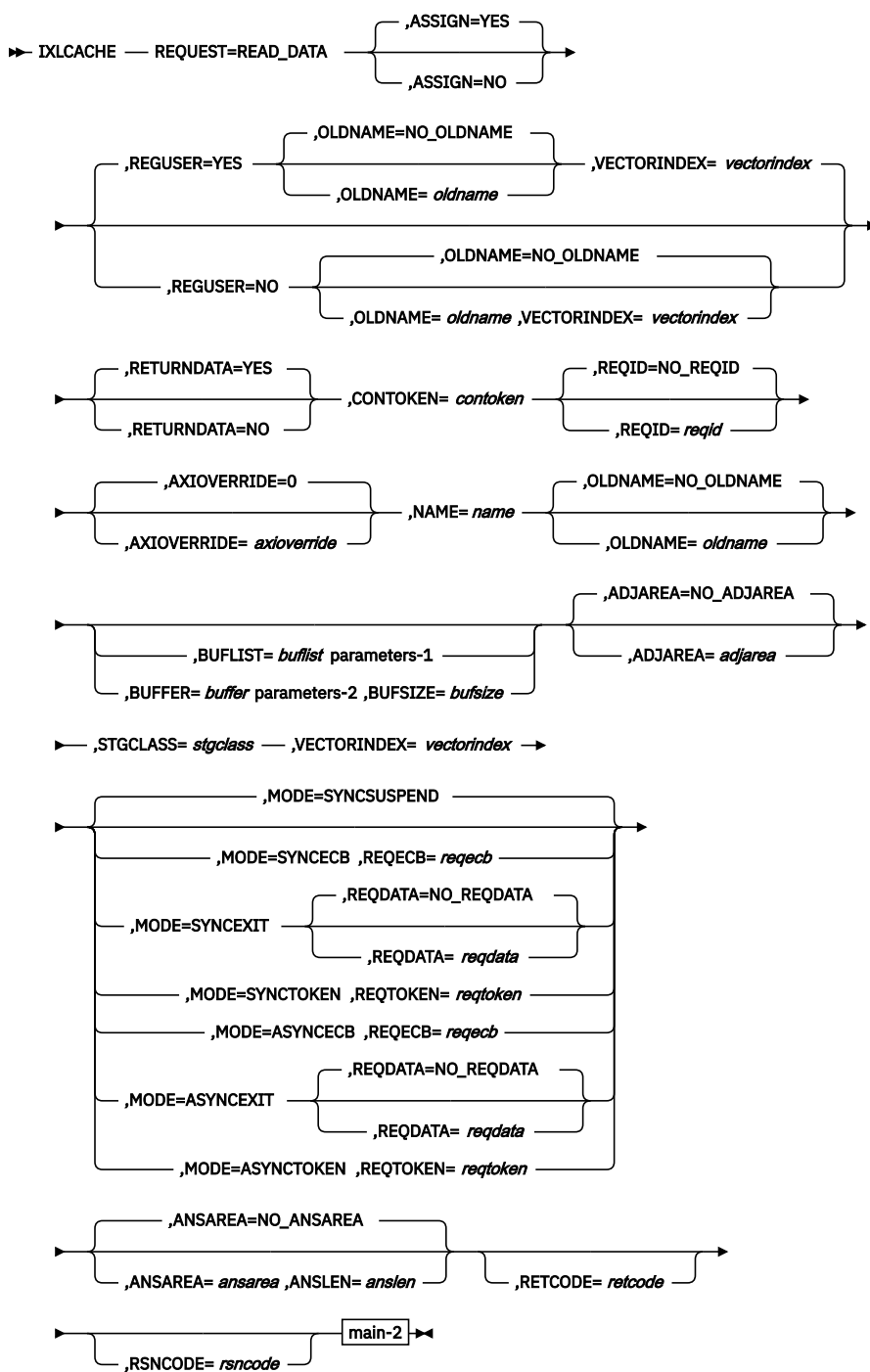
- You can specify that registration of your interest in the data item is not to be performed (REGUSER=NO). Be aware that entries in a data item for which no interest has been registered are higher-priority candidates for reclaim processing.

For cache structures allocated in a coupling facility with CFLEVEL=4 or higher, you can specify RETURNDATA=NO to suppress the read function when a REQUEST=READ\_DATA is issued. By specifying RETURNDATA=NO, the READ\_DATA request will register interest in the entry without returning the associated data. This is useful when data is cached for the entry but you do not want the data to be read. Note that if adjunct is supported by the structure (ADJUNCT=YES on the IXLCONN invocation), data is cached for the entry, and you have specified ADJAREA, then the adjunct will be returned. The CAAADJAREAVALID bit in the cache answer area is set to indicate whether adjunct data was returned.

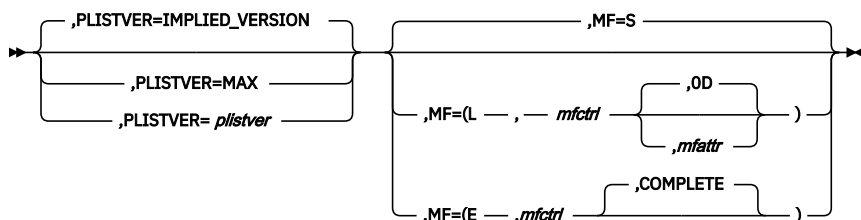
### Syntax Diagram

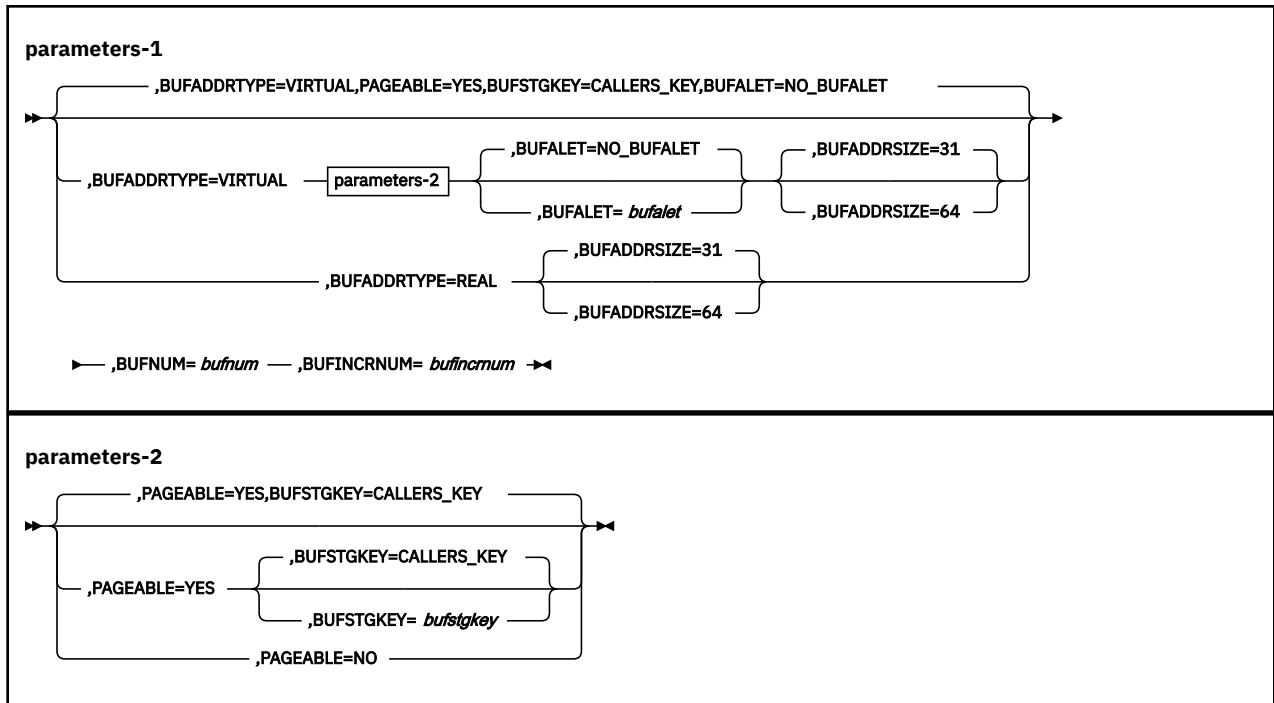
The syntax for the IXLCACHE REQUEST=READ\_DATA is as follows:

## main diagram



## main-2





**Note:** You must specify `ANSAREA=ansarea`, `ANSLEN=anslen` if you:

- Specify `MODE=SYNCTOKEN` or `MODE=ASYNCTOKEN`, or
- Specify `AXIOVERRIDE=0` (or default to 0) and specified `ASYNCXI=1` on the `IXLCONN` invocation when connecting to the cache structure.

## Parameter Descriptions

The parameter descriptions for `REQUEST=READ_DATA` are listed in alphabetical order. Default values are underlined:

### **REQUEST=READ\_DATA**

Use this input parameter to specify that the data item identified by `NAME` be read from the cache structure and stored in the areas specified by `BUFFER`, `BUFLIST` and/or `ADJAREA`.

Directory-only users can also use this parameter to define a new data item to the cache structure and register interest in that data item.

### **,ADJAREA=NO\_ADJAREA**

### **,ADJAREA=adjarea**

Use this output parameter to specify a storage area to contain the adjunct data that was read from one or more buffers designated entry.

If the structure supports adjunct data and `ADJAREA` is not specified, or if `ADJAREA=NO_ADJAREA` is specified, then no adjunct data is returned.

If the structure does not support adjunct data and `ADJAREA` is specified, then no adjunct data is returned, and the request will complete with a return code `X'4'` and a reason code of `IXLSNCOENOADJUNCTDATA`.

If the structure supports adjunct data and `ADJAREA` is specified, but there is no data and adjunct associated with the entry, then:

- For `READ_DATA` requests, the request completes with return code `X'4'` and reason code `IXLSNCOENOREADDATA`.
- For `CASTOUT_DATA` requests, the request completes with return code `X'8'` and reason code `IXLSNCOECOUNCHANGED`.

(Adjunct areas for a structure are established through the `IXLCONN` macro.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 64-byte virtual storage area that the adjunct data will be read in.

**,ANSAREA=NO ANSAREA**

**,ANSAREA=ansarea**

Use this output parameter to specify an answer area to contain information returned from the request. The format of the answer area is described by the IXLYCAA mapping macro. On a successful completion of the request, the following information is returned in the answer area:

- An indication of whether the entry has changed in the cache structure (field CAACHANGED).
- The parity of the returned data item (field CAAPARITY).
- The cast-out lock state (field CAACOLLOCKSTATE).
- The cast-out lock value (field CAACOLLOCKVAL).
- If the request replaced a previously specified local cache vector index for the entry, the replaced index is returned (fields CAAINVLCVI and CAAINVLCVINUM).
- If BUFFER or BUFLIST is specified, the number of elements in the retrieved entry is returned (field CAAELEMNUM).
- An adjunct area validity indicator (field CAAADJAREVALID).

The following additional fields are returned in the answer area when the structure is allocated in a coupling facility with CFLEVEL=4 or higher:

- A data-cached indicator (field CAADATAACACHED)
- If BUFFER or BUFLIST is not specified, the number of elements in the entry whether or not BUFFER or BUFLIST was specified (field CAAELEMNUM).

The following additional field is returned in the answer area when the connection specified ASYNXCI=1 on the IXLCONN invocation when connecting to the cache structure and the cache structure is allocated in a CFLEVEL=23 or higher coupling facility:

- An asynchronous cross-invalidation sequence number (CAAASYNXISEQNUM) if cross-invalidations were initiated asynchronously to the completion of the request.

See [Chapter 29, "IXLAXISN," on page 449](#) for a description of how to use the returned CAAASYNXISEQNUM to determine when cross-invalidates of local caches associated with the request have completed.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an area (with a length of ANSLEN) where the answer area information returned from the request will be stored.

**,ANSLEN=anslen**

Use this inppaameter to specify the size of the storage area specified by ANSAREA.

Use either CAALEVELOLEN, CAALEVEL1LEN or CAALEVEL2LEN of the IXLYCAA mapping macro to determine the minimum size of the answer area. The answer area length must be at least large enough to accommodate the level of the IXLYCAA mapping appropriate to the requested function.

- When the connection specified ASYNXCI=1 on the IXLCONN invocation and AXIOVERRIDE=0 was specified or defaulted to for the IXLCACHE request, the answer area length is a required parameter and must be a minimum value of CAALEVEL2LEN to contain a returned asynchronous cross-invalidation sequence number (CAAASYNXISEQNUM).
- When the value of PLISTVER is 4 or above, the minimum answer area length is CAALEVEL1LEN.
- When the value of PLISTVER is 0 - 3, the minimum answer area length is CAALEVELOLEN.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12 to ) of a 2-byte field that contains the length of the answer area (ANSAREA).

**,ASSIGN=YES**

**,ASSIGN=NO**

Use this input parameter to specify whether a directory entry in the cache structure will be allocated for (assigned to) the data item specified by NAME.

**YES**

- If a directory entry is not already assigned, this option requests that one be assigned for data item NAME. No data is read.
- If a directory entry is already assigned, the assignment remains. If there is a data entry, its contents are read.

**NO**

This option assumes a directory entry is already assigned to the data item.

- If a directory entry is not already assigned, a return code X'8', reason code IXLRSNCODENOENTRY is returned.
- If a directory entry is already assigned, the assignment remains. If there is a data entry, its contents are read.

Regardless of the option you choose, if a directory entry is assigned but there is no associated entry data, a return code IXLRETCODEWARNING, reason code IXLRSNCODENOREADDATA is returned.

**,AXIOVERRIDE=0****,AXIOVERRIDE=***axiooverride*

Use this input parameter to specify whether the asynchronous cross-invalidation control setting of IxlConnAsyncXiYes (1) for the connection identified by CONTOKEN should be overridden for this request. Valid values are 0 (IxlCacheAXiOverrideNo) or 1 (IxlCacheAXiOverrideYes).

The AXIOVERRIDE keyword is meaningful to processing only when the connection specified ASYNCXI=1 on the IXLCONN invocation when connecting to the cache structure and the cache structure is allocated in a CFLEVEL=23 or higher coupling facility.

A value of 0 (IxlCacheAXiOverrideNo) indicates that the asynchronous cross-invalidation control for the connection as specified on the IXLCONN invocation should be used for the request. Cross-invalidation against local caches for this request will preferentially be initiated asynchronously to the completion of the request when asynchronous cross-invalidations are supported by the coupling facility where the cache structure is allocated.

A value of 1 (IxlCacheAXiOverrideYes) indicates that the ASYNCXI specification of IxlConnAsyncXiYes (1) by the connector on the IXLCONN invocation should be overridden for this request only. Cross-invalidations generated by this request will be processed synchronously to the completion of the request.

Any value other than 0 or 1 for AXIOVERRIDE will have the same behavior as specifying a value of 0 (IxlCacheAXiOverrideNo).

If cross-invalidation against local caches for this request were initiated asynchronously to the completion of the request, an asynchronous cross-invalidation sequence number is returned in CAAASYNXISEQNUM of the cache answer area (ANSAREA).

The asynchronous cross-invalidation sequence number can be used on a subsequent invocation of IXLAXISN to ensure that the asynchronous cross-invalidations associated with this request have completed.

When cross-invalidation is initiated synchronously to the completion of the request or no cross-invalidation occurred for the request, no asynchronous cross-invalidation sequence number is returned.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12 2 to ) of a one-byte input field that contains the value indicating whether the asynchronous cross-invalidation control setting of the connector should be overridden

**,BUFADDRSIZE=31****,BUFADDRSIZE=64**

Use this input parameter to specify whether a 31-bit or a 64-bit address is specified by a BUFLIST entry.

**31**

The entry in BUFLIST is 31 bits in size.

**64**

The entry in BUFLIST is 64 bits in size.

**,BUFADDRTYPE=VIRTUAL****,BUFADDRTYPE=REAL**

Use this input parameter to specify whether the buffer addresses specified in the BUFLIST list are virtual storage or real storage addresses.

**VIRTUAL**

The buffer addresses are virtual storage addresses. The virtual storage can be pageable or nonpageable. See the PAGEABLE parameter for information about managing storage binds when specifying virtual storage addresses.

**REAL**

The buffer addresses are real storage addresses.

It is the caller's responsibility to manage the binds between the data buffer virtual storage and the real storage addresses provided. The caller must ensure that the data buffer virtual storage remains bound to the real storage addresses provided until the request completes.

**,BUFALET=NO\_BUFALET****,BUFALET=bufalet**

Use this input parameter to specify an access list entry token (ALET) to be used in referencing all of the buffers specified by BUFLIST.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 4-byte field that contains the ALET.

**,BUFFER=buffer**

Use this output parameter to specify a buffer area to hold the entry data that is read from the cache structure.

You can define the buffer size to be a total of up to 65536 bytes. Other requirements depend on the size you select:

- If you specify a buffer size of less than or equal to 4096 bytes, you must ensure that the buffer:
  - Is 256, 512, 1024, 2048, or 4096 bytes.
  - Starts on a 256-byte boundary.
  - Does not cross a 4096-byte boundary.
  - Does not start below storage address 512.
- If you specify a buffer size of greater than 4096 bytes, you must ensure that the buffer:
  - Is a multiple of 4096 bytes.
  - Is less than or equal to 65536 bytes.
  - Starts on a 4096-byte boundary.

See the BUFSIZE parameter description for defining the size of the buffer. See [z/OS MVS Programming: Sysplex Services Guide](#) for more information on buffers.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an area (with a length of BUFSIZE) to receive the entry requested from the cache structure.

**,BUFINCRNUM=bufincrnum**

Use this input parameter to specify the number of 256-byte segments comprising each buffer in the BUFLIST list.

Valid BUFINCRNUM values are 1, 2, 4, 8, or 16, which correspond to BUFLIST buffer sizes of 256, 512, 1024, 2048, and 4096 bytes respectively.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 1-byte field that contains 1, 2, 4, 8, or 16.

**,BUFLIST=buflist**

Use this output parameter to specify a list of buffers to hold the entry data that is read from the cache structure. BUFLIST specifies a 128-byte storage area that consists of a list of 0 to 16 buffer address elements.

The format of the list is a set of 8-byte elements. The BUFADDRSIZE keyword denotes whether four or eight bytes of the element are used.

- If BUFADDRSIZE=31 is specified, then the first four bytes of each element are reserved space and the last four bytes contain the address of the buffer.
- If BUFADDRSIZE=64 is specified, then the full eight bytes specify the address of the buffer.

**The BUFLIST buffers must:**

- Reside in the same address space or data space as defined by BUFALET.
- Be the same size: either 256, 512, 1024, 2048, or 4096 bytes as defined by BUFINCRNUM.
- Start on a 256-byte boundary and not cross a 4096-byte boundary.
- Not start below storage address 512.

**Note:** The buffers do not have to be contiguous in storage. XES treats BUFLIST buffers as a single buffer even if the buffers are not contiguous.

See the BUFNUM and BUFINCRNUM keyword descriptions for specifying the number and size of buffers.

See the BUFALET keyword for information on virtual buffers.

See [z/OS MVS Programming: Sysplex Services Guide](#) for more information on buffers.

**To Code:** Specify the RS-type name or address (using register 2 to 12) of a 128-byte field that contains a list of buffer addresses.

**,BUFNUM=bufnum**

Use this input parameter to specify the number of buffers in the BUFLIST list. Valid BUFNUM values are 0 - 16. A value of zero indicates that no data is to be read into the buffers.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 1-byte field that contains the number of buffers (0 - 16) in the list (BUFLIST).

**,BUFSIZE=bufsize**

Use this input parameter to specify the size of the BUFFER area. See the BUFFER parameter description for valid buffer sizes.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a fullword field that contains the size of the buffer (BUFFER) in bytes.

**,BUFSTGKEY=CALLERS\_KEY****,BUFSTGKEY=bufstgkey**

Use this input parameter to specify a storage key that you define and use when referencing the buffers specified by BUFLIST or the buffer that is specified by BUFFER.

If you do not specify BUFSTGKEY, or if you specify BUFSTGKEY=CALLERS\_KEY, all references to one or more buffers are performed by using the caller's PSW key.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of an 8-bit field that contains the storage key in the format B'kkkkxxxx', where kkkk is the key and xxxx is ignored.

**,CONTO=conken**

Use this input parameter to specify the connect token that was returned by the IXLCONN service in the IXLCONN answer area, which is mapped which is by IXLYCONA. The connect token uniquely identifies your connection to the cache structure, and must be specified on each IXLCACHE invocation.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field that contains the connect token.

**,MF=S**  
**,MF=(L,mfctrl)**  
**,MF=(L,mfctrl,mfattr)**  
**,MF=(L,mfctrl,0D)**  
**,MF=(M,mfctrl)**  
**,MF=(M,mfctrl,COMPLETE)**  
**,MF=(M,mfctrl,NOCHECK)**  
**,MF=(E,mfctrl)**  
**,MF=(E,mfctrl,COMPLETE)**  
**,MF=(E,mfctrl,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

**,mfctrl**

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

**,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

**,COMPLETE**

**,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

**COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if SMILE=*var* were an optional parameter and the default is SMILE=NO\_SMILE then it would not be documented. However, if the default was SMILE=:~), then it would be documented because a value would be the default.

**NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.



**,MODE=SYNCSUSPEND**  
**,MODE=SYNCECB**  
**,MODE=SYNCEXIT**  
**,MODE=SYNCTOKEN**  
**,MODE=ASYNCECB**  
**,MODE=ASYNCEXIT**  
**,MODE=ASYNCTOKEN**

Use this input parameter to specify:

- Whether the request is to be performed synchronously or asynchronously
- How you want to be notified of request completion if the request is processed asynchronously.

See *z/OS MVS Programming: Sysplex Services Guide* for more information on understanding synchronous and asynchronous cache operations.

#### **SYNCSUSPEND**

The request processes synchronously. If necessary, the request is suspended until it can complete synchronously. To use this option, your program must be enabled for I/O and external interrupts.

#### **SYNCECB**

The request processes synchronously if possible. If the request processes asynchronously, the ECB specified by REQECB is posted when the request completes.

#### **SYNCEXIT**

The request processes synchronously if possible. If the request processes asynchronously, your complete exit is given control when the request completes.

#### **SYNCTOKEN**

The request processes synchronously if possible. If the request processes asynchronously, an asynchronous request token is returned to the area specified by REQTOKEN. Use the returned request token on the IXLFCOMP macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=SYNCTOKEN is specified.

#### **ASYNCECB**

The request processes asynchronously. The ECB specified by REQECB is posted when the request completes.

#### **ASYNCEXIT**

The request processes asynchronously. Your complete exit is given control when the request completes.

#### **ASYNCTOKEN**

The request processes asynchronously. An asynchronous request token is returned to the area specified by REQTOKEN. Use the returned request token on the IXLFCOMP macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=ASYNCTOKEN is specified.

**,NAME=name**

Use this input parameter to specify the name of the data item to be read from the cache structure.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-byte field that contains the name of the data item.

**,OLDNAME=NO\_OLDNAME**

**,OLDNAME=oldname**

Use this input parameter to specify the name of the data item for which your interest should be unregistered.

- For structures allocated in a coupling facility of CFLEVEL=4 or lower, you must also specify the name of the data item for which interest is to be registered after the interest is unregistered in OLDNAME. Identify the data item to which interest is to be registered with NAME. The VECTORINDEX currently associated with the entry specified by OLDNAME will be reassigned to the name of the data item specified by NAME.

- For structures allocated in a coupling facility of CFLEVEL=5 or higher, you can specify that interest in the name of the data item specified by OLDNAME is to be unregistered without registering interest in the entry specified by NAME.

In either case, whenever deregistration of interest is requested, VECTORINDEX must be specified.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field that contains the name of the old data item that was associated with VECTORINDEX.

**,PAGEABLE=YES**

**,PAGEABLE=NO**

Use this input parameter to identify whether the storage areas specified by BUFFER or BUFLIST are in pageable or potentially pageable storage.

#### **YES**

Specify this option to indicate that the BUFFER or BUFLIST buffers reside in pageable virtual storage. XES performs the required page fixing to fix the buffers in real storage while the cache or list request transfers data to or from the coupling facility.

This includes storage obtained from pageable subpools, disabled reference (DREF) storage, and may include storage that has the potential to become pageable during the processing of a request. (An example is address space storage owned by any swappable address space, for which a PGSER FIX has been successfully processed, but for which the owning address space gets swapped during processing of a cache or list request.) This does not include implicitly non-pageable storage (for example, storage obtained from non-pageable subpools).

The system takes responsibility for managing binds to central storage for the duration of the cache or list request, regardless of what address space owns the storage or whether the storage-owning address space is swappable or nonswappable. The storage can be owned by any address space.

High shared virtual storage areas (above 2GB) may not be used.

#### **NO**

Specify this option to indicate that the BUFFER or BUFLIST buffers reside in non-pageable virtual storage. XES does not page fix the buffers in real storage.

This includes implicitly non-pageable storage areas (for example, storage obtained from non-pageable subpools), and may include storage that has the potential to become pageable during the processing of a request (An example is address space storage owned by any swappable address space, for which a PGSER FIX has been successfully processed, but for which the owning address space gets swapped-out during processing of a cache or list request.)

The system takes responsibility for managing binds to central storage for the duration of the cache or list request, if and only if the non-pageable storage is owned by either the requester's address space or the connector's address space. If the storage is owned by any other address space, then the invoker is responsible for ensuring that the virtual storage remains non-pageable for the duration of the request (including the case in which the storage is owned by a swappable address space that is swapped during processing of an IXLCACHE or IXLLIST request). Subject to this consideration, the storage can be owned by any address space. See *z/OS MVS Programming: Sysplex Services Guide*.

**,PLISTVER=IMPLIED\_VERSION**

**,PLISTVER=MAX**

**,PLISTVER=plistver**

Use this input parameter to specify the version of the macro. See [“Understanding IXLCACHE Version Support” on page 461](#) for a description of the options available with PLISTVER.

**,REGUSER=YES**

**,REGUSER=NO**

Use this input parameter to specify whether the request should register (or update) interest in the data item to be cast-out.

**YES**

Connection interest in the data item will be registered for the connection specified by CONTOKEN. If interest is already registered, the vector index specified by VECTORINDEX replaces any previously specified index for the data item.

Specify OLDNAME to deregister any outstanding interest for the specified vector index for a different entry.

**NO**

Connection interest in the data item will not be registered.

For structures allocated in a coupling facility with CFLEVEL=5 or higher, OLDNAME may be specified to deregister any outstanding interest for the specified local cache vector index for a different entry. If OLDNAME is specified, then VECTORINDEX is required.

**,REQDATA=NO REQDATA****,REQDATA=reqdata**

Use this input parameter with MODE=SYNCEXIT or MODE=ASYNCEXIT to pass any data you choose to the complete exit. The exit will get control only if the request is processed asynchronously.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the data to be passed to the complete exit.

**,REQECB=reqecb**

Use this output parameter with either MODE=SYNCECB or MODE=ASYNCECB to specify the address of an ECB, which is to be posted when the request completes if the request was processed asynchronously.

Before coding REQECB, you must ensure that:

- You initialize the ECB before you issue the request.
- The ECB resides in either common storage or the home address space where IXLCONN was issued.
- Any tasks that wait for the ECB to be posted reside in the home address space where IXLCONN was issued.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that contains the address of the ECB to be posted when the request completes. The ECB must be aligned on a fullword boundary.

**,REQID=NO REQID****,REQID=reqid**

Use this input parameter to specify a user-defined request identifier to be associated with the request. You can specify this request identifier on the IXLPURGE macro to cancel a request that has not yet been processed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the user-defined request identifier.

**,REQTOKEN=reqtoken**

Use this output parameter with either MODE=SYNCTOKEN or MODE=ASYNCTOKEN to specify the address of a storage area to receive the request token that is returned when the request will be processed asynchronously. This token, which uniquely identifies the request, must be used as input to the IXLFCOMP macro, which you use to determine if the request has completed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field where the system will put the request token.

**,RETCODE=retcode**

Use this output parameter to specify a field to contain the return code. (The return code is also returned in register 15.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the return code when the request has completed.

**,RETURNDATA=YES**

**,RETURNDATA=NO**

Use this input parameter to specify whether the data, if any exists for the data entry specified by NAME, is to be read. This option is valid only for cache structures allocated in a coupling facility with CFLEVEL=4 or higher. If ADJAREA is specified and adjunct data is supported by the structure, then the CAAADJAREAAVALID bit is set in the answer area:

- CAAADJAREAAVALID=B'1' indicates that the system returned adjunct data in ADJAREA.
- CAAADJAREAAVALID=B'0' indicates that data was not cached for the data entry specified by NAME or the structure did not support adjunct data.

**RETURNDATA=YES**

Specifies that the data is to be read into the storage area specified by either BUFFER or BUFLIST and/or ADJAREA.

**RETURNDATA=NO**

Specifies that for structures allocated in a coupling facility with CFLEVEL=4 or higher, the data entry read function will be suppressed. Adjunct data, if it exists and ADJAREA is specified, will be returned.

If RETURNDATA=NO is specified for a structure allocated in a coupling facility with CFLEVEL=3 or lower, this parameter is ignored and the data, if any exists for the specified NAME, is returned.

**,RSNCODE=rsncode**

Use this output parameter to specify a field to contain the reason code returned, if applicable. (The reason code is also returned in register 0.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the reason code (if any) when the request has completed.

**,STGCLASS=stgclass**

Use this input parameter to specify a storage class that the entry to be read is to be assigned to. Any previous assignment is updated to the new specification.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 1-byte field that contains the storage class.

**,VECTORINDEX=vectorindex**

Use this input parameter to specify an index into your local cache vector to be associated with the data item specified by NAME, to be disassociated with the data item specified by OLDNAME, or both. The vector index identifies a vector entry that cache services will use to indicate both your interest in the data item and the validity of the copy of the data item in your local cache buffer.

If the vector index you specify is already associated with another data item, you must disassociate the vector index from the old name by specifying OLDNAME before the vector index can be associated with the data item specified by NAME.

VECTORINDEX is required when REGUSER=YES is specified or defaulted, or whenever OLDNAME is specified. VECTORINDEX is not required when REGUSER=NO is specified without OLDNAME.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 4-byte field that contains the vector index for NAME or OLDNAME.

## ABEND Codes

Abend X'026' (See [z/OS MVS Programming: Sysplex Services Guide](#) for more information on this abend.)

## Return and Reason Codes

When the system returns control to your program:

- GPR 15 (and RETCODE, if specified) contains a return code.
- If the return code is not zero, GPR 0 (and RSNCODE, if specified) contains a reason code.

**Note:** The return and reason codes will also be put into the answer area (mapped by IXLYCAA), if there is one, when the request completes.

The IXLYCON macro provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

<b>0</b>	IXLRETCODEOK
<b>4</b>	IXLRETCODEWARNING
<b>8</b>	IXLRETCODEPARMERROR
<b>C</b>	IXLRETCODEENVERROR
<b>10</b>	IXLRETCODECOMPERROR

The following table contains hexadecimal return and reason codes, the equate symbols associated with each reason code, and the meaning and suggested action for each return and reason code.

Table 38. Return and Reason Codes for the IXLCACHE REQUEST=READ_DATA Macro		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0	None.	<p><b>Meaning:</b> If you specified a MODE value of ASYNCECB, ASYNCEXIT, or ASYNCTOKEN, the request has been successfully initiated. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed. For any other MODE value, the request has successfully completed.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=ASYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=ASYNCEXIT, the connection's complete exit will be given control when the request has completed.</li> <li>• If you specified MODE=ASYNCTOKEN, use REQTOKEN when calling IXLFCOMP to determine when the request has completed.</li> </ul>

Table 38. Return and Reason Codes for the IXLCACHE REQUEST=READ\_DATA Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	xxxx0402	<p><b>Equate Symbol:</b> IXLRSNCODEASYNCH</p> <p><b>Meaning:</b> The request could not be processed synchronously. It will be processed asynchronously. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed.</p> <ul style="list-style-type: none"> <li>• If you specified MODE=SYNCECB, the ECB specified in REQECB will be posted when the request has finished.</li> <li>• If you specified MODE=SYNCEXIT, the connection's complete exit will be given control when the request has finished.</li> <li>• If you specified MODE=SYNCTOKEN, the REQTOKEN contains an asynchronous request token that may be used on the IXLFCOMP macro to determine when the request has completed.</li> </ul> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=SYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=SYNCEXIT, the connection's completion exit will be given control when the request has completed.</li> <li>• If you specified MODE=SYNCTOKEN, use REQTOKEN when calling IXLFCOMP to determine when the request has completed.</li> </ul>

Table 38. Return and Reason Codes for the IXLCACHE REQUEST=READ\_DATA Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	xxxx040A	<p><b>Equate Symbol:</b> IXLRSCODENOREADDATA</p> <p><b>Meaning:</b> The request registered interest in the named data item, but either there is no entry data in the cache structure or the read function was suppressed on the request. Therefore, no data is returned in BUFLIST or BUFFER. For structures allocated in a coupling facility with CFLEVEL=4 or higher, if the structure entry contained data and adjunct and ADJAREA was specified, then the adjunct is returned. The CAAADJAREAVALID bit in the answer area is set to B'1' if the adjunct is returned or B'0' if adjunct does not exist for the entry.</p> <p>The following answer area fields are filled in:</p> <ul style="list-style-type: none"> <li>• CAAADJAREAVALID</li> <li>• CAAINVLCVI</li> <li>• CAAINVLCVINUM</li> </ul> <p>For CFLEVEL=4 and higher coupling facilities, the following additional answer area fields are returned:</p> <ul style="list-style-type: none"> <li>• CAACHANGED</li> <li>• CAACOLLOCKSTATE</li> <li>• CAACOLLOCKVAL</li> <li>• CAADATAACACHED</li> <li>• CAAELEMNUM</li> <li>• CAAPARITY</li> </ul> <p>For CFLEVEL=5 and higher coupling facilities, the following additional answer area field is returned:</p> <ul style="list-style-type: none"> <li>• CAAVERSION</li> </ul> <p><b>Action:</b> No action is necessary. If this return code and reason code are unexpected, you may want to:</p> <ul style="list-style-type: none"> <li>• Verify the NAME of the data item.</li> <li>• Verify that the CONTOKEN is for the correct structure.</li> </ul>

Table 38. Return and Reason Codes for the IXLCACHE REQUEST=READ\_DATA Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	xxxx040C	<p><b>Equate Symbol:</b> IXLRSCODENOADJUNCTDATA</p> <p><b>Meaning:</b> The request specified that adjunct data was to be read, but the structure does not support adjunct areas. No adjunct data was retrieved; however, entry data was returned in BUFLIST or BUFFER if requested.</p> <p>The following fields are returned in the answer area:</p> <ul style="list-style-type: none"> <li>• CAACHANGED</li> <li>• CAACOLCKSTATE</li> <li>• CAACOLCKVAL</li> <li>• CAAELEMNUM</li> <li>• CAAINVLCVI</li> <li>• CAAINVLCVINUM</li> <li>• CAAPARITY</li> </ul> <p>For CFLEVEL=4 and higher coupling facilities, the following additional answer area field is returned:</p> <ul style="list-style-type: none"> <li>• CAADATACACHED</li> </ul> <p>For CFLEVEL=5 and higher coupling facilities, the following additional answer area field is returned:</p> <ul style="list-style-type: none"> <li>• CAAVERSION</li> </ul> <p><b>Action:</b> No action is necessary. However, if you expected this structure to have adjunct data, you may want to:</p> <ul style="list-style-type: none"> <li>• Verify the CONTOKEN is from the correct invocation of IXLCONN.</li> <li>• Check the IXLCONN macro that defined the structure to be sure adjunct data was specified.</li> </ul>



Table 38. Return and Reason Codes for the IXLCACHE REQUEST=READ\_DATA Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	xxxx040D	<p><b>Equate Symbol:</b> IXLRSNCODEBADREADADJDATA</p> <p><b>Meaning:</b> Program error. The request specified that adjunct data was to be read, but the specified storage area for adjunct data (ADJAREA) is not addressable. The adjunct data was not retrieved; however, the entry data was returned in BUFLIST or BUFFER, if requested.</p> <p>The following fields are returned in the answer area:</p> <ul style="list-style-type: none"> <li>• CAACHANGED</li> <li>• CAACLOCKSTATE</li> <li>• CAACLOCKVAL</li> <li>• CAAELEMNUM</li> <li>• CAAINVLCVI</li> <li>• CAAINVLCVINUM</li> <li>• CAAPARITY</li> </ul> <p>For CFLEVEL=4 and higher coupling facilities, the following additional answer area field is returned:</p> <ul style="list-style-type: none"> <li>• CAADATACACHED</li> </ul> <p>For CFLEVEL=5 and higher coupling facilities, the following additional answer area field is returned:</p> <ul style="list-style-type: none"> <li>• CAAVERSION</li> </ul> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the ADJAREA address.</li> <li>• ADJAREA must be addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are calling IXLCACHE while disabled, ADJAREA must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLCACHE in AR-mode and you specified the ADJAREA address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are calling IXLCACHE in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLCACHE macro.</li> </ul> <p>Correct the address specified by ADJAREA and rerun the request asking for adjunct data only.</p>

Table 38. Return and Reason Codes for the IXLCACHE REQUEST=READ\_DATA Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0801	<p><b>Equate Symbol:</b> IXLRSNCODEBADPARMLIST</p> <p><b>Meaning:</b> The parameter list (mfctrl) for this request is not addressable. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the parameter list address.</li> <li>• The parameter list must be addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are issuing this macro while disabled, the parameter list must reside in either page-fixed or DREF storage.</li> <li>• If you are issuing this macro in AR-mode and you specified the parameter list address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are issuing this macro in AR-mode, SYSSTATE ASCENV=AR must be issued before this macro.</li> </ul>
8	xxxx0804	<p><b>Equate Symbol:</b> IXLRSNCODEBADVERSION#</p> <p><b>Meaning:</b> The version number in the macro parameter list is not compatible with the level of XES currently being used. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that your program did not overlay the parameter list storage.</li> <li>• Verify that your program was assembled with the correct macro library for the release of MVS on which your program is running.</li> <li>• Verify that your program is running on an MVS system that supports the version of the macro you are using.</li> </ul>

Table 38. Return and Reason Codes for the IXLCACHE REQUEST=READ\_DATA Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx080A	<p><b>Equate Symbol:</b> IXLRNCODEBADCONTOKEN</p> <p><b>Meaning:</b> Program error. The specified connect token (CONTOKEN) was not valid for one of the following reasons:</p> <ol style="list-style-type: none"> <li>1. The user with the connection identifier represented by the token has disconnected from the structure.</li> <li>2. The connector's task (the task that issued IXLCONN) ended.</li> <li>3. The specified token is not the token that was returned from IXLCONN.</li> <li>4. The request was issued from an address space other than the address space in which IXLCONN was issued.</li> <li>5. The connect token was invalidated during rebuild.</li> <li>6. The connect token was invalidated by XES.</li> </ol> <p><b>Note:</b> The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Verify that the CONTOKEN value specified was valid and for the correct structure. The numbers on the following actions correspond to the numbers above.</p> <ol style="list-style-type: none"> <li>1. Discontinue use. Perform recovery and cleanup for the structure and check protocol for use of structure to determine why you did not know of the disconnection.</li> <li>2. Discontinue use. Perform recovery and cleanup for the structure and check protocol for use of structure to determine why you did not know that the task ended.</li> <li>3. Issue IXLCONN to connect to the structure. The contoken is returned in the CONACONTOKEN field of the answer area specified on the IXLCONN request.</li> <li>4. Issue your request from the same address space the IXLCONN was issued.</li> <li>5. Participate in the rebuild. When it is complete, try again.</li> <li>6. Discontinue use. Perform recovery and cleanup for the structure and check protocol for use of structure.</li> </ol> <p>You may want to issue IXCQUERY to get more information about the structure.</p>
8	xxxx0819	<p><b>Equate Symbol:</b> IXLRNCODEBADVECTOROP</p> <p><b>Meaning:</b> The VECTORINDEX specified is not valid. Request processing was suppressed.</p> <p><b>Action:</b> Specify a vector index that is within the current range of the number of vector entries for the local vector requested for this structure. The number of vector entries is determined by the VECTORLEN parameter specified on the IXLCONN macro and may be changed by issuing the IXLVECTR macro.</p>

Table 38. Return and Reason Codes for the IXLCACHE REQUEST=READ\_DATA Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0824	<p><b>Equate Symbol:</b> IXLRSNCODEWRONGSTRTYPE</p> <p><b>Meaning:</b> The connect token specified by CONTOKEN is not to a cache structure.</p> <p><b>Action:</b> Verify the connect token for this cache structure.</p> <p><b>Note:</b> The connect token can be found in the IXLCONN answer area. Verify the value specified on the TYPE= parameter of the IXLCONN request for this structure.</p>
8	xxxx0825	<p><b>Equate Symbol:</b> IXLRSNCODENOENTRY</p> <p><b>Meaning:</b> Program error. The request with ASSIGN=NO specified failed because the entry designated by NAME is not in the cache structure.</p> <p><b>Action:</b> No action is necessary. However, if this return code and reason code are unexpected, try the following:</p> <ul style="list-style-type: none"> <li>• Verify that NAME is uncorrupted.</li> <li>• Verify that CONTOKEN is for the correct structure.</li> </ul> <p>You might want to issue the request with ASSIGN=YES to allocate a directory entry for this NAME.</p>
8	xxxx082D	<p><b>Equate Symbol:</b> IXLRSNCODEBADSTGCLASS</p> <p><b>Meaning:</b> Program error. The storage class specified by STGCLASS exceeds the maximum number of storage classes defined for the cache structure.</p> <p><b>Action:</b> The number of storage classes allowed for a structure are defined on the IXLCONN macro by the NUMSTGCLASS parameter. Correct the STGCLASS parameter to specify a valid storage class.</p>
8	xxxx0833	<p><b>Equate Symbol:</b> IXLRSNCODEBADPGBLATTR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list is specified as being pageable (PAGEABLE=YES) but is not.</p> <p><b>Action:</b> Change the buffer area(s) to pageable storage, or specify PAGEABLE=NO. See the PAGEABLE parameter description for specification instructions.</p>

Table 38. Return and Reason Codes for the IXLCACHE REQUEST=READ\_DATA Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0834	<p><b>Equate Symbol:</b> IXLRNCODEBADNONPGBLATTR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list is specified as being nonpageable (PAGEABLE=NO) but is either pageable or not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The type of storage in the buffer area corresponds to the value you specified on the PAGEABLE parameter. See the PAGEABLE parameter description for specification instructions.</li> <li>• The correct buffer address was used.</li> <li>• The buffer area(s) were not previously freed.</li> <li>• If BUFLIST was specified and your program is running in AR mode, ensure that: <ul style="list-style-type: none"> <li>– The BUFALET specification is correct.</li> <li>– You specified SYSSTATE ASCENV=AR before issuing the macro.</li> </ul> </li> <li>• If the caller is running in AR-mode and the BUFFER or BUFLIST parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> </ul>
8	xxxx0835	<p><b>Equate Symbol:</b> IXLRNCODEBADDATAADDR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list, is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The correct buffer address was used for BUFFER or for a buffer within the BUFLIST.</li> <li>• The buffer area(s) were not previously freed.</li> <li>• The buffer area(s) were allocated in a storage key that matches the key specified using the BUFSTGKEY parameter, or, if BUFSTGKEY is omitted, the caller's PSW key.</li> <li>• If the caller is running in AR-mode and the BUFFER or BUFLIST parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If the caller is running in AR-mode, SYSSTATE ASCENV=AR must be specified before issuing this macro.</li> <li>• If BUFLIST was specified and your program is running in AR mode the BUFALET specification is correct.</li> </ul>

Table 38. Return and Reason Codes for the IXLCACHE REQUEST=READ\_DATA Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0836	<p><b>Equate Symbol:</b> IXLRNCODEBADREALADDR</p> <p><b>Meaning:</b> Program error. Real storage addresses were provided in a BUFLIST list, but one of the buffers is not addressable in central storage.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that BUFADDRTYPE was specified as you intended.</li> <li>• Ensure that the real buffer addresses specified by BUFLIST are valid.</li> </ul>
8	xxxx0838	<p><b>Equate Symbol:</b> IXLRNCODEBADANSAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by ANSAREA is not addressable.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the ANSAREA address.</li> <li>• ANSAREA must be addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are calling IXLCACHE while disabled, ANSAREA must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLCACHE in AR-mode and you specified the ANSAREA address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are calling IXLCACHE in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLCACHE macro.</li> </ul>
8	xxxx0839	<p><b>Equate Symbol:</b> IXLRNCODEBADREQTOKENAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by REQTOKEN is not addressable.</p> <p><b>Action:</b> Ensure that the request token area specified by REQTOKEN is valid:</p> <ul style="list-style-type: none"> <li>• Verify the REQTOKEN address.</li> <li>• If you are calling IXLCACHE while disabled, REQTOKEN must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLCACHE in AR-mode and you specified the REQTOKEN address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are calling IXLCACHE in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLCACHE macro.</li> </ul>

Table 38. Return and Reason Codes for the IXLCACHE REQUEST=READ\_DATA Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx083D	<p><b>Equate Symbol:</b> IXLRSNCODEBADANSLEN</p> <p><b>Meaning:</b> Program error. The length of the answer area, as specified by ANSLEN, is not sufficient to contain answer area information. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Increase the size of the answer area provided for the request, and rerun your program. Depending on the macro version number, specify the length of the answer area as follows:</p> <ul style="list-style-type: none"> <li>• When the connection specified ASYNCXI=1 on the IXLCONN invocation and AXIOVERRIDE=0 was specified or defaulted to for the IXLCACHE request, the answer area length must be a minimum value of CAALEVEL2LEN</li> <li>• When the value of the macro version number (PLISTVER) is 4 or more, the minimum answer area length is CAALEVEL1LEN.</li> <li>• When the value of the macro version number (PLISTVER) is 0-3, the minimum answer area length is CAALEVEL0LEN.</li> </ul>
8	xxxx0851	<p><b>Equate Symbol:</b> IXLRSNCODENOSUSPENDISABLE</p> <p><b>Meaning:</b> Program error. The request failed because MODE=SYNCSUSPEND was specified, but the caller is disabled and cannot be suspended.</p> <p><b>Action:</b> Either specify another MODE value or become enabled (release the CPU lock); then reissue the request.</p>
8	xxxx0864	<p><b>Equate Symbol:</b> IXLRSNCODEBADBUFSIZE</p> <p><b>Meaning:</b> Program Error. The size of the BUFLIST areas or BUFFER area is not large enough to contain the data being read. The number of elements in the retrieved entry is returned in the answer area in the field CAAELEMNUM.</p> <p><b>Action:</b> If more space is available, specify a larger buffer size and reissue the request.</p>
8	xxxx0865	<p><b>Equate Symbol:</b> IXLRSNCODEBADBUFSPEC</p> <p><b>Meaning:</b> Program error. There is an error in the buffer specification.</p> <p><b>Action:</b> Check the following:</p> <ul style="list-style-type: none"> <li>• If BUFLIST was specified, check the requirements for BUFLIST, BUFNUM, and BUFINCRNUM.</li> <li>• If BUFFER was specified, check the requirements for BUFFER and BUFSIZE.</li> <li>• Buffer pointer(s) in BUFLIST.</li> <li>• Buffer boundaries.</li> </ul>

Table 38. Return and Reason Codes for the IXLCACHE REQUEST=READ\_DATA Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0866	<p><b>Equate Symbol:</b> IXLRNCODEBADBUFKEY</p> <p><b>Meaning:</b> Program error. The buffer storage key specified by BUFSTGKEY is incorrect or BUFSTGKEY was not specified and the PSW key does not match the key of the buffers.</p> <p>The data cannot be stored in the specified buffer area.</p> <p><b>Action:</b> Check the following:</p> <ul style="list-style-type: none"> <li>• Determine if the key of the storage being used for the buffers is different from the PSW key.</li> <li>• If BUFSTGKEY was specified, verify that the key was put in the correct bits (see the explanation of this parameter for more information).</li> <li>• If you are calling IXLCACHE in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLCACHE macro.</li> </ul>
8	xxxx0867	<p><b>Equate Symbol:</b> IXLRNCODEBADBUFLIST</p> <p><b>Meaning:</b> Program error. The 128-byte storage area specified by BUFLIST is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The correct BUFLIST address was used.</li> <li>• The BUFLIST area was not previously freed.</li> <li>• If the caller is running in AR-mode and the BUFLIST parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If the caller is disabled, then the BUFLIST must reside in either nonpageable or disabled reference storage.</li> <li>• If your program is running in AR mode, ensure that you specified SYSSTATE ASCENV=AR before issuing the macro.</li> </ul>
C	xxxx0C06	<p><b>Equate Symbol:</b> IXLRNCODENOCONN</p> <p><b>Meaning:</b> Environmental error. No connectivity to the cache structure exists. This may occur due to operator commands such as VARY PATH,OFFLINE or CONFIG CHP,OFFLINE or hardware errors such as facility or path failures. The CONTOKEN is invalidated. The request fails.</p> <p><b>Action:</b> Either disconnect from the structure (using IXLDISC) or initiate a rebuild of the structure, if allowed (using IXLREBLD).</p>



Table 38. Return and Reason Codes for the IXLCACHE REQUEST=READ\_DATA Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx08B9	<p><b>Equate Symbol:</b> IXLRNOCODENOANSAREA</p> <p><b>Meaning:</b> An answer area was not specified when one is required. The requested service determined that conditions exist that require an ANSAREA to complete the request.</p> <p><b>Action:</b> Provide an answer area (ANSAREA) and answer area length (ANSLEN) on the IXLCACHE macro invocation for the request. ANSAREA is required when the connection specified ASYNCCI=1 on the IXLCONN invocation when connecting to the cache structure, AXIOVERRIDE=0 was specified or defaulted to for the IXLCACHE request and the request is one of the following:</p> <ul style="list-style-type: none"> <li>• CROSS_INVALID</li> <li>• CROSS_INVALLIST</li> <li>• DELETE_NAME</li> <li>• DELETE_NAMELIST</li> <li>• READ_DATA</li> <li>• REG_NAMELIST</li> <li>• WRITE_DATA</li> <li>• WRITE_DATA LIST</li> </ul>
C	xxxx0C13	<p><b>Equate Symbol:</b> IXLRNOCODEREQPURGED</p> <p><b>Meaning:</b> Environmental error. The request was purged prior to completion of the request. Possible reasons include:</p> <ul style="list-style-type: none"> <li>• The connector failed.</li> <li>• The connector disconnected.</li> <li>• The requestor failed.</li> <li>• The request was purged by IXLPURGE.</li> <li>• Requests were purged when the connector provided an IXLEERSP response for the Rebuild Quiesce, Rebuild Stop, or Rebuild Cleanup event.</li> <li>• The connector invoked IXLREBLD REQUEST=COMPLETE.</li> <li>• The secondary address space was no longer valid.</li> </ul> <p><b>Action:</b> None if this is expected. Otherwise, determine why the connector failed.</p>
C	xxxx0C14	<p><b>Equate Symbol:</b> IXLRNOCODESTATUSUNKNOWN</p> <p><b>Meaning:</b> Environmental error. The IXLCACHE request has completed, but the final disposition of the request cannot be determined. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Verify the validity of your data by comparing the expected results with what is in the coupling facility.</p>

Table 38. Return and Reason Codes for the IXLCACHE REQUEST=READ\_DATA Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C17	<p><b>Equate Symbol:</b> IXLRNCODESTRFULL</p> <p><b>Meaning:</b> Environmental error. Allocation of a directory entry and/or data entry was necessary, but was unavailable or could not be reclaimed. The answer area field, CAASTGCLFULL, contains the storage class from which the reclaiming operation failed.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Determine if any data items may be cast-out to make room for this item.</li> <li>• Check your usage of storage classes to see if some data items can be moved to a different storage class (preferably with a lower priority) so some entries in the structure can be freed.</li> <li>• Determine if a rebuild of the structure is necessary to make room for more data entries/items.</li> </ul>
C	xxxx0C25	<p><b>Equate Symbol:</b> IXLRNCODESTRFAILURE</p> <p><b>Meaning:</b> Environmental error. The cache structure failed prior to completion of the request.</p> <p><b>Action:</b> Attempt to rebuild the structure using IXLREBLD, or disconnect from the structure using IXLDISC.</p>
C	xxxx0CA0	<p><b>Equate Symbol:</b> IXLRNCODEQUIESCEDSUSPENDFAIL</p> <p><b>Meaning:</b> Environmental error. The request is failed because the structure is quiesced for a system-managed process and SUSPEND=FAIL is specified on the IXLCONN.</p> <p><b>Action:</b> None, if this is expected.</p>
C	xxxxFFFF	<p><b>Equate Symbol:</b> IXLRNCODENOTAVAILABLE</p> <p><b>Meaning:</b> Environmental error. XES functions are not available. The coupling facility hardware might not be present.</p> <p><b>Action:</b> XES may not be used when XCF is running local mode. XES requires that the coupling facility be installed.</p>
10	xxxx10xx	<p><b>Equate Symbol:</b> IXLRNCODECOMPERROR</p> <p><b>Meaning:</b> System error. XES processing failure. The state of the structure and the disposition of the request are unpredictable. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Save the reason code information, and contact the IBM support center.</p>

---

## Chapter 41. IXLCACHE REQUEST=READ\_DIRINFO

### Description

---

A READ\_DIRINFO request allows you to retrieve directory information from the structure and store it in the storage areas specified by BUFLIST or BUFFER. You can specify that all directory entries be processed (CRITERIA=ALL) or only those directory entries that contain changed or locked-for-cast-out data (CRITERIA=CHANGED). You can further filter the selection of entries for processing by specifying NAME and optionally NAMEMASK.

You can request that all directory entry information be returned for the specified data items (DIRINFOFMT=DIRENTRYLIST) or only a subset of the information be returned (DIRINFOFMT=NAMELIST).

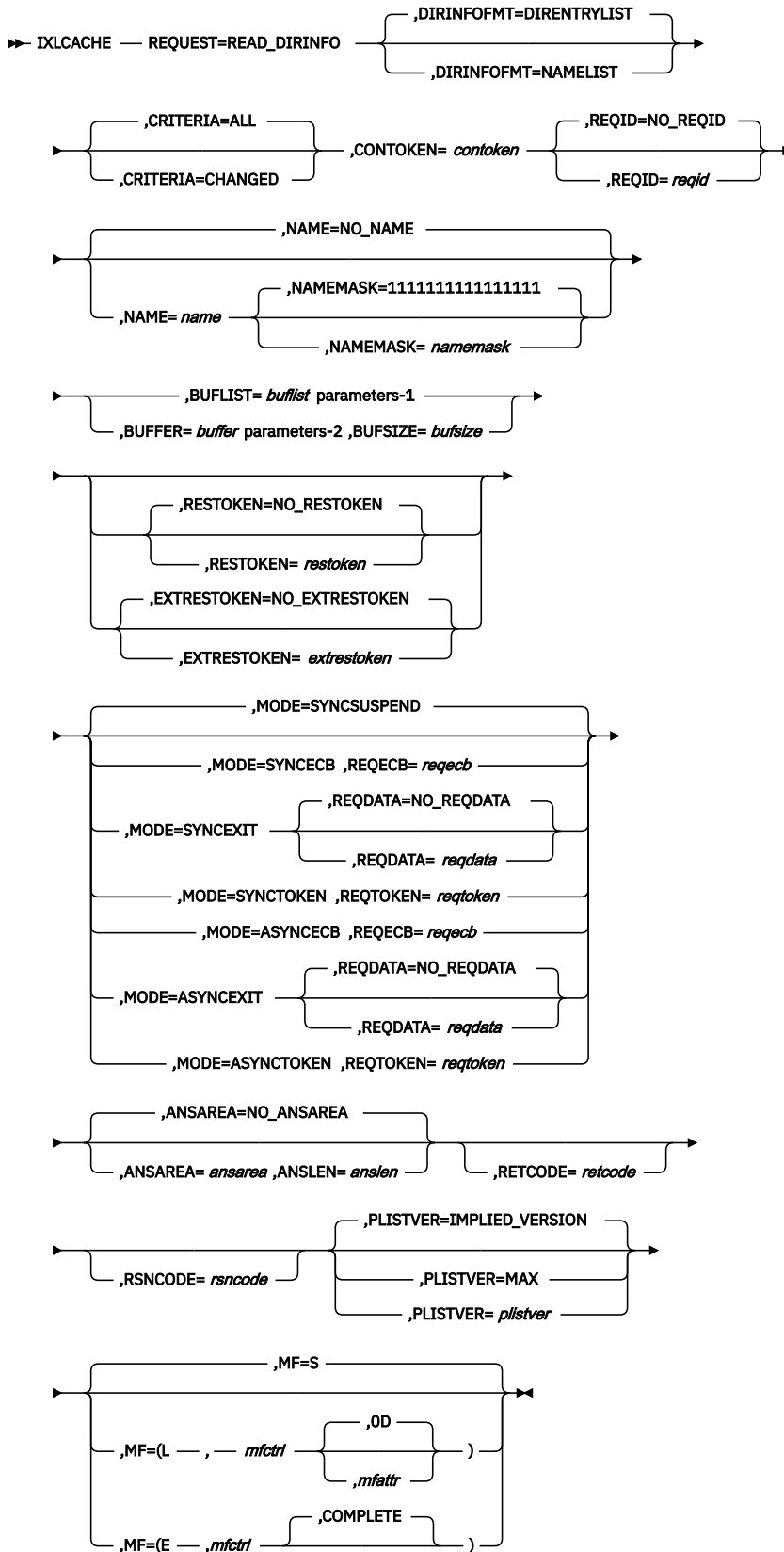
If the request completes prematurely because it exceeds the model-dependent time-out criteria or the specified buffer area (BUFLIST or BUFFER) is full, a restart token (RESTOKEN) or an extended restart token (EXTRESTOKEN) is returned in the answer area. The token can be specified on the next READ\_DIRINFO request to resume processing with the next directory entry to be processed. Resumed requests are processed identically whether using the RESTOKEN or EXTRESTOKEN to specify the starting location.

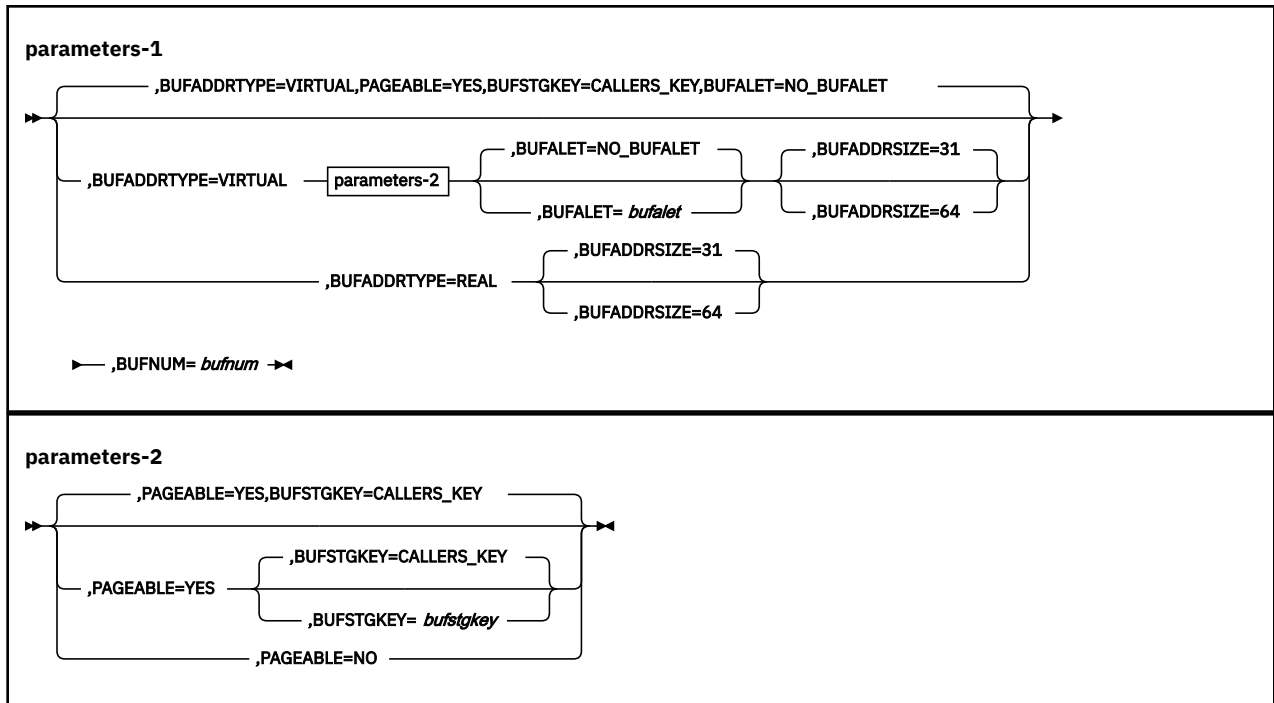
### Syntax Diagram

---

The syntax diagram for IXLCACHE REQUEST=READ\_DIRINFO is as follows:

## main diagram





**Note:** When MODE=SYNCTOKEN or MODE=ASYNCTOKEN is specified, ANSAREA=ansarea,ANSLEN=anslen is required.

## Parameter Descriptions

The parameter descriptions for REQUEST=READ\_DIRINFO are listed in alphabetical order. Default values are underlined:

### REQUEST=READ\_DIRINFO

Use this input parameter to specify that directory information be retrieved from the directory entries specified by CRITERIA, NAME and NAMEMASK, and placed in the storage areas specified by BUFLIST or BUFFER. The format and content of the directory information returned is specified by the DIRINFOFMT parameter.

### ,ANSAREA=NO\_ANSAREA

### ,ANSAREA=ansarea

Use this output parameter to specify an answer area to contain information returned from the request. The information may include a restart token or extended restart token when the request exceeds the model-dependent time-out criteria or the specified buffer area is filled. The format of the answer area is described by the IXLYCAA mapping macro.

If the request completes successfully, the answer area contains the number of directory entries processed by the request (field CAADIRCOUNT).

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an area (with a length of ANSLEN) where the information returned from the request will be put.

### ,ANSLEN=anslen

Use this inppaameter to specify the size of the storage area specified by ANSAREA.

Use either CAALEVELOLEN, CAALEVEL1LEN or CAALEVEL2LEN of the IXLYCAA mapping macro to determine the minimum size of the answer area. The answer area length must be at least large enough to accommodate the level of the IXLYCAA mapping appropriate to the requested function.

- When the connection specified ASYNCXI=1 on the IXLCONN invocation and AXIOVERRIDE=0 was specified or defaulted to for the IXLCACHE request, the answer area length is a required parameter and must be a minimum value of CAALEVEL2LEN to contain a returned asynchronous cross-invalidation sequence number (CAAASYNXISEQNUM).

- When the value of PLISTVER is 4 or above, the minimum answer area length is CAALEVEL1LEN.
- When the value of PLISTVER is 0 - 3, the minimum answer area length is CAALEVEL0LEN.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12 2 to ) of a 2-byte field that contains the length of the answer area (ANSAREA).

**,BUFADDRSIZE=31**

**,BUFADDRSIZE=64**

Use this input parameter to specify whether a 31-bit or a 64-bit address is specified by a BUFLIST entry.

**31**

The entry in BUFLIST is 31 bits in size.

**64**

The entry in BUFLIST is 64 bits in size.

**,BUFADDRTYPE=VIRTUAL**

**,BUFADDRTYPE=REAL**

Use this input parameter to specify whether the buffer addresses specified in the BUFLIST list are virtual storage or real storage addresses.

**VIRTUAL**

The buffer addresses are virtual storage addresses. The virtual storage can be pageable or nonpageable. See the PAGEABLE parameter for information about managing storage binds when specifying virtual storage addresses.

**REAL**

The buffer addresses are real storage addresses.

It is the caller's responsibility to manage the binds between the data buffer virtual storage and the real storage addresses provided. The caller must ensure that the data buffer virtual storage remains bound to the real storage addresses provided until the request completes.

**,BUFALET=NO\_BUFALET**

**,BUFALET=*bufalet***

Use this input parameter to specify an access list entry token (ALET) to be used in referencing all of the buffers specified by BUFLIST.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 4-byte field that contains the ALET.

**,BUFFER=*buffer***

Use this output parameter to specify a buffer area to contain the directory information returned from the request.

You must ensure that the storage area specified by BUFFER:

- Is a multiple of 4096 bytes.
- Is less than or equal to 65536 bytes.
- Starts on a 4096-byte boundary.
- Does not start below storage address 512.

Use the BUFSIZE description for specifying the size of the buffer.

**Note:** See the DIRINFOFMT parameter for information about the format of the data returned in the buffers.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an area (with a length of BUFSIZE) to contain the data returned from the request.

**,BUFLIST=*buflist***

Use this output parameter to specify a list of buffers to hold the directory information returned from the request. BUFLIST specifies a 128-byte storage area that consists of a list of 0 to 16 buffer addresses.

The format of the list is a set of 8-byte elements. The BUFADDRSIZE keyword denotes whether four or eight bytes of the element are used.

- If BUFADDRSIZE=31 is specified, then the first four bytes of each element are reserved space and the last four bytes contain the address of the buffer.
- If BUFADDRSIZE=64 is specified, then the full eight bytes specify the address of the buffer.

**The BUFLIST buffers must:**

- Reside in the same address space or same data space.
- Be 4096 bytes.
- Start on a 4096-byte boundary.
- Not start below storage address 512.

**Note:** The buffers do not have to be contiguous in storage. Cache services treats BUFLIST buffers as a single buffer even if the buffers are not contiguous.

Use the BUFNUM parameter to specify the number of buffers in the buffer list.

**Note:** See the DIRINFOFMT parameter for information about the format of the data returned in the buffers.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 128-byte area that contains a list of buffer addresses.

**,BUFNUM=*bufnum***

Use this input parameter to specify the number of buffers in the BUFLIST list. Valid BUFNUM values are 0 - 16. A value of zero indicates that no data is to be read into the buffers.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 1-byte field that contains the number of buffers (0 - 16) in the list (BUFLIST).

**,BUFSIZE=*bufsize***

Use this input parameter to specify the size of the BUFFER area. See the BUFFER parameter description for valid buffer sizes.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a fullword field that contains the size of the buffer (BUFFER) in bytes.

**,BUFSTGKEY=CALLERS\_KEY**

**,BUFSTGKEY=*bufstgkey***

Use this input parameter to specify a storage key that you define and use when referencing the buffers specified by BUFLIST or the buffer that is specified by BUFFER.

If you do not specify BUFSTGKEY, or if you specify BUFSTGKEY=CALLERS\_KEY, all references to one or more buffers are performed by using the caller's PSW key.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of an 8-bit field that contains the storage key in the format B'kkkkxxxx', where kkkk is the key and xxxx is ignored.

**,CONTO=*conken***

Use this input parameter to specify the connect token that was returned by the IXLCONN service in the IXLCONN answer area, which is mapped which is by IXLYCONA. The connect token uniquely identifies your connection to the cache structure, and must be specified on each IXLCACHE invocation.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field that contains the connect token.

**,CRITERIA=ALL**

**,CRITERIA=CHANGED**

Use this input parameter to specify the criteria for selecting which directory entries will be read. (You can further limit the selection by specifying NAME and optionally NAMEMASK.)

**ALL**

Directory entry information associated with all cached data items will be retrieved.

**CHANGED**

Directory entry information associated with only those data items that are changed or locked-for-cast-out will be retrieved.

**,DIRINFOFMT=DIRENTRYLIST****,DIRINFOFMT=NAMELIST**

Use this input parameter to specify what directory information will be returned to the BUFFER area or BUFLIST buffers for each of the specified data items.

**DIRENTRYLIST**

All the directory entry information is returned. The information returned includes:

- Name
- User data
- Storage class assigned to
- Indication of whether data is currently cached
- Indication of whether data is changed (if cached)
- Castout class assigned to (if changed)
- Parity bits
- Value and stat of the cast-out lock
- Indication of which connected users have registered interest
- Size of the data entry
- For structures allocated in a coupling facility of CFLEVEL=5 or higher, the version number.

See the IXLYDEIB macro for the format of the data returned in the BUFFER area or the BUFLIST buffers.

**NAMELIST**

A subset of the directory entry information is returned. The information returned includes:

- Name
- User data
- Size of the data entry.

See the IXLYCANB macro for the format of the data returned in the BUFFER area or BUFLIST buffers.

For more information about the format of the returned information, see *z/OS MVS Programming: Sysplex Services Guide* or the mapping macros as described in *z/OS MVS Data Areas* in the *z/OS Internet library* ([www.ibm.com/servers/resourcelink/svc00100.nsf/pages/zosInternetLibrary](http://www.ibm.com/servers/resourcelink/svc00100.nsf/pages/zosInternetLibrary)).

**,EXTRESTOKEN=NO\_EXTRESTOKEN****,EXTRESTOKEN=extrestoken**

Use this input parameter to specify an extended restart token that can be used to resume processing of a READ\_DIRINFO request that completed prematurely because it exceeded the model-dependent time-out criteria or because the specified buffer area is full. The extended restart token is returned in the answer area (field CAAEXTRESTOKEN), and should be specified on the next READ\_DIRINFO request to resume processing with the next data item to be processed.

If the request does not exceed the model-dependent time-out criteria or the buffer does not become full, the extended restart token will not be provided.

**Note:**

1. Specifying an extended restart token of all zeros causes cache services to treat all of the entries as unprocessed.



2. Do not specify an extended restart token other than the one returned in the answer area or one set to all zeros, because results will be unpredictable.
3. Specifying an extended restart token requires that the length of the answer area be at least the length of CAALEVEL1LEN.

Requestors that specify IXLCONN ALLOWAUTO=YES must use the extended restart token. Requestors that specify or default to IXLCONN ALLOWAUTO=NO must use the standard restart token (RESTOKEN).

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-byte field that contains the extended restart token.

**,MF=S**  
**,MF=(L,mfctrl)**  
**,MF=(L,mfctrl,mfattr)**  
**,MF=(L,mfctrl,0D)**  
**,MF=(M,mfctrl)**  
**,MF=(M,mfctrl,COMPLETE)**  
**,MF=(M,mfctrl,NOCHECK)**  
**,MF=(E,mfctrl)**  
**,MF=(E,mfctrl,COMPLETE)**  
**,MF=(E,mfctrl,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

**,mfctrl**

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

**,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

**,COMPLETE**

**,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

**COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if SMILE=*var* were an optional parameter and the default is SMILE=NO\_SMILE then it would not be documented. However, if the default was SMILE=:), then it would be documented because a value would be the default.

**NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

**,MODE=SYNCSUSPEND**

**,MODE=SYNCECB**

**,MODE=SYNCEXIT**

**,MODE=SYNCTOKEN**

**,MODE=ASYNCECB**

**,MODE=ASYNCEXIT**

**,MODE=ASYNCTOKEN**

Use this input parameter to specify:

- Whether the request is to be performed synchronously or asynchronously
- How you want to be notified of request completion if the request is processed asynchronously.

See *z/OS MVS Programming: Sysplex Services Guide* for more information on understanding synchronous and asynchronous cache operations.

**SYNCSUSPEND**

The request processes synchronously. If necessary, the request is suspended until it can complete synchronously. To use this option, your program must be enabled for I/O and external interrupts.

**SYNCECB**

The request processes synchronously if possible. If the request processes asynchronously, the ECB specified by REQECB is posted when the request completes.

**SYNCEXIT**

The request processes synchronously if possible. If the request processes asynchronously, your complete exit is given control when the request completes.

**SYNCTOKEN**

The request processes synchronously if possible. If the request processes asynchronously, an asynchronous request token is returned to the area specified by REQTOKEN. Use the returned request token on the IXLFCOMP macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=SYNCTOKEN is specified.

**ASYNCECB**

The request processes asynchronously. The ECB specified by REQECB is posted when the request completes.

**ASYNCEXIT**

The request processes asynchronously. Your complete exit is given control when the request completes.

**ASYNCTOKEN**

The request processes asynchronously. An asynchronous request token is returned to the area specified by REQTOKEN. Use the returned request token on the IXLFCOMP macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=ASYNCTOKEN is specified.

**,NAME=NO\_NAME**

**,NAME=name**

Use this input parameter to filter by name, the data item for which associated directory entry information will be read. You may use this parameter along with the NAMEMASK parameter to select certain data items. See the NAMEMASK parameter for more information on this topic.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-byte field that contains the name of the data item.

**,NAMEMASK=1111111111111111**

**,NAMEMASK=namemask**

Use this input parameter to specify which characters in the name specified by NAME are to be used in selecting data items for processing. This parameter allows you to select multiple data items based on common characters in NAME.

The position of each bit in NAMEMASK corresponds to the same relative character position in NAME. A one indicates that the corresponding letter should be used in selecting entries; a zero indicates that the corresponding letter should not be used.

Specifying a name mask with all zeros causes all names to be selected for processing. Specifying a name mask with all ones causes only the name specified by NAME to be selected.

For more information on how NAMEMASK may be used, see *z/OS MVS Programming: Sysplex Services Guide*.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12 2 to ) of a 2-byte field that contains the bit-mask for the name specified on the NAME keyword.

**,PAGEABLE=YES**

**,PAGEABLE=NO**

Use this input parameter to identify whether the storage areas specified by BUFFER or BUFLIST are in pageable or potentially pageable storage.

#### **YES**

Specify this option to indicate that the BUFFER or BUFLIST buffers reside in pageable virtual storage. XES performs the required page fixing to fix the buffers in real storage while the cache or list request transfers data to or from the coupling facility.

This includes storage obtained from pageable subpools, disabled reference (DREF) storage, and may include storage that has the potential to become pageable during the processing of a request. (An example is address space storage owned by any swappable address space, for which a PGSER FIX has been successfully processed, but for which the owning address space gets swapped during processing of a cache or list request.) This does not include implicitly non-pageable storage (for example, storage obtained from non-pageable subpools).

The system takes responsibility for managing binds to central storage for the duration of the cache or list request, regardless of what address space owns the storage or whether the storage-owning address space is swappable or nonswappable. The storage can be owned by any address space.

High shared virtual storage areas (above 2GB) may not be used.

#### **NO**

Specify this option to indicate that the BUFFER or BUFLIST buffers reside in non-pageable virtual storage. XES does not page fix the buffers in real storage.

This includes implicitly non-pageable storage areas (for example, storage obtained from non-pageable subpools), and may include storage that has the potential to become pageable during the processing of a request (An example is address space storage owned by any swappable address space, for which a PGSER FIX has been successfully processed, but for which the owning address space gets swapped-out during processing of a cache or list request.)

The system takes responsibility for managing binds to central storage for the duration of the cache or list request, if and only if the non-pageable storage is owned by either the requester's address space or the connector's address space. If the storage is owned by any other address space, then the invoker is responsible for ensuring that the virtual storage remains non-pageable for the duration of the request (including the case in which the storage is owned by a swappable address space that is swapped during processing of an IXLCACHE or IXLLIST request). Subject to this consideration, the storage can be owned by any address space. See *z/OS MVS Programming: Sysplex Services Guide*.

**,PLISTVER=IMPLIED\_VERSION****,PLISTVER=MAX****,PLISTVER=*plistver***

Use this input parameter to specify the version of the macro. See “[Understanding IXLCACHE Version Support](#)” on page 461 for a description of the options available with PLISTVER.

**,REQDATA=NO\_REQDATA****,REQDATA=*reqdata***

Use this input parameter with MODE=SYNCEXIT or MODE=ASYNCEXIT to pass any data you choose to the complete exit. The exit will get control only if the request is processed asynchronously.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the data to be passed to the complete exit.

**,REQECB=*reqecb***

Use this output parameter with either MODE=SYNCECB or MODE=ASYNCECB to specify the address of an ECB, which is to be posted when the request completes if the request was processed asynchronously.

Before coding REQECB, you must ensure that:

- You initialize the ECB before you issue the request.
- The ECB resides in either common storage or the home address space where IXLCONN was issued.
- Any tasks that wait for the ECB to be posted reside in the home address space where IXLCONN was issued.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that contains the address of the ECB to be posted when the request completes. The ECB must be aligned on a fullword boundary.

**,REQID=NO\_REQID****,REQID=*reqid***

Use this input parameter to specify a user-defined request identifier to be associated with the request. You can specify this request identifier on the IXPURGE macro to cancel a request that has not yet been processed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the user-defined request identifier.

**,REQTOKEN=*reqtoken***

Use this output parameter with either MODE=SYNCTOKEN or MODE=ASYNCTOKEN to specify the address of a storage area to receive the request token that is returned when the request will be processed asynchronously. This token, which uniquely identifies the request, must be used as input to the IXLFComp macro, which you use to determine if the request has completed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field where the system will put the request token.

**,RESTOKEN=NO\_RESTORE****,RESTOKEN=*restoken***

Use this input parameter to specify a restart token that can be used to resume processing of a READ\_DIRINFO request that completes prematurely because it exceeds the model-dependent time-out criteria or the specified buffer area (BUFLIST or BUFFER) is full. The restart token is returned in the answer area, and should be specified on the next READ\_DIRINFO request to resume processing with the next directory entry to be processed.

If the request does not exceed the model-dependent time-out criteria or the buffer does not become full, the returned token will not be provided.

**Note:**

1. Specifying a restart token of all zeros causes cache services to treat all of the directory entries as unprocessed.

2. Do not specify a restart token other than the one returned in the answer area or one set to all zeros because results will be unpredictable.

Requestors that specify or default to IXLCONN ALLOWAUTO=NO must use the standard restart token. Requestors that specify IXLCONN ALLOWAUTO=YES must use the extended restart token (EXTRESTOKEN).

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-byte field that contains the restart token.

#### **,RETCODE=retcode**

Use this output parameter to specify a field to contain the return code. (The return code is also returned in register 15.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the return code when the request has completed.

#### **,RSNCODE=rsncode**

Use this output parameter to specify a field to contain the reason code returned, if applicable. (The reason code is also returned in register 0.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the reason code (if any) when the request has completed.

## ABEND Codes

---

Abend X'026' (See [z/OS MVS System Codes](#) for more information on this abend.)

## Return and Reason Codes

---

When the system returns control to your program:

- GPR 15 (and RETCODE, if specified) contains a return code.
- If the return code is not zero, GPR 0 (and RSNCODE, if specified) contains a reason code.

**Note:** The return and reason codes will also be put into the answer area (mapped by IXLYCAA), if there is one, when the request completes.

The IXLYCON macro provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

<b>0</b>	IXLRETCODEOK
<b>4</b>	IXLRETCODEWARNING
<b>8</b>	IXLRETCODEPARMERROR
<b>C</b>	IXLRETCODEENVEERROR
<b>10</b>	IXLRETCODECOMPERROR

The following table contains hexadecimal return and reason codes, the equate symbols associated with each reason code, and the meaning and suggested action for each return and reason code.

Table 39. Return and Reason Codes for the IXLCACHE REQUEST=READ\_DIRINFO Macro

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0	None.	<p><b>Meaning:</b> If you specified a MODE value of ASYNCECB, ASYNCEXIT, or ASYNCTOKEN, the request has been successfully initiated. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed. For any other MODE value, the request has successfully completed.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=ASYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=ASYNCEXIT, the connection's complete exit will be given control when the request has completed.</li> <li>• If you specified MODE=ASYNCTOKEN, use REQTOKEN when calling IXLFComp to determine when the request has completed.</li> </ul>
4	xxxx0402	<p><b>Equate Symbol:</b> IXLRNCodeASYNCH</p> <p><b>Meaning:</b> The request could not be processed synchronously. It will be processed asynchronously. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed.</p> <ul style="list-style-type: none"> <li>• If you specified MODE=SYNCECB, the ECB specified in REQECB will be posted when the request has finished.</li> <li>• If you specified MODE=SYNCEXIT, the connection's complete exit will be given control when the request has finished.</li> <li>• If you specified MODE=SYNCTOKEN, the REQTOKEN contains an asynchronous request token that may be used on the IXLFComp macro to determine when the request has completed.</li> </ul> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=SYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=SYNCEXIT, the connection's completion exit will be given control when the request has completed.</li> <li>• If you specified MODE=SYNCTOKEN, use REQTOKEN when calling IXLFComp to determine when the request has completed.</li> </ul>

Table 39. Return and Reason Codes for the IXLCACHE REQUEST=READ\_DIRINFO Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	xxxx0409	<p><b>Equate Symbol:</b> IXLRNCODETIMEOUT</p> <p><b>Meaning:</b> The request has completed prematurely because the model-dependent time-out criteria of the coupling facility has been exceeded. The following information has been returned in the answer area:</p> <ul style="list-style-type: none"> <li>• The number of elements returned in the BUFFER or BUFLIST area (field CAADIRCOUNT)</li> <li>• A token for restarting the request (field CAARESTOKEN or CAAEXTRESTOKEN).</li> </ul> <p><b>Action:</b> Reissue the request using the restart token (RESTOKEN or CAAEXTRESTOKEN).</p> <p>Be sure to process the information returned from this request before reissuing the request. The data returned from this request will be overwritten if you specify the same buffer address. Continue to reissue the request until the return code indicates that all processing has completed.</p> <p>For more information about premature request completion, see <a href="#">z/OS MVS Programming: Sysplex Services Guide</a>.</p>
4	xxxx040F	<p><b>Equate Symbol:</b> IXLRNCODEBUFFERFULL</p> <p><b>Meaning:</b> The request completed prematurely because the buffer specified by BUFFER, or the buffers specified by BUFLIST, is full. The following information has been returned in the answer area:</p> <ul style="list-style-type: none"> <li>• The number of elements returned in the BUFFER or BUFLIST area (field CAADIRCOUNT)</li> <li>• A token for restarting the request (field CAARESTOKEN or CAAEXTRESTOKEN)</li> </ul> <p><b>Action:</b> Reissue the request, or increase the size of the buffer(s) and rerun your program. You can reissue the request using the restart token (RESTOKEN or EXTRESTOKEN).</p> <p>Be sure to process the information returned from this request before reissuing the request. The data returned from this request will be overwritten if you specify the same buffer address. Continue to reissue the request until the return code indicates that all processing has completed.</p> <p>For more information about premature request completion, see <a href="#">z/OS MVS Programming: Sysplex Services Guide</a>.</p>

Table 39. Return and Reason Codes for the IXLCACHE REQUEST=READ\_DIRINFO Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0801	<p><b>Equate Symbol:</b> IXLRSNCODEBADPARMLIST</p> <p><b>Meaning:</b> The parameter list (mfctrl) for this request is not addressable. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the parameter list address.</li> <li>• The parameter list must be addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are issuing this macro while disabled, the parameter list must reside in either page-fixed or DREF storage.</li> <li>• If you are issuing this macro in AR-mode and you specified the parameter list address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are issuing this macro in AR-mode, SYSSTATE ASCENV=AR must be issued before this macro.</li> </ul>
8	xxxx0804	<p><b>Equate Symbol:</b> IXLRSNCODEBADVERSION#</p> <p><b>Meaning:</b> The version number in the macro parameter list is not compatible with the level of XES currently being used. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that your program did not overlay the parameter list storage.</li> <li>• Verify that your program was assembled with the correct macro library for the release of MVS on which your program is running.</li> <li>• Verify that your program is running on an MVS system that supports the version of the macro you are using.</li> </ul>



Table 39. Return and Reason Codes for the IXLCACHE REQUEST=READ\_DIRINFO Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx080A	<p><b>Equate Symbol:</b> IXLSRNCODEBADCONTOKEN</p> <p><b>Meaning:</b> Program error. The specified connect token (CONTOKEN) was not valid for one of the following reasons:</p> <ol style="list-style-type: none"> <li>1. The user with the connection identifier represented by the token has disconnected from the structure.</li> <li>2. The connector's task (the task that issued IXLCONN) ended.</li> <li>3. The specified token is not the token that was returned from IXLCONN.</li> <li>4. The request was issued from an address space other than the address space in which IXLCONN was issued.</li> <li>5. The connect token was invalidated during rebuild.</li> <li>6. The connect token was invalidated by XES.</li> </ol> <p><b>Note:</b> The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Verify that the CONTOKEN value specified was valid and for the correct structure. The numbers on the following actions correspond to the numbers above.</p> <ol style="list-style-type: none"> <li>1. Discontinue use. Perform recovery and cleanup for the structure and check protocol for use of structure to determine why you did not know of the disconnection.</li> <li>2. Discontinue use. Perform recovery and cleanup for the structure and check protocol for use of structure to determine why you did not know that the task ended.</li> <li>3. Issue IXLCONN to connect to the structure. The contoken is returned in the CONACONTOKEN field of the answer area specified on the IXLCONN request.</li> <li>4. Issue your request from the same address space the IXLCONN was issued.</li> <li>5. Participate in the rebuild. When it is complete, try again.</li> <li>6. Discontinue use. Perform recovery and cleanup for the structure and check protocol for use of structure.</li> </ol> <p>You may want to issue IXCQUERY to get more information about the structure.</p>
8	xxxx0824	<p><b>Equate Symbol:</b> IXLSRNCODEWRONGSTRTYPE</p> <p><b>Meaning:</b> The connect token specified by CONTOKEN is not to a cache structure.</p> <p><b>Action:</b> Verify the connect token for this cache structure.</p> <p><b>Note:</b> The connect token can be found in the IXLCONN answer area. Verify the value specified on the TYPE= parameter of the IXLCONN request for this structure.</p>

Table 39. Return and Reason Codes for the IXLCACHE REQUEST=READ\_DIRINFO Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0833	<p><b>Equate Symbol:</b> IXLRSNCODEBADPGBLATTR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list is specified as being pageable (PAGEABLE=YES) but is not.</p> <p><b>Action:</b> Change the buffer area(s) to pageable storage, or specify PAGEABLE=NO. See the PAGEABLE parameter description for specification instructions.</p>
8	xxxx0834	<p><b>Equate Symbol:</b> IXLRSNCODEBADNONPGBLATTR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list is specified as being nonpageable (PAGEABLE=NO) but is either pageable or not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The type of storage in the buffer area corresponds to the value you specified on the PAGEABLE parameter. See the PAGEABLE parameter description for specification instructions.</li> <li>• The correct buffer address was used.</li> <li>• The buffer area(s) were not previously freed.</li> <li>• If BUFLIST was specified and your program is running in AR mode, ensure that: <ul style="list-style-type: none"> <li>– The BUFALET specification is correct.</li> <li>– You specified SYSSTATE ASCENV=AR before issuing the macro.</li> </ul> </li> <li>• If the caller is running in AR-mode and the BUFFER or BUFLIST parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> </ul>

Table 39. Return and Reason Codes for the IXLCACHE REQUEST=READ\_DIRINFO Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0835	<p><b>Equate Symbol:</b> IXLRNCODEBADDATAADDR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list, is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The correct buffer address was used for BUFFER or for a buffer within the BUFLIST.</li> <li>• The buffer area(s) were not previously freed.</li> <li>• The buffer area(s) were allocated in a storage key that matches the key specified using the BUFSTGKEY parameter, or, if BUFSTGKEY is omitted, the caller's PSW key.</li> <li>• If the caller is running in AR-mode and the BUFFER or BUFLIST parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If the caller is running in AR-mode, SYSSTATE ASCENV=AR must be specified before issuing this macro.</li> <li>• If BUFLIST was specified and your program is running in AR mode the BUFALET specification is correct.</li> </ul>
8	xxxx0836	<p><b>Equate Symbol:</b> IXLRNCODEBADREALADDR</p> <p><b>Meaning:</b> Program error. Real storage addresses were provided in a BUFLIST list, but one of the buffers is not addressable in central storage.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that BUFADDRTYPE was specified as you intended.</li> <li>• Ensure that the real buffer addresses specified by BUFLIST are valid.</li> </ul>
8	xxxx0838	<p><b>Equate Symbol:</b> IXLRNCODEBADANSAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by ANSAREA is not addressable.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the ANSAREA address.</li> <li>• ANSAREA must be addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are calling IXLCACHE while disabled, ANSAREA must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLCACHE in AR-mode and you specified the ANSAREA address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are calling IXLCACHE in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLCACHE macro.</li> </ul>

Table 39. Return and Reason Codes for the IXLCACHE REQUEST=READ\_DIRINFO Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0839	<p><b>Equate Symbol:</b> IXLRNCODEBADREQTOKENAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by REQTOKEN is not addressable.</p> <p><b>Action:</b> Ensure that the request token area specified by REQTOKEN is valid:</p> <ul style="list-style-type: none"> <li>• Verify the REQTOKEN address.</li> <li>• If you are calling IXLCACHE while disabled, REQTOKEN must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLCACHE in AR-mode and you specified the REQTOKEN address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are calling IXLCACHE in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLCACHE macro.</li> </ul>
8	xxxx083D	<p><b>Equate Symbol:</b> IXLRNCODEBADANSLEN</p> <p><b>Meaning:</b> Program error. The length of the answer area, as specified by ANSLEN, is not sufficient to contain answer area information. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Increase the size of the answer area provided for the request, and rerun your program. Depending on the macro version number, specify the length of the answer area as follows:</p> <ul style="list-style-type: none"> <li>• When the connection specified ASYNCXI=1 on the IXLCONN invocation and AXIOVERRIDE=0 was specified or defaulted to for the IXLCACHE request, the answer area length must be a minimum value of CAALEVEL2LEN</li> <li>• When the value of the macro version number (PLISTVER) is 4 or more, the minimum answer area length is CAALEVEL1LEN.</li> <li>• When the value of the macro version number (PLISTVER) is 0-3, the minimum answer area length is CAALEVEL0LEN.</li> </ul>
8	xxxx0849	<p><b>Equate Symbol:</b> IXLRNCODEBADRESTOKEN</p> <p><b>Meaning:</b> Program error. The restart token specified by RESTOKEN is not valid.</p> <p><b>Action:</b> Determine why the restart token cannot be used to resume the request. Possible causes are:</p> <ul style="list-style-type: none"> <li>• The specified token does not correspond to the restart token returned in the answer area of the previous request.</li> <li>• The user specified RESTOKEN when EXTRESTOKEN was required.</li> <li>• The user specified EXTRESTOKEN when RESTOKEN was required.</li> </ul> <p>For more information about premature request completion, see <a href="#">z/OS MVS Programming: Sysplex Services Guide</a>.</p>

Table 39. Return and Reason Codes for the IXLCACHE REQUEST=READ\_DIRINFO Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0851	<p><b>Equate Symbol:</b> IXLRNOCODENOSUSPENDISABLE</p> <p><b>Meaning:</b> Program error. The request failed because MODE=SYNCSUSPEND was specified, but the caller is disabled and cannot be suspended.</p> <p><b>Action:</b> Either specify another MODE value or become enabled (release the CPU lock); then reissue the request.</p>
8	xxxx0864	<p><b>Equate Symbol:</b> IXLRNOCODEBADBUFSIZE</p> <p><b>Meaning:</b> Program error. The size of the BUFLIST areas or BUFFER area is not large enough to contain the data being read. No data is returned.</p> <p><b>Action:</b> If more space is available, specify a larger buffer size and reissue the request. Consider using the BUFLIST parameter if BUFFER was specified.</p>
8	xxxx0865	<p><b>Equate Symbol:</b> IXLRNOCODEBADBUFSPEC</p> <p><b>Meaning:</b> Program error. There is an error in the buffer specification.</p> <p><b>Action:</b> Check the following:</p> <ul style="list-style-type: none"> <li>• If BUFLIST was specified, check the requirements for BUFLIST and BUFNUM.</li> <li>• If BUFFER was specified, check the requirements for BUFFER and BUFSIZE.</li> <li>• Buffer pointer(s) in BUFLIST.</li> <li>• Buffer boundaries.</li> </ul>
8	xxxx0866	<p><b>Equate Symbol:</b> IXLRNOCODEBADBUFKEY</p> <p><b>Meaning:</b> Program error. The buffer storage key specified by BUFSTGKEY is incorrect or BUFSTGKEY was not specified and the PSW key does not match the key of the buffers.</p> <p>The data cannot be stored in the specified buffer area.</p> <p><b>Action:</b> Check the following:</p> <ul style="list-style-type: none"> <li>• Determine if the key of the storage being used for the buffers is different from the PSW key.</li> <li>• If BUFSTGKEY was specified, verify that the key was put in the correct bits (see the explanation of this parameter for more information).</li> <li>• If you are calling IXLCACHE in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLCACHE macro.</li> </ul>

Table 39. Return and Reason Codes for the IXLCACHE REQUEST=READ\_DIRINFO Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0867	<p><b>Equate Symbol:</b> IXLRNCOBEBADBUFLIST</p> <p><b>Meaning:</b> Program error. The 128-byte storage area specified by BUFLIST is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The correct BUFLIST address was used.</li> <li>• The BUFLIST area was not previously freed.</li> <li>• If the caller is running in AR-mode and the BUFLIST parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If the caller is disabled, then the BUFLIST must reside in either nonpageable or disabled reference storage.</li> <li>• If your program is running in AR mode, ensure that you specified SYSSTATE ASCENV=AR before issuing the macro.</li> </ul>
8	xxxx0887	<p><b>Equate Symbol:</b> IXLRNCOBEBADEXTRESTOKEN</p> <p><b>Meaning:</b> Program error. The extended restart token specified by EXTRESTOKEN is not valid. The specified token refers to an older instance of the target structure. A system-managed process occurred between the time a request returned the extended restart token and the time the connector tried to continue the request using that token.</p> <p><b>Action:</b> Discard the results of the initial request and reissue the request with an EXTRESTOKEN value of zero. For more information about restarting requests, see <a href="#">z/OS MVS Programming: Sysplex Services Guide</a>.</p>
C	xxxx0C06	<p><b>Equate Symbol:</b> IXLRNCOBENOCN</p> <p><b>Meaning:</b> Environmental error. No connectivity to the cache structure exists. This may occur due to operator commands such as VARY PATH,OFFLINE or CONFIG CHP,OFFLINE or hardware errors such as facility or path failures. The CONTOKEN is invalidated. The request fails.</p> <p><b>Action:</b> Either disconnect from the structure (using IXLDISC) or initiate a rebuild of the structure, if allowed (using IXLREBLD).</p>

Table 39. Return and Reason Codes for the IXLCACHE REQUEST=READ\_DIRINFO Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C13	<p><b>Equate Symbol:</b> IXLRSNCODEREQPURGED</p> <p><b>Meaning:</b> Environmental error. The request was purged prior to completion of the request. Possible reasons include:</p> <ul style="list-style-type: none"> <li>• The connector failed.</li> <li>• The connector disconnected.</li> <li>• The requestor failed.</li> <li>• The request was purged by IXLPURGE.</li> <li>• Requests were purged when the connector provided an IXLEERSP response for the Rebuild Quiesce, Rebuild Stop, or Rebuild Cleanup event.</li> <li>• The connector invoked IXLREBLD REQUEST=COMPLETE.</li> <li>• The secondary address space was no longer valid.</li> </ul> <p><b>Action:</b> None if this is expected. Otherwise, determine why the connector failed.</p>
C	xxxx0C14	<p><b>Equate Symbol:</b> IXLRSNCODESTATUSUNKNOWN</p> <p><b>Meaning:</b> Environmental error. The IXLCACHE request has completed, but the final disposition of the request cannot be determined. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Verify the validity of your data by comparing the expected results with what is in the coupling facility.</p>
C	xxxx0C25	<p><b>Equate Symbol:</b> IXLRSNCODESTRFAILURE</p> <p><b>Meaning:</b> Environmental error. The cache structure failed prior to completion of the request.</p> <p><b>Action:</b> Attempt to rebuild the structure using IXLREBLD, or disconnect from the structure using IXLDISC.</p>
C	xxxx0CA0	<p><b>Equate Symbol:</b> IXLRSNCODEQUIESCEDSUSPENDFAIL</p> <p><b>Meaning:</b> Environmental error. The request is failed because the structure is quiesced for a system-managed process and SUSPEND=FAIL is specified on the IXLCONN.</p> <p><b>Action:</b> None, if this is expected.</p>
C	xxxxFFFF	<p><b>Equate Symbol:</b> IXLRSNCODENOTAVAILABLE</p> <p><b>Meaning:</b> Environmental error. XES functions are not available. The coupling facility hardware might not be present.</p> <p><b>Action:</b> XES may not be used when XCF is running local mode. XES requires that the coupling facility be installed.</p>

Table 39. Return and Reason Codes for the IXLCACHE REQUEST=READ\_DIRINFO Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
10	xxxx10xx	<b>Equate Symbol:</b> IXLRNCODECOMPERROR <b>Meaning:</b> System error. XES processing failure. The state of the structure and the disposition of the request are unpredictable. The answer area (ANSAREA) fields do not contain valid information. <b>Action:</b> Save the reason code information, and contact the IBM support center.



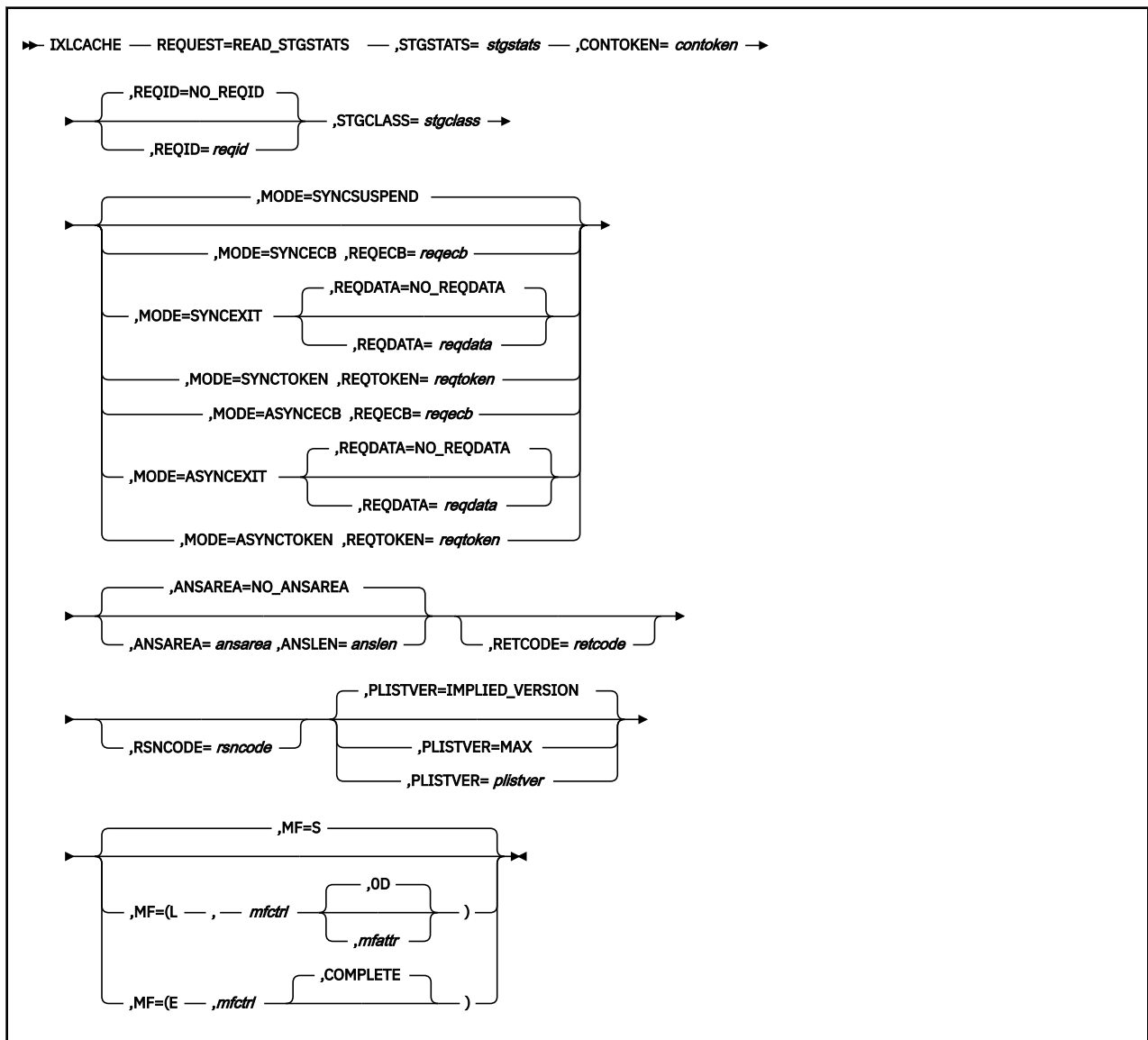
## Chapter 42. IXLCACHE REQUEST=READ\_STGSTATS

### Description

A READ\_STGSTATS request allows you to retrieve statistical information for a specified storage class (STGCLASS) and store it in the area specified by STGSTATS. The format of the output returned by this request is mapped by IXLYCSCS. For more information on the statistical data returned for storage classes, see *z/OS MVS Programming: Sysplex Services Guide*.

### Syntax Diagram

The syntax diagram for IXLCACHE REQUEST=READ\_STGSTATS is as follows:



**Note:** When MODE=SYNCTOKEN or MODE=ASYNCTOKEN is specified, ANSAREA= *ansarea* and ANSLEN= *anslen* must be specified.

## Parameter Descriptions

The parameter descriptions for REQUEST=READ\_STGSTATS are listed in alphabetical order. Default values are underlined:

### **REQUEST=READ\_STGSTATS**

Use this input parameter to specify that statistical information for the specified storage class (STGCLASS) be returned in the storage area specified by STGSTATS.

### **,ANSAREA=NO ANSAREA**

### **,ANSAREA=*ansarea***

Use this output parameter to specify an answer area to contain information returned from the request. The format of the answer area is described by the IXLYCAA mapping macro.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an area (with a length of ANSLEN) where the information returned by the request will be put.

### **,ANSLEN=*anslen***

Use this inppaameter to specify the size of the storage area specified by ANSAREA.

Use either CAALEVELLEN, CAALEVEL1LEN or CAALEVEL2LEN of the IXLYCAA mapping macro to determine the minimum size of the answer area. The answer area length must be at least large enough to accommodate the level of the IXLYCAA mapping appropriate to the requested function.

- When the connection specified ASYNCXI=1 on the IXLCONN invocation and AXIOVERRIDE=0 was specified or defaulted to for the IXLCACHE request, the answer area length is a required parameter and must be a minimum value of CAALEVEL2LEN to contain a returned asynchronous cross-invalidation sequence number (CAAASYNCXISEQNUM).
- When the value of PLISTVER is 4 or above, the minimum answer area length is CAALEVEL1LEN.
- When the value of PLISTVER is 0 - 3, the minimum answer area length is CAALEVELLEN.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12 2 to ) of a 2-byte field that contains the length of the answer area (ANSAREA).

### **,CONTO=*conken***

Use this input parameter to specify the connect token that was returned by the IXLCONN service in the IXLCONN answer area, which is mapped which is by IXLYCONA. The connect token uniquely identifies your connection to the cache structure, and must be specified on each IXLCACHE invocation.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field that contains the connect token.

### **,MF=S**

### **,MF=(L,*mfctrl*)**

### **,MF=(L,*mfctrl*,*mfattr*)**

### **,MF=(L,*mfctrl*,0D)**

### **,MF=(M,*mfctrl*)**

### **,MF=(M,*mfctrl*,COMPLETE)**

### **,MF=(M,*mfctrl*,NOCHECK)**

### **,MF=(E,*mfctrl*)**

### **,MF=(E,*mfctrl*,COMPLETE)**

### **,MF=(E,*mfctrl*,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

#### **,mfctrl**

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

#### **,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

#### **,COMPLETE**

#### **,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

#### **COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if SMILE=*var* were an optional parameter and the default is SMILE=NO\_SMILE then it would not be documented. However, if the default was SMILE=:), then it would be documented because a value would be the default.

#### **NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

#### **,MODE=SYNCSUSPEND**

#### **,MODE=SYNCECB**

#### **,MODE=SYNCEXIT**

#### **,MODE=SYNCTOKEN**

#### **,MODE=ASYNCECB**

#### **,MODE=ASYNCEXIT**

#### **,MODE=ASYNCTOKEN**

Use this input parameter to specify:

- Whether the request is to be performed synchronously or asynchronously
- How you want to be notified of request completion if the request is processed asynchronously.

See *z/OS MVS Programming: Sysplex Services Guide* for more information on understanding synchronous and asynchronous cache operations.

#### **SYNCSUSPEND**

The request processes synchronously. If necessary, the request is suspended until it can complete synchronously. To use this option, your program must be enabled for I/O and external interrupts.

#### **SYNCECB**

The request processes synchronously if possible. If the request processes asynchronously, the ECB specified by REQECB is posted when the request completes.

#### **SYNCEXIT**

The request processes synchronously if possible. If the request processes asynchronously, your complete exit is given control when the request completes.

**SYNCTOKEN**

The request processes synchronously if possible. If the request processes asynchronously, an asynchronous request token is returned to the area specified by REQTOKEN. Use the returned request token on the IXLFCOMP macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=SYNCTOKEN is specified.

**ASYNCECB**

The request processes asynchronously. The ECB specified by REQECB is posted when the request completes.

**ASYNCEXIT**

The request processes asynchronously. Your complete exit is given control when the request completes.

**ASYNCTOKEN**

The request processes asynchronously. An asynchronous request token is returned to the area specified by REQTOKEN. Use the returned request token on the IXLFCOMP macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=ASYNCTOKEN is specified.

**,PLISTVER=IMPLIED\_VERSION****,PLISTVER=MAX****,PLISTVER=*plistver***

Use this input parameter to specify the version of the macro. See [“Understanding IXLCACHE Version Support” on page 461](#) for a description of the options available with PLISTVER.

**,REQDATA=NO\_REQDATA****,REQDATA=*reqdata***

Use this input parameter with MODE=SYNCEXIT or MODE=ASYNCEXIT to pass any data you choose to the complete exit. The exit will get control only if the request is processed asynchronously.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the data to be passed to the complete exit.

**,REQECB=*reqecb***

Use this output parameter with either MODE=SYNCECB or MODE=ASYNCECB to specify the address of an ECB, which is to be posted when the request completes if the request was processed asynchronously.

Before coding REQECB, you must ensure that:

- You initialize the ECB before you issue the request.
- The ECB resides in either common storage or the home address space where IXLCONN was issued.
- Any tasks that wait for the ECB to be posted reside in the home address space where IXLCONN was issued.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that contains the address of the ECB to be posted when the request completes. The ECB must be aligned on a fullword boundary.

**,REQID=NO\_REQID****,REQID=*reqid***

Use this input parameter to specify a user-defined request identifier to be associated with the request. You can specify this request identifier on the IXLPURGE macro to cancel a request that has not yet been processed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the user-defined request identifier.

**,REQTOKEN=*reqtoken***

Use this output parameter with either MODE=SYNCTOKEN or MODE=ASYNCTOKEN to specify the address of a storage area to receive the request token that is returned when the request will be

processed asynchronously. This token, which uniquely identifies the request, must be used as input to the IXLFCOMP macro, which you use to determine if the request has completed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field where the system will put the request token.

**,RETCODE=retcode**

Use this output parameter to specify a field to contain the return code. (The return code is also returned in register 15.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the return code when the request has completed.

**,RSNCODE=rsncode**

Use this output parameter to specify a field to contain the reason code returned, if applicable. (The reason code is also returned in register 0.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the reason code (if any) when the request has completed.

**,STGCLASS=stgclass**

Use this input parameter to specify the storage class for which statistical information will be returned.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 1-byte field that contains the storage class.

**,STGSTATS=stgstats**

Use this output parameter to specify a storage area to contain the storage class statistics returned from the request. The format of the STGSTATS area is described by mapping macro IXLYCSCS.

For more information about the data returned by this request, see *z/OS MVS Programming: Sysplex Services Guide*.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 256-byte area where the storage statistics information will be placed.

## ABEND Codes

---

Abend X'026' (See *z/OS MVS System Codes* for more information on this abend.)

## Return and Reason Codes

---

When the system returns control to your program:

- GPR 15 (and RETCODE, if specified) contains a return code.
- If the return code is not zero, GPR 0 (and RSNCODE, if specified) contains a reason code.

**Note:** The return and reason codes will also be put into the answer area (mapped by IXLYCAA), if there is one, when the request completes.

The IXLYCON macro provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

<b>0</b>	IXLRETCODEOK
<b>4</b>	IXLRETCODEWARNING
<b>8</b>	IXLRETCODEPARMERROR
<b>C</b>	IXLRETCODEENVERROR
<b>10</b>	IXLRETCODECOMPERROR

The following table contains hexadecimal return and reason codes, the equate symbols associated with each reason code, and the meaning and suggested action for each return and reason code.

Table 40. Return and Reason Codes for the IXLCACHE REQUEST=READ_STGSTATS Macro		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0	None.	<p><b>Meaning:</b> If you specified a MODE value of ASYNCECB, ASYNCEXIT, or ASYNCTOKEN, the request has been successfully initiated. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed. For any other MODE value, the request has successfully completed.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=ASYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=ASYNCEXIT, the connection's complete exit will be given control when the request has completed.</li> <li>• If you specified MODE=ASYNCTOKEN, use REQTOKEN when calling IXLFComp to determine when the request has completed.</li> </ul>
4	xxxx0402	<p><b>Equate Symbol:</b> IXLRNCodeASYNCH</p> <p><b>Meaning:</b> The request could not be processed synchronously. It will be processed asynchronously. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed.</p> <ul style="list-style-type: none"> <li>• If you specified MODE=SYNCECB, the ECB specified in REQECB will be posted when the request has finished.</li> <li>• If you specified MODE=SYNCEXIT, the connection's complete exit will be given control when the request has finished.</li> <li>• If you specified MODE=SYNCTOKEN, the REQTOKEN contains an asynchronous request token that may be used on the IXLFComp macro to determine when the request has completed.</li> </ul> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=SYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=SYNCEXIT, the connection's completion exit will be given control when the request has completed.</li> <li>• If you specified MODE=SYNCTOKEN, use REQTOKEN when calling IXLFComp to determine when the request has completed.</li> </ul>

Table 40. Return and Reason Codes for the IXLCACHE REQUEST=READ\_STGSTATS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0801	<p><b>Equate Symbol:</b> IXLRSNCODEBADPARMLIST</p> <p><b>Meaning:</b> The parameter list (mfctrl) for this request is not addressable. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the parameter list address.</li> <li>• The parameter list must be addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are issuing this macro while disabled, the parameter list must reside in either page-fixed or DREF storage.</li> <li>• If you are issuing this macro in AR-mode and you specified the parameter list address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are issuing this macro in AR-mode, SYSSTATE ASCENV=AR must be issued before this macro.</li> </ul>
8	xxxx0804	<p><b>Equate Symbol:</b> IXLRSNCODEBADVERSION#</p> <p><b>Meaning:</b> The version number in the macro parameter list is not compatible with the level of XES currently being used. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that your program did not overlay the parameter list storage.</li> <li>• Verify that your program was assembled with the correct macro library for the release of MVS on which your program is running.</li> <li>• Verify that your program is running on an MVS system that supports the version of the macro you are using.</li> </ul>

Table 40. Return and Reason Codes for the IXLCACHE REQUEST=READ\_STGSTATS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx080A	<p><b>Equate Symbol:</b> IXLRNSCODEBADCONTOKEN</p> <p><b>Meaning:</b> Program error. The specified connect token (CONTOKEN) was not valid for one of the following reasons:</p> <ol style="list-style-type: none"> <li>1. The user with the connection identifier represented by the token has disconnected from the structure.</li> <li>2. The connector's task (the task that issued IXLCONN) ended.</li> <li>3. The specified token is not the token that was returned from IXLCONN.</li> <li>4. The request was issued from an address space other than the address space in which IXLCONN was issued.</li> <li>5. The connect token was invalidated during rebuild.</li> <li>6. The connect token was invalidated by XES.</li> </ol> <p><b>Note:</b> The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Verify that the CONTOKEN value specified was valid and for the correct structure. The numbers on the following actions correspond to the numbers above.</p> <ol style="list-style-type: none"> <li>1. Discontinue use. Perform recovery and cleanup for the structure and check protocol for use of structure to determine why you did not know of the disconnection.</li> <li>2. Discontinue use. Perform recovery and cleanup for the structure and check protocol for use of structure to determine why you did not know that the task ended.</li> <li>3. Issue IXLCONN to connect to the structure. The contoken is returned in the CONACONTOKEN field of the answer area specified on the IXLCONN request.</li> <li>4. Issue your request from the same address space the IXLCONN was issued.</li> <li>5. Participate in the rebuild. When it is complete, try again.</li> <li>6. Discontinue use. Perform recovery and cleanup for the structure and check protocol for use of structure.</li> </ol> <p>You may want to issue IXCQUERY to get more information about the structure.</p>
8	xxxx0824	<p><b>Equate Symbol:</b> IXLRNSCODEWRONGSTRTYPE</p> <p><b>Meaning:</b> The connect token specified by CONTOKEN is not to a cache structure.</p> <p><b>Action:</b> Verify the connect token for this cache structure.</p> <p><b>Note:</b> The connect token can be found in the IXLCONN answer area. Verify the value specified on the TYPE= parameter of the IXLCONN request for this structure.</p>



Table 40. Return and Reason Codes for the IXLCACHE REQUEST=READ\_STGSTATS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx082D	<p><b>Equate Symbol:</b> IXLRNCODEBADSTGCLASS</p> <p><b>Meaning:</b> Program error. The storage class specified by STGCLASS exceeds the maximum number of storage classes defined for the cache structure.</p> <p><b>Action:</b> The number of storage classes allowed for a structure are defined on the IXLCONN macro by the NUMSTGCLASS parameter. Correct the STGCLASS parameter to specify a valid storage class.</p>
8	xxxx0838	<p><b>Equate Symbol:</b> IXLRNCODEBADANSAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by ANSAREA is not addressable.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the ANSAREA address.</li> <li>• ANSAREA must be addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are calling IXLCACHE while disabled, ANSAREA must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLCACHE in AR-mode and you specified the ANSAREA address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are calling IXLCACHE in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLCACHE macro.</li> </ul>
8	xxxx0839	<p><b>Equate Symbol:</b> IXLRNCODEBADREQTOKENAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by REQTOKEN is not addressable.</p> <p><b>Action:</b> Ensure that the request token area specified by REQTOKEN is valid:</p> <ul style="list-style-type: none"> <li>• Verify the REQTOKEN address.</li> <li>• If you are calling IXLCACHE while disabled, REQTOKEN must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLCACHE in AR-mode and you specified the REQTOKEN address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are calling IXLCACHE in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLCACHE macro.</li> </ul>

Table 40. Return and Reason Codes for the IXLCACHE REQUEST=READ\_STGSTATS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx083D	<p><b>Equate Symbol:</b> IXLRNCODEBADANSLEN</p> <p><b>Meaning:</b> Program error. The length of the answer area, as specified by ANSLEN, is not sufficient to contain answer area information. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Increase the size of the answer area provided for the request, and rerun your program. Depending on the macro version number, specify the length of the answer area as follows:</p> <ul style="list-style-type: none"> <li>• When the connection specified ASYNCCI=1 on the IXLCONN invocation and AXIOVERRIDE=0 was specified or defaulted to for the IXLCACHE request, the answer area length must be a minimum value of CAALEVEL2LEN</li> <li>• When the value of the macro version number (PLISTVER) is 4 or more, the minimum answer area length is CAALEVEL1LEN.</li> <li>• When the value of the macro version number (PLISTVER) is 0-3, the minimum answer area length is CAALEVEL0LEN.</li> </ul>
8	xxxx0851	<p><b>Equate Symbol:</b> IXLRNCODENOSUSPENDISABLE</p> <p><b>Meaning:</b> Program error. The request failed because MODE=SYNCSUSPEND was specified, but the caller is disabled and cannot be suspended.</p> <p><b>Action:</b> Either specify another MODE value or become enabled (release the CPU lock); then reissue the request.</p>
8	xxxx0869	<p><b>Equate Symbol:</b> IXLRNCODEBADSTGSTATS</p> <p><b>Meaning:</b> Program error. The storage area specified by STGSTATS is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The address passed in STGSTATS is valid.</li> <li>• The STGSTATS area was not previously freed.</li> <li>• If the caller is running in AR-mode and STGSTATS was specified using explicit register notation, ensure that the corresponding access register was updated appropriately.</li> <li>• If the caller is disabled, then STGSTATS must reside in either nonpageable or disabled reference storage.</li> <li>• If your program is running in AR mode, ensure that you specified SYSSTATE ASCENV=AR before issuing the IXLCACHE macro.</li> </ul>

Table 40. Return and Reason Codes for the IXLCACHE REQUEST=READ\_STGSTATS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C06	<p><b>Equate Symbol:</b> IXLRSNCODENOCOONN</p> <p><b>Meaning:</b> Environmental error. No connectivity to the cache structure exists. This may occur due to operator commands such as VARY PATH,OFFLINE or CONFIG CHP,OFFLINE or hardware errors such as facility or path failures. The CONTOKEN is invalidated. The request fails.</p> <p><b>Action:</b> Either disconnect from the structure (using IXLDISC) or initiate a rebuild of the structure, if allowed (using IXLREBLD).</p>
C	xxxx0C13	<p><b>Equate Symbol:</b> IXLRSNCODEREQPURGED</p> <p><b>Meaning:</b> Environmental error. The request was purged prior to completion of the request. Possible reasons include:</p> <ul style="list-style-type: none"> <li>• The connector failed.</li> <li>• The connector disconnected.</li> <li>• The requestor failed.</li> <li>• The request was purged by IXLPURGE.</li> <li>• Requests were purged when the connector provided an IXLEERSP response for the Rebuild Quiesce, Rebuild Stop, or Rebuild Cleanup event.</li> <li>• The connector invoked IXLREBLD REQUEST=COMPLETE.</li> <li>• The secondary address space was no longer valid.</li> </ul> <p><b>Action:</b> None if this is expected. Otherwise, determine why the connector failed.</p>
C	xxxx0C14	<p><b>Equate Symbol:</b> IXLRSNCODESTATUSUNKNOWN</p> <p><b>Meaning:</b> Environmental error. The IXLCACHE request has completed, but the final disposition of the request cannot be determined. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Verify the validity of your data by comparing the expected results with what is in the coupling facility.</p>
C	xxxx0C25	<p><b>Equate Symbol:</b> IXLRSNCODESTRFAILURE</p> <p><b>Meaning:</b> Environmental error. The cache structure failed prior to completion of the request.</p> <p><b>Action:</b> Attempt to rebuild the structure using IXLREBLD, or disconnect from the structure using IXLDISC.</p>
C	xxxx0CA0	<p><b>Equate Symbol:</b> IXLRSNCODEQUIESCEDSUSPENDFAIL</p> <p><b>Meaning:</b> Environmental error. The request is failed because the structure is quiesced for a system-managed process and SUSPEND=FAIL is specified on the IXLCONN.</p> <p><b>Action:</b> None, if this is expected.</p>

Table 40. Return and Reason Codes for the IXLCACHE REQUEST=READ\_STGSTATS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxxFFFF	<p><b>Equate Symbol:</b> IXLRSNCODENOTAVAILABLE</p> <p><b>Meaning:</b> Environmental error. XES functions are not available. The coupling facility hardware might not be present.</p> <p><b>Action:</b> XES may not be used when XCF is running local mode. XES requires that the coupling facility be installed.</p>
10	xxxx10xx	<p><b>Equate Symbol:</b> IXLRSNCODECOMPERROR</p> <p><b>Meaning:</b> System error. XES processing failure. The state of the structure and the disposition of the request are unpredictable. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Save the reason code information, and contact the IBM support center.</p>

---

## Chapter 43. IXLCACHE REQUEST=REG\_NAMELIST

---

### Description

A REG\_NAMELIST request allows you to specify a list of data items for which you want to register interest. Each data item is represented by a registration block, mapped by the IXLYCRRB macro. The registration block contains information identifying the data item, its storage class, its vector index, whether to assign a directory entry, and whether the vector index is being reassigned to the current entry from some other entry in which interest will be deregistered. The registration blocks are contained in the storage area specified by BUFFER.

The system references the registration blocks by an index into the buffer and processes them sequentially beginning with the registration block identified by the first index (STARTINDEX) and ending with the data item identified by the last index (ENDINDEX).

The system returns state information for each processed data item represented by a registration block. The information is returned in the area specified by NSBAREA, and is mapped by the IXLYNSB macro.

A REG\_NAMELIST request can complete prematurely because of a failure processing one of the registration blocks in the list or because the request exceeds the time-out criteria for the coupling facility. (Time-out criteria is model-dependent.) When a request completes prematurely, the system returns an index value (CAARNLINDEX) in the answer area which you can use to restart the REG\_NAMELIST request.

A REG\_NAMELIST request can be issued only for cache structures allocated in a coupling facility of CFLEVEL=2 or higher. REG\_NAMELIST requests issued for a cache structure allocated in a coupling facility of CFLEVEL=0 or 1 will fail.

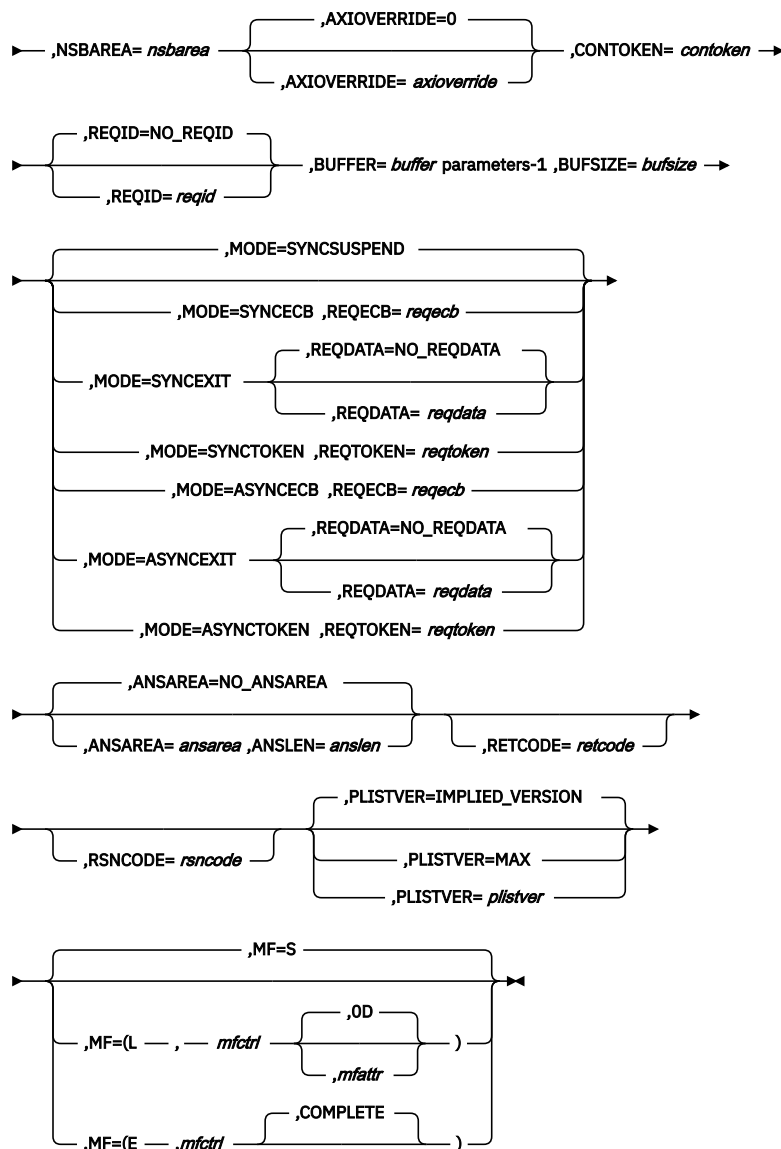
---

### Syntax Diagram

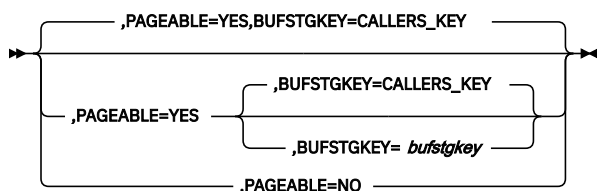
The syntax for the IXLCACHE REQUEST=REG\_NAMELIST is as follows:

## main diagram

➤ IXLCACHE — REQUEST=REG\_NAMELIST — ,STARTINDEX= *startindex* — ,ENDINDEX= *endindex* ➔



## parameters-1



**Note:** You must specify ANSAREA=*ansarea*, ANSLEN=*anslen* if you:

- Specify MODE=SYNCTOKEN or MODE=ASYNCTOKEN, or
- Specify AXIOVERRIDE=0 (or default to 0) and specified ASYNCXI=1 on the IXLCONN invocation when connecting to the cache structure.

## Parameter Descriptions

The parameter descriptions for REQUEST=REG\_NAMELIST are listed in alphabetical order. Default values are underlined:

### **REQUEST=REG\_NAMELIST**

Use this input parameter to specify that you want to register interest in a list of data items. Each data item is represented by a registration block, mapped by the IXLYCRRB macro, stored in the area specified by BUFFER.

### **,ANSAREA=NO\_ANSAREA**

### **,ANSAREA=ansarea**

Use this output parameter to specify an answer area to contain information returned from the request. The format of the answer area is described by the IXLYCAA mapping macro.

An asynchronous cross-invalidation sequence number (CAAASYNXISEQNUM) is returned from a request that initiates cross-invalidates of local caches asynchronously to the completion of the request.

See the [Chapter 29, “IXLAXISN,” on page 449](#) service for a description of how to use the returned CAAASYNXISEQNUM to determine when cross-invalidates of local caches associated with the request have completed.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an area (with a length of ANSLEN) where the answer area information returned from the request will be stored.

### **,ANSLEN=anslen**

Use this inppaameter to specify the size of the storage area specified by ANSAREA.

Use either CAALEVELOLEN, CAALEVEL1LEN or CAALEVEL2LEN of the IXLYCAA mapping macro to determine the minimum size of the answer area. The answer area length must be at least large enough to accommodate the level of the IXLYCAA mapping appropriate to the requested function.

- When the connection specified ASYNXCI=1 on the IXLCONN invocation and AXIOVERRIDE=0 was specified or defaulted to for the IXLCACHE request, the answer area length is a required parameter and must be a minimum value of CAALEVEL2LEN to contain a returned asynchronous cross-invalidation sequence number (CAAASYNXISEQNUM).
- When the value of PLISTVER is 4 or above, the minimum answer area length is CAALEVEL1LEN.
- When the value of PLISTVER is 0 - 3, the minimum answer area length is CAALEVELOLEN.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12 2 to ) of a 2-byte field that contains the length of the answer area (ANSAREA).

### **,AXIOVERRIDE=0**

### **,AXIOVERRIDE=axioverrride**

Use this input parameter to specify whether the asynchronous cross-invalidation control setting of IxlConnAsyncXiYes (1) for the connection identified by CONTOKEN should be overridden for this request. Valid values are 0 (IxlCacheAXiOverrideNo) or 1 (IxlCacheAXiOverrideYes).

The AXIOVERRIDE keyword is meaningful to processing only when the connection specified ASYNXCI=1 on the IXLCONN invocation when connecting to the cache structure and the cache structure is allocated in a CFLEVEL=23 or higher coupling facility.

A value of 0 (IxlCacheAXiOverrideNo) indicates that the asynchronous cross-invalidation control for the connection as specified on the IXLCONN invocation should be used for the request. Cross-invalidates against local caches for this request will preferentially be initiated asynchronously to the completion of the request when asynchronous cross-invalidations are supported by the coupling facility where the cache structure is allocated.

A value of 1 (IxlCacheAXiOverrideYes) indicates that the ASYNXCI specification of IxlConnAsyncXiYes (1) by the connector on the IXLCONN invocation should be overridden for this request only. Cross-invalidations generated by this request will be processed synchronously to the completion of the request.

Any value other than 0 or 1 for AXIOVERRIDE will have the same behavior as specifying a value of 0 (IxIcAchAXiOverrideNo).

If cross-invalidation against local caches for this request were initiated asynchronously to the completion of the request, an asynchronous cross-invalidation sequence number is returned in CAAASYNCXISEQNUM of the cache answer area (ANSAREA).

The asynchronous cross-invalidation sequence number can be used on a subsequent invocation of IXLAXISN to ensure that the asynchronous cross-invalidation associated with this request have completed.

When cross-invalidation is initiated synchronously to the completion of the request or no cross-invalidation occurred for the request, no asynchronous cross-invalidation sequence number is returned.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12 2 to ) of a one-byte input field that contains the value indicating whether the asynchronous cross-invalidation control setting of the connector should be overridden

### **,BUFFER=buffer**

Use this input parameter to specify a buffer area to contain an array of from 1 to 32 registration blocks that define the data items that are to be registered. The BUFFER area must be addressable from your primary address space or from your PASN access list.

The format of the registration block is described by mapping macro IXLYCRRB.

Only 31-bit addressable virtual storage areas (below 2GB) are supported by the BUFFER specification.

You can define the buffer size to be a total of up to 65536 bytes, but it should not be larger than what you actually require to hold the maximum number of registration blocks, subject to the other buffer length requirements stated below.

Other requirements depend on the size you select:

- If you specify a buffer size of less than or equal to 4096 bytes, you must ensure that the buffer:
  - Is 256, 512, 1024, 2048, or 4096 bytes.
  - Starts on a 256-byte boundary.
  - Does not cross a 4096-byte boundary.
  - Does not start below storage address 512.
- If you specify a buffer size of greater than 4096 bytes, you must ensure that the buffer:
  - Is a multiple of 4096 bytes.
  - Is less than or equal to 65536 bytes.
  - Starts on a 4096-byte boundary.
  - Does not start below storage address 512.

See the BUFSIZE parameter description for defining the size of the buffer. See [z/OS MVS Programming: Sysplex Services Guide](#) for more information on buffers.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an area (with a length of BUFSIZE) to contain the registration blocks.

### **,BUFSIZE=bufsize**

Use this input parameter to specify the size of the BUFFER area. See the BUFFER parameter description for valid buffer sizes.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a fullword field that contains the size of the buffer (BUFFER) in bytes.

### **,BUFSTGKEY=CALLERS\_KEY**

### **,BUFSTGKEY=bufstgkey**

Use this input parameter to specify a storage key that you define and use when referencing the buffer specified by BUFFER.



If you do not specify BUFSTGKEY, or if you specify BUFSTGKEY=CALLERS\_KEY, all references to the buffer are in the caller's PSW key.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 1-byte field that contains the storage key in the format kkkkxxxx, where kkkk is the key and xxxx is ignored.

**,CONTO=conken**

Use this input parameter to specify the connect token that was returned by the IXLCONN service in the IXLCONN answer area, which is mapped which is by IXLYCONA. The connect token uniquely identifies your connection to the cache structure, and must be specified on each IXLCACHE invocation.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field that contains the connect token.

**,ENDINDEX=endindex**

Use this input parameter to specify the ending index for registration block processing. The index value must be greater than or equal to the value specified for STARTINDEX. Valid values are 1 - 32.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 2-byte field containing the ending index value.

**,MF=S**

**,MF=(L,mfctrl)**

**,MF=(L,mfctrl,mfattr)**

**,MF=(L,mfctrl,0D)**

**,MF=(M,mfctrl)**

**,MF=(M,mfctrl,COMPLETE)**

**,MF=(M,mfctrl,NOCHECK)**

**,MF=(E,mfctrl)**

**,MF=(E,mfctrl,COMPLETE)**

**,MF=(E,mfctrl,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

**,mfctrl**

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

**,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

**,COMPLETE**

**,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

**COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if `SMILE=var` were an optional parameter and the default is `SMILE=NO_SMILE` then it would not be documented. However, if the default was `SMILE=-`, then it would be documented because a value would be the default.

**NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

**,MODE=SYNCSUSPEND**

**,MODE=SYNCECB**

**,MODE=SYNCEXIT**

**,MODE=SYNCTOKEN**

**,MODE=ASYNCECB**

**,MODE=ASYNCEXIT**

**,MODE=ASYNCTOKEN**

Use this input parameter to specify:

- Whether the request is to be performed synchronously or asynchronously
- How you want to be notified of request completion if the request is processed asynchronously.

See *z/OS MVS Programming: Sysplex Services Guide* for more information on understanding synchronous and asynchronous cache operations.

**SYNCSUSPEND**

The request processes synchronously. If necessary, the request is suspended until it can complete synchronously. To use this option, your program must be enabled for I/O and external interrupts.

**SYNCECB**

The request processes synchronously if possible. If the request processes asynchronously, the ECB specified by REQECB is posted when the request completes.

**SYNCEXIT**

The request processes synchronously if possible. If the request processes asynchronously, your complete exit is given control when the request completes.

**SYNCTOKEN**

The request processes synchronously if possible. If the request processes asynchronously, an asynchronous request token is returned to the area specified by REQTOKEN. Use the returned request token on the IXLFCOMP macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when `MODE=SYNCTOKEN` is specified.

**ASYNCECB**

The request processes asynchronously. The ECB specified by REQECB is posted when the request completes.

**ASYNCEXIT**

The request processes asynchronously. Your complete exit is given control when the request completes.

**ASYNCTOKEN**

The request processes asynchronously. An asynchronous request token is returned to the area specified by REQTOKEN. Use the returned request token on the IXLFCOMP macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when `MODE=ASYNCTOKEN` is specified.

**,NSBAREA=nsbarea**

Use this output parameter to specify a storage area to contain name state information after the system has processed the registration blocks specified in the BUFFER area. The NSBAREA area must be addressable from your primary address space or from your PASN access list.

The format of the NSBAREA area is described by mapping macro IXLYNSB.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 256-byte area to contain the name state information.

**,PAGEABLE=YES****,PAGEABLE=NO**

Use this input parameter to identify whether the storage area specified by BUFFER is in pageable or potentially pageable storage, and, if the area could become pageable, designate responsibility for ensuring that the area remains page-fixed for the duration of the request.

**PAGEABLE=YES**

Specify this option when you are using any one of the following types of storage and you want the system to ensure that the storage becomes or remains page-fixed for the duration of the request:

- Pageable storage
- Disabled reference (DREF) storage
- Page-fixed storage that could become pageable before the request completes processing.

**PAGEABLE=NO**

Specify this option when using any one of the following types of storage:

- Fixed storage
- Page-fixed storage that you will ensure remains page-fixed for the duration of the request

If you specify PAGEABLE=NO and the request processes asynchronously, you must keep the storage fixed until you are sure the request has completed. (The MODE value specified determines how you are notified of request completion.) See [z/OS MVS Programming: Sysplex Services Guide](#) for these guidelines.

**,PLISTVER=IMPLIED\_VERSION****,PLISTVER=MAX****,PLISTVER=plistver**

Use this input parameter to specify the version of the macro. See “[Understanding IXLCACHE Version Support](#)” on page 461 for a description of the options available with PLISTVER.

**,REQDATA=NO\_REQDATA****,REQDATA=reqdata**

Use this input parameter with MODE=SYNCEXIT or MODE=ASYNCEXIT to pass any data you choose to the complete exit. The exit will get control only if the request is processed asynchronously.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the data to be passed to the complete exit.

**,REQECB=reqecb**

Use this output parameter with either MODE=SYNCECB or MODE=ASYNCECB to specify the address of an ECB, which is to be posted when the request completes if the request was processed asynchronously.

Before coding REQECB, you must ensure that:

- You initialize the ECB before you issue the request.
- The ECB resides in either common storage or the home address space where IXLCONN was issued.
- Any tasks that wait for the ECB to be posted reside in the home address space where IXLCONN was issued.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that contains the address of the ECB to be posted when the request completes. The ECB must be aligned on a fullword boundary.

**,REQID=NO\_REQID****,REQID=reqid**

Use this input parameter to specify a user-defined request identifier to be associated with the request. You can specify this request identifier on the IXLPURGE macro to cancel a request that has not yet been processed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the user-defined request identifier.

**,REQTOKEN=reqtoken**

Use this output parameter with either MODE=SYNCTOKEN or MODE=ASYNCTOKEN to specify the address of a storage area to receive the request token that is returned when the request will be processed asynchronously. This token, which uniquely identifies the request, must be used as input to the IXLFCOMP macro, which you use to determine if the request has completed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field where the system will put the request token.

**,RETCODE=retcode**

Use this output parameter to specify a field to contain the return code. (The return code is also returned in register 15.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the return code when the request has completed.

**,RSNCODE=rsncode**

Use this output parameter to specify a field to contain the reason code returned, if applicable. (The reason code is also returned in register 0.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the reason code (if any) when the request has completed.

**,STARTINDEX=startindex**

Use this input parameter to specify the starting index for registration block processing. Valid STARTINDEX values are from 1 to the value of ENDINDEX. The first registration block has index number 1.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 2-byte field containing the starting index value.

## ABEND Codes

---

Abend X'026' (See [z/OS MVS Programming: Sysplex Services Guide](#) for more information on this abend.)

## Return and Reason Codes

---

When the system returns control to your program:

- GPR 15 (and RETCODE, if specified) contains a return code.
- If the return code is not zero, GPR 0 (and RSNCODE, if specified) contains a reason code.

**Note:** The return and reason codes will also be put into the answer area (mapped by IXLYCAA), if there is one, when the request completes.

The IXLYCON macro provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

**0**

IXLRETCODEOK

**4**

IXLRETCODEWARNING

**8**

IXLRETCODEPARMERROR

**C**

IXLRETCODEENVERROR

**10**

IXLRETCODECOMPERROR

The following table contains hexadecimal return and reason codes, the equate symbols associated with each reason code, and the meaning and suggested action for each return and reason code.

Table 41. Return and Reason Codes for the IXLCACHE REQUEST=REG_NAMELIST Macro		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0	None.	<p><b>Meaning:</b> If you specified a MODE value of ASYNCECB, ASYNCEXIT, or ASYNCTOKEN, the request has been successfully initiated. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed. For any other MODE value, the request has successfully completed.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=ASYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=ASYNCEXIT, the connection's complete exit will be given control when the request has completed.</li> <li>• If you specified MODE=ASYNCTOKEN, use REQTOKEN when calling IXLFComp to determine when the request has completed.</li> </ul>
4	xxxx0402	<p><b>Equate Symbol:</b> IXLRSNCODEASYNCH</p> <p><b>Meaning:</b> The request could not be processed synchronously. It will be processed asynchronously. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed.</p> <ul style="list-style-type: none"> <li>• If you specified MODE=SYNCECB, the ECB specified in REQECB will be posted when the request has finished.</li> <li>• If you specified MODE=SYNCEXIT, the connection's complete exit will be given control when the request has finished.</li> <li>• If you specified MODE=SYNCTOKEN, the REQTOKEN contains an asynchronous request token that may be used on the IXLFComp macro to determine when the request has completed.</li> </ul> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=SYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=SYNCEXIT, the connection's completion exit will be given control when the request has completed.</li> <li>• If you specified MODE=SYNCTOKEN, use REQTOKEN when calling IXLFComp to determine when the request has completed.</li> </ul>

Table 41. Return and Reason Codes for the IXLCACHE REQUEST=REG\_NAMELIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	xxxx0409	<p><b>Equate Symbol:</b> IXLRSNCODETIMEOUT</p> <p><b>Meaning:</b> The request has completed prematurely because the model-dependent time-out criteria of the coupling facility has been exceeded. The index of the next registration block to be processed has been returned in the answer area (field CAARNLINDEX).</p> <p><b>Action:</b> Reissue the request.</p> <p>Be sure to process the information returned from this request before reissuing the request. The data returned from this request will be overwritten if you specify the same buffer address. Continue to reissue the request until the return code indicates that all processing has completed.</p> <p>The entries that were processed are indexed from STARTINDEX to CAARNLINDEX-1.</p> <p>To restart the request, update STARTINDEX with the value of CAARNLINDEX, the index of the next name element to be processed. For more information about premature request completion, see <a href="#">z/OS MVS Programming: Sysplex Services Guide</a>.</p>
8	xxxx0801	<p><b>Equate Symbol:</b> IXLRSNCODEBADPARMLIST</p> <p><b>Meaning:</b> The parameter list (mfctrl) for this request is not addressable. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the parameter list address.</li> <li>• The parameter list must be addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are issuing this macro while disabled, the parameter list must reside in either page-fixed or DREF storage.</li> <li>• If you are issuing this macro in AR-mode and you specified the parameter list address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are issuing this macro in AR-mode, SYSSTATE ASCENV=AR must be issued before this macro.</li> </ul>
8	xxxx0804	<p><b>Equate Symbol:</b> IXLRSNCODEBADVERSION#</p> <p><b>Meaning:</b> The version number in the macro parameter list is not compatible with the level of XES currently being used. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that your program did not overlay the parameter list storage.</li> <li>• Verify that your program was assembled with the correct macro library for the release of MVS on which your program is running.</li> <li>• Verify that your program is running on an MVS system that supports the version of the macro you are using.</li> </ul>

Table 41. Return and Reason Codes for the IXLCACHE REQUEST=REG\_NAMELIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx080A	<p><b>Equate Symbol:</b> IXLRNCODEBADCONTOKEN</p> <p><b>Meaning:</b> Program error. The specified connect token (CONTOKEN) was not valid for one of the following reasons:</p> <ol style="list-style-type: none"> <li>1. The user with the connection identifier represented by the token has disconnected from the structure.</li> <li>2. The connector's task (the task that issued IXLCONN) ended.</li> <li>3. The specified token is not the token that was returned from IXLCONN.</li> <li>4. The request was issued from an address space other than the address space in which IXLCONN was issued.</li> <li>5. The connect token was invalidated during rebuild.</li> <li>6. The connect token was invalidated by XES.</li> </ol> <p><b>Note:</b> The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Verify that the CONTOKEN value specified was valid and for the correct structure. The numbers on the following actions correspond to the numbers above.</p> <ol style="list-style-type: none"> <li>1. Discontinue use. Perform recovery and cleanup for the structure and check protocol for use of structure to determine why you did not know of the disconnection.</li> <li>2. Discontinue use. Perform recovery and cleanup for the structure and check protocol for use of structure to determine why you did not know that the task ended.</li> <li>3. Issue IXLCONN to connect to the structure. The contoken is returned in the CONACONTOKEN field of the answer area specified on the IXLCONN request.</li> <li>4. Issue your request from the same address space the IXLCONN was issued.</li> <li>5. Participate in the rebuild. When it is complete, try again.</li> <li>6. Discontinue use. Perform recovery and cleanup for the structure and check protocol for use of structure.</li> </ol> <p>You may want to issue IXCQUERY to get more information about the structure.</p>
8	xxxx0819	<p><b>Equate Symbol:</b> IXLRNCODEBADVECTOROP</p> <p><b>Meaning:</b> The vector index in the registration block indexed by CAARNLINDEX is not valid. None of the registration blocks have been processed. All vector indexes for all registration blocks in the buffer have been set to indicate that the local buffer is not valid.</p> <p><b>Action:</b> Correct the vector index value and reissue the REG_NAMELIST request with the same STARTINDEX and ENDINDEX values.</p>

Table 41. Return and Reason Codes for the IXLCACHE REQUEST=REG\_NAMELIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0824	<p><b>Equate Symbol:</b> IXLSRNCODEWRONGSTRTYPE</p> <p><b>Meaning:</b> The connect token specified by CONTOKEN is not to a cache structure.</p> <p><b>Action:</b> Verify the connect token for this cache structure.</p> <p><b>Note:</b> The connect token can be found in the IXLCONN answer area. Verify the value specified on the TYPE= parameter of the IXLCONN request for this structure.</p>
8	xxxx082D	<p><b>Equate Symbol:</b> IXLSRNCODEBADSTGCLASS</p> <p><b>Meaning:</b> Program error. A REG_NAMELIST registration block specified a storage class value that exceeded the maximum defined storage class for the structure. CAARNLINDEX contains the index of the registration block that contained the storage class value in error. All registration blocks preceding this block were processed.</p> <p><b>Action:</b> Correct the storage class value in the registration block indexed by CAARNLINDEX and reissue the REG_NAMELIST request starting with that registration block.</p>
8	xxxx0833	<p><b>Equate Symbol:</b> IXLSRNCODEBADPGBLATTR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER is specified as being pageable (PAGEABLE=YES) but is not.</p> <p><b>Action:</b> Change the buffer area to pageable storage, or specify PAGEABLE=NO. See the PAGEABLE parameter description for specification instructions.</p>
8	xxxx0834	<p><b>Equate Symbol:</b> IXLSRNCODEBADNONPGBLATTR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER is specified as being nonpageable (PAGEABLE=NO) but is either pageable or not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The type of storage in the buffer area corresponds to the value you specified on the PAGEABLE parameter. See the PAGEABLE parameter description for specification instructions.</li> <li>• The correct buffer address was used.</li> <li>• The buffer area was not previously freed.</li> <li>• If the caller is running in AR-mode and the BUFFER parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> </ul>



Table 41. Return and Reason Codes for the IXLCACHE REQUEST=REG\_NAMELIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0835	<p><b>Equate Symbol:</b> IXLSRNCODEBADDATAADDR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER is not addressable. All bits in the local cache vector may be reset to overindicate that the data items are not valid.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The correct buffer address was used for BUFFER.</li> <li>• The buffer area was not previously freed.</li> <li>• The buffer area was allocated in a storage key that matches the key specified using the BUFSTGKEY parameter, or, if BUFSTGKEY is omitted, the caller's PSW key.</li> <li>• The buffer area is addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If the caller is running in AR-mode and the BUFFER parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If the caller is running in AR-mode, SYSSTATE ASCENV=AR must be specified before issuing this macro.</li> </ul>
8	xxxx0838	<p><b>Equate Symbol:</b> IXLSRNCODEBADANSAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by ANSAREA is not addressable.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the ANSAREA address.</li> <li>• ANSAREA must be addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are calling IXLCACHE while disabled, ANSAREA must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLCACHE in AR-mode and you specified the ANSAREA address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are calling IXLCACHE in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLCACHE macro.</li> </ul>

Table 41. Return and Reason Codes for the IXLCACHE REQUEST=REG\_NAMELIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0839	<p><b>Equate Symbol:</b> IXLRNCODEBADREQTOKENAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by REQTOKEN is not addressable.</p> <p><b>Action:</b> Ensure that the request token area specified by REQTOKEN is valid:</p> <ul style="list-style-type: none"> <li>• Verify the REQTOKEN address.</li> <li>• If you are calling IXLCACHE while disabled, REQTOKEN must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLCACHE in AR-mode and you specified the REQTOKEN address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are calling IXLCACHE in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLCACHE macro.</li> </ul>
8	xxxx083D	<p><b>Equate Symbol:</b> IXLRNCODEBADANSLEN</p> <p><b>Meaning:</b> Program error. The length of the answer area, as specified by ANSLEN, is not sufficient to contain answer area information. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Increase the size of the answer area provided for the request, and rerun your program. Depending on the macro version number, specify the length of the answer area as follows:</p> <ul style="list-style-type: none"> <li>• When the connection specified ASYNCXI=1 on the IXLCONN invocation and AXIOVERRIDE=0 was specified or defaulted to for the IXLCACHE request, the answer area length must be a minimum value of CAALEVEL2LEN</li> <li>• When the value of the macro version number (PLISTVER) is 4 or more, the minimum answer area length is CAALEVEL1LEN.</li> <li>• When the value of the macro version number (PLISTVER) is 0-3, the minimum answer area length is CAALEVEL0LEN.</li> </ul>
8	xxxx0851	<p><b>Equate Symbol:</b> IXLRNCODENOSUSPENDISABLE</p> <p><b>Meaning:</b> Program error. The request failed because MODE=SYNCSUSPEND was specified, but the caller is disabled and cannot be suspended.</p> <p><b>Action:</b> Either specify another MODE value or become enabled (release the CPU lock); then reissue the request.</p>
8	xxxx0865	<p><b>Equate Symbol:</b> IXLRNCODEBADBUFSPEC</p> <p><b>Meaning:</b> Program error. There is an error in the buffer specification.</p> <p><b>Action:</b> Check the following:</p> <ul style="list-style-type: none"> <li>• Requirements for BUFFER and BUFSIZE.</li> <li>• Buffer boundaries.</li> </ul>

Table 41. Return and Reason Codes for the IXLCACHE REQUEST=REG\_NAMELIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0866	<p><b>Equate Symbol:</b> IXLRNCODEBADBUFKEY</p> <p><b>Meaning:</b> Program error. The buffer storage key specified by BUFSTGKEY is incorrect or BUFSTGKEY was not specified and the PSW key does not match the key of the buffers.</p> <p>The data cannot be stored in the specified buffer area.</p> <p><b>Action:</b> Check the following:</p> <ul style="list-style-type: none"> <li>• Determine if the key of the storage being used for the buffers is different from the PSW key.</li> <li>• If BUFSTGKEY was specified, verify that the key was put in the correct bits (see the explanation of this parameter for more information).</li> <li>• If you are calling IXLCACHE in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLCACHE macro.</li> </ul>
8	xxxx0874	<p><b>Equate Symbol:</b> IXLRNCODEBADRNINDEX</p> <p><b>Meaning:</b> Program error. Either the value specified for either STARTINDEX or ENDINDEX is not valid or the size of the buffer specified by BUFFER is smaller than required based on the value of ENDINDEX.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The values specified by STARTINDEX and ENDINDEX are in the range of 1 to 32.</li> <li>• The value specified by ENDINDEX is greater than or equal to the value of STARTINDEX.</li> <li>• The value specified by ENDINDEX does not imply a larger BUFFER size than was actually specified on the REG_NAMELIST request.</li> </ul>
8	xxxx0875	<p><b>Equate Symbol:</b> IXLRNCODEBADNSBAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by NSBAREA is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The address passed in NSBAREA is valid.</li> <li>• The NSBAREA area was not previously freed.</li> <li>• The NSBAREA area is addressable from the caller's primary address space or from the caller's PASN access list.</li> <li>• If the caller is running in AR-mode and NSBAREA was specified using explicit register notation, ensure that the corresponding access register was updated appropriately.</li> <li>• If the caller is disabled, then NSBAREA must reside in either nonpageable or disabled reference storage.</li> <li>• If your program is running in AR mode, ensure that you specified SYSSTATE ASCENV=AR before issuing the IXLCACHE macro.</li> </ul>

Table 41. Return and Reason Codes for the IXLCACHE REQUEST=REG\_NAMELIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx08AD	<p><b>Equate Symbol:</b> IXLRNCOBDEBADHIGHSHAREDVIRT</p> <p><b>Meaning:</b> Program error. The request specified a high shared virtual storage area (above 2GB).</p> <p><b>Action:</b> None required.</p>
C	xxxx0C06	<p><b>Equate Symbol:</b> IXLRNCOBDENOCONN</p> <p><b>Meaning:</b> Environmental error. No connectivity to the cache structure exists. This may occur due to operator commands such as VARY PATH,OFFLINE or CONFIG CHP,OFFLINE or hardware errors such as facility or path failures. The CONTOKEN is invalidated. The request fails.</p> <p><b>Action:</b> Either disconnect from the structure (using IXLDISC) or initiate a rebuild of the structure, if allowed (using IXLREBLD).</p>
8	xxxx08B9	<p><b>Equate Symbol:</b> IXLRNCOBDENOANSAREA</p> <p><b>Meaning:</b> An answer area was not specified when one is required. The requested service determined that conditions exist that require an ANSAREA to complete the request.</p> <p><b>Action:</b> Provide an answer area (ANSAREA) and answer area length (ANSLEN) on the IXLCACHE macro invocation for the request. ANSAREA is required when the connection specified ASYNCCI=1 on the IXLCONN invocation when connecting to the cache structure, AXIOVERRIDE=0 was specified or defaulted to for the IXLCACHE request and the request is one of the following:</p> <ul style="list-style-type: none"> <li>• CROSS_INVALID</li> <li>• CROSS_INVALLIST</li> <li>• DELETE_NAME</li> <li>• DELETE_NAMELIST</li> <li>• READ_DATA</li> <li>• REG_NAMELIST</li> <li>• WRITE_DATA</li> <li>• WRITE_DATA_LIST</li> </ul>

Table 41. Return and Reason Codes for the IXLCACHE REQUEST=REG\_NAMELIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C13	<p><b>Equate Symbol:</b> IXLRSNCODEREQPURGED</p> <p><b>Meaning:</b> Environmental error. The request was purged prior to completion of the request. Possible reasons include:</p> <ul style="list-style-type: none"> <li>• The connector failed.</li> <li>• The connector disconnected.</li> <li>• The requestor failed.</li> <li>• The request was purged by IXLPURGE.</li> <li>• Requests were purged when the connector provided an IXLEERSP response for the Rebuild Quiesce, Rebuild Stop, or Rebuild Cleanup event.</li> <li>• The connector invoked IXLREBLD REQUEST=COMPLETE.</li> <li>• The secondary address space was no longer valid.</li> </ul> <p><b>Action:</b> None if this is expected. Otherwise, determine why the connector failed.</p>
C	xxxx0C14	<p><b>Equate Symbol:</b> IXLRSNCODESTATUSUNKNOWN</p> <p><b>Meaning:</b> Environmental error. The IXLCACHE request has completed, but the final disposition of the request cannot be determined. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Verify the validity of your data by comparing the expected results with what is in the coupling facility.</p>
C	xxxx0C17	<p><b>Equate Symbol:</b> IXLRSNCODESTRFULL</p> <p><b>Meaning:</b> Environmental error. Allocation of a directory entry was necessary, but was unavailable or could not be reclaimed. In the answer area, CAASTGCLFULL contains the storage class from which the reclaiming operation failed. CAARNLINDEX contains the index of the registration block that was being processed when the error occurred. All prior registration blocks were processed.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Determine if any data items may be cast-out to make room for this item.</li> <li>• Check your usage of storage classes to see if some data items can be moved to a different storage class (preferably with a lower priority) so some entries in the structure can be freed.</li> <li>• Determine if a rebuild or an alter of the structure is necessary to make room for more data entries/items.</li> </ul> <p>After correcting the error, restart the REG_NAMELIST request starting with the registration block indexed by CAARNLINDEX.</p>

Table 41. Return and Reason Codes for the IXLCACHE REQUEST=REG\_NAMELIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C25	<b>Equate Symbol:</b> IXLRNCODESTRFAILURE <b>Meaning:</b> Environmental error. The cache structure failed prior to completion of the request. <b>Action:</b> Attempt to rebuild the structure using IXLREBLD, or disconnect from the structure using IXLDISC.
C	xxxx0C68	<b>Equate Symbol:</b> IXLRNCODEBADREQCFLEVEL <b>Meaning:</b> Environmental error. The request type is not permitted for the level of coupling facility in which the target structure is allocated. <b>Action:</b> Either continue processing using single READ_DATA requests to perform the function of the REG_NAMELIST request or disconnect from the structure (using IXLDISC) and request that the installation provide a coupling facility of the correct CFLEVEL (CFLEVEL=2 or higher).
C	xxxx0CA0	<b>Equate Symbol:</b> IXLRNCODEQUIESCEDSUSPENDFAIL <b>Meaning:</b> Environmental error. The request is failed because the structure is quiesced for a system-managed process and SUSPEND=FAIL is specified on the IXLCONN. <b>Action:</b> None, if this is expected.
C	xxxxFFFF	<b>Equate Symbol:</b> IXLRNCODENOTAVAILABLE <b>Meaning:</b> Environmental error. XES functions are not available. The coupling facility hardware might not be present. <b>Action:</b> XES may not be used when XCF is running local mode. XES requires that the coupling facility be installed.
10	xxxx10xx	<b>Equate Symbol:</b> IXLRNCODECOMPERROR <b>Meaning:</b> System error. XES processing failure. The state of the structure and the disposition of the request are unpredictable. The answer area (ANSAREA) fields do not contain valid information. <b>Action:</b> Save the reason code information, and contact the IBM support center.

---

## Chapter 44. IXLCACHE REQUEST=RESET\_REFBIT

### Description

---

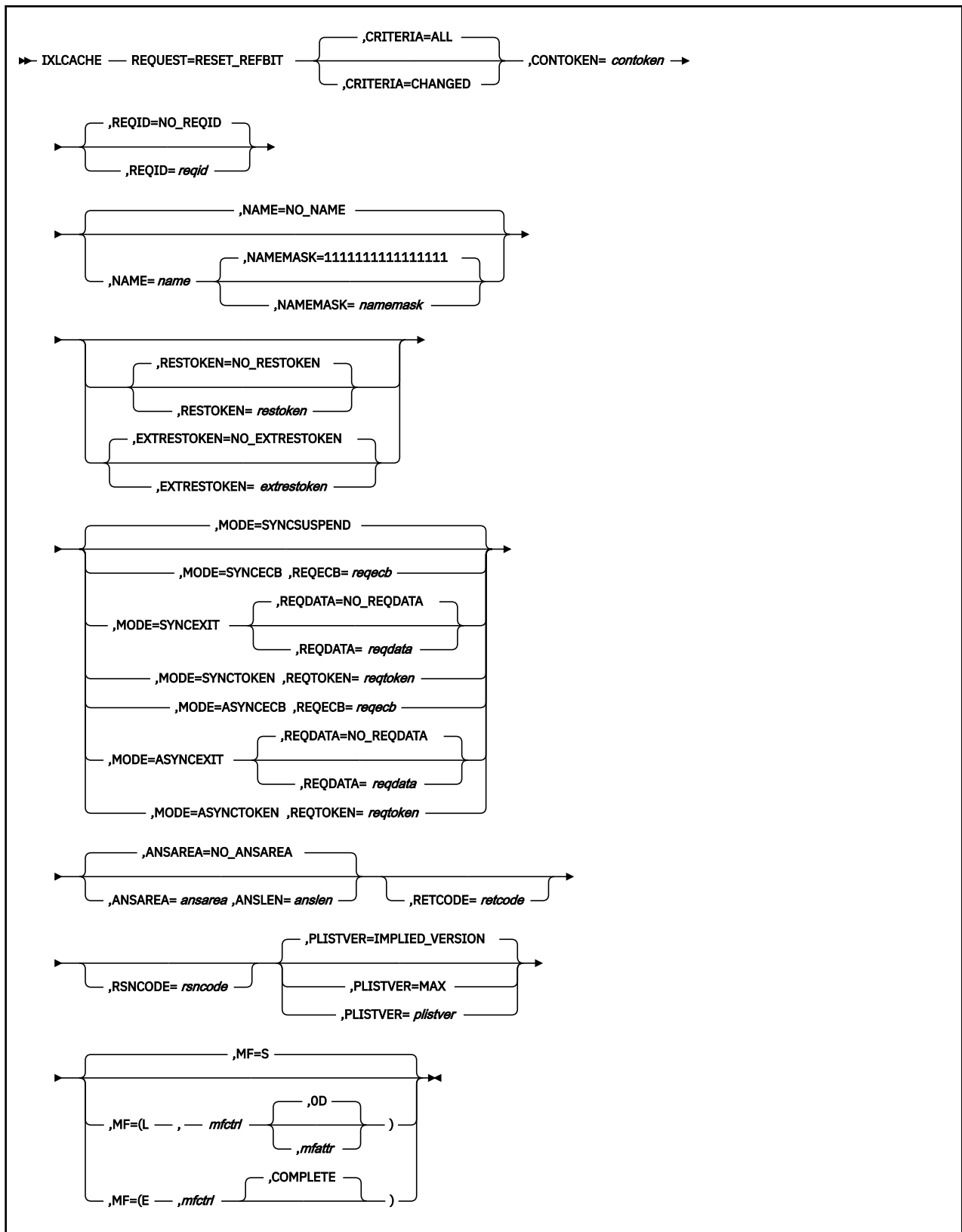
A RESET\_REFBIT request allows you to reset the reference bit in the directory entries for the specified cached data items to indicate that the data items were not recently referenced. You can specify that all directory entries in the structure are processed (CRITERIA=ALL) or only those that contain changed or locked-for-cast-out data (CRITERIA=CHANGED). You can further filter the selection of entries to be processed by specifying NAME and NAMEMASK. The total number of processed directory entries and the number of these directory entries for which the reference bit was actually reset are both returned in the answer area (ANSAREA).

If the request exceeds the model-dependent time-out criteria before processing completes, either a restart token (RESTOKEN) or an extended restart token (EXTRESTOKEN) is returned in the answer area. The token can be specified on the next RESET\_REFBIT request to resume processing with the next data item to be processed. Resumed requests are processed identically whether using the RESTOKEN or EXTRESTOKEN to specify the starting location.

### Syntax Diagram

---

The syntax diagram for IXLCACHE REQUEST=RESET\_REFBIT is as follows:



**Note:** When `MODE=SYNCTOKEN` or `MODE=ASYNCTOKEN` is specified, `ANSAREA=ansarea,ANSLEN=anslen` is required.



## Parameter Descriptions

The parameter descriptions for REQUEST=RESET\_REFBIT are listed in alphabetical order. Default values are underlined:

### **REQUEST=RESET\_REFBIT**

Use this input parameter to specify that the directory entry reference bit for the entries specified by CRITERIA, NAME, and NAMEMASK be reset to indicate that entries are unreferenced. The total number of directory entries processed and the number of these directory entries that had their reference bit reset are returned in the answer area.

### **,ANSAREA=NO\_ANSAREA**

### **,ANSAREA=ansarea**

Use this output parameter to specify an answer area to contain information returned from the request. The format of the answer area is described by the IXLYCAA mapping macro. The following information is returned in the answer area:

- The number of directory entries processed by the request (field CAADIRCOUNT).
- The number of processed directory entries for which the reference bit was reset (field CAAREFCOUNT).
- See the RESTOKEN and EXTRESTOKEN parameters for a description of the restart token.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an area (with a length of ANSLEN) where the information returned from the request is to be put.

### **,ANSLEN=anslen**

Use this inppaameter to specify the size of the storage area specified by ANSAREA.

Use either CAALEVEL0LEN, CAALEVEL1LEN or CAALEVEL2LEN of the IXLYCAA mapping macro to determine the minimum size of the answer area. The answer area length must be at least large enough to accommodate the level of the IXLYCAA mapping appropriate to the requested function.

- When the connection specified ASYNCXI=1 on the IXLCONN invocation and AXIOVERRIDE=0 was specified or defaulted to for the IXLCACHE request, the answer area length is a required parameter and must be a minimum value of CAALEVEL2LEN to contain a returned asynchronous cross-invalidation sequence number (CAAASYNCXISEQNUM).
- When the value of PLISTVER is 4 or above, the minimum answer area length is CAALEVEL1LEN.
- When the value of PLISTVER is 0 - 3, the minimum answer area length is CAALEVEL0LEN.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12 2 to ) of a 2-byte field that contains the length of the answer area (ANSAREA).

### **,CONTO=conken**

Use this input parameter to specify the connect token that was returned by the IXLCONN service in the IXLCONN answer area, which is mapped which is by IXLYCONA. The connect token uniquely identifies your connection to the cache structure, and must be specified on each IXLCACHE invocation.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field that contains the connect token.

### **,CRITERIA=ALL**

### **,CRITERIA=CHANGED**

Use this input parameter to specify the criteria for selecting which directory entries will be processed. (You can further limit the selection by specifying NAME with or without NAMEMASK.)

#### **ALL**

The reference bit will be reset in all allocated directory entries.

#### **CHANGED**

The reference bit will be reset in only those directory entries that contain changed or locked-for-cast-out data.

**,EXTRESTOKEN=NO\_EXTRESTOKEN****,EXTRESTOKEN=extrestoken**

Use this input parameter to specify an extended restart token that can be used to resume processing of a RESET\_REFBIT request that exceeded the model-dependent time-out criteria. The extended restart token is returned in the answer area (field CAAEXTRESTOKEN), and should be specified on the next RESET\_REFBIT request to resume processing with the next data item to be processed.

If the request does not exceed the model-dependent time-out criteria, the extended restart token will not be provided.

**Note:**

1. Specifying an extended restart token of all zeros causes cache services to treat all of the entries as unprocessed.
2. Do not specify an extended restart token other than the one returned in the answer area or one set to all zeros, because results will be unpredictable.
3. Specifying an extended restart token requires that the length of the answer area be at least the length of CAALEVEL1LEN.

Requestors that specify IXLCONN ALLOWAUTO=YES must use the extended restart token. Requestors that specify or default to IXLCONN ALLOWAUTO=NO must use the standard restart token (RESTOKEN).

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-byte field that contains the extended restart token.

**,MF=S****,MF=(L,mfctrl)****,MF=(L,mfctrl,mfattr)****,MF=(L,mfctrl,0D)****,MF=(M,mfctrl)****,MF=(M,mfctrl,COMPLETE)****,MF=(M,mfctrl,NOCHECK)****,MF=(E,mfctrl)****,MF=(E,mfctrl,COMPLETE)****,MF=(E,mfctrl,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

**,mfctrl**

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

**,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

**,COMPLETE****,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

**COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if SMILE=*var* were an optional parameter and the default is SMILE=NO\_SMILE then it would not be documented. However, if the default was SMILE=:), then it would be documented because a value would be the default.

**NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

**,MODE=SYNCSUSPEND****,MODE=SYNCECB****,MODE=SYNCEXIT****,MODE=SYNCTOKEN****,MODE=ASYNCECB****,MODE=ASYNCEXIT****,MODE=ASYNCTOKEN**

Use this input parameter to specify:

- Whether the request is to be performed synchronously or asynchronously
- How you want to be notified of request completion if the request is processed asynchronously.

See *z/OS MVS Programming: Sysplex Services Guide* for more information on understanding synchronous and asynchronous cache operations.

**SYNCSUSPEND**

The request processes synchronously. If necessary, the request is suspended until it can complete synchronously. To use this option, your program must be enabled for I/O and external interrupts.

**SYNCECB**

The request processes synchronously if possible. If the request processes asynchronously, the ECB specified by REQECB is posted when the request completes.

**SYNCEXIT**

The request processes synchronously if possible. If the request processes asynchronously, your complete exit is given control when the request completes.

**SYNCTOKEN**

The request processes synchronously if possible. If the request processes asynchronously, an asynchronous request token is returned to the area specified by REQTOKEN. Use the returned request token on the IXLFCOMP macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=SYNCTOKEN is specified.

**ASYNCECB**

The request processes asynchronously. The ECB specified by REQECB is posted when the request completes.

**ASYNCEXIT**

The request processes asynchronously. Your complete exit is given control when the request completes.

**ASYNCTOKEN**

The request processes asynchronously. An asynchronous request token is returned to the area specified by REQTOKEN. Use the returned request token on the IXLFCOMP macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=ASYNCTOKEN is specified.

**,NAME=NO\_NAME**

**,NAME=name**

Use this input parameter to filter by name the data items for which the associated directory entry reference bit will be reset. You may use this parameter along with the NAMEMASK parameter to selectively filter out data items. See the NAMEMASK parameter for more information on this option.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-byte field that contains the name of the data item.

**,NAMEMASK=1111111111111111**

**,NAMEMASK=namemask**

Use this input parameter to specify which characters in the name specified by NAME are to be used in selecting data items for processing. This parameter allows you to select multiple data items based on common characters in NAME.

The position of each bit in NAMEMASK corresponds to the same relative character position in NAME. A one indicates that the corresponding letter should be used in selecting entries; a zero indicates that the corresponding letter should not be used.

Specifying a name mask with all zeros causes all names to be selected for processing. Specifying a name mask with all ones causes only the name specified by NAME to be selected.

For more information on how NAMEMASK may be used, see [z/OS MVS Programming: Sysplex Services Guide](#).

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 2-byte field that contains the bit-mask for the name specified on the NAME keyword.

**,PLISTVER=IMPLIED\_VERSION**

**,PLISTVER=MAX**

**,PLISTVER=plistver**

Use this input parameter to specify the version of the macro. See “[Understanding IXLCACHE Version Support](#)” on page 461 for a description of the options available with PLISTVER.

**,REQDATA=NO\_REQDATA**

**,REQDATA=reqdata**

Use this input parameter with MODE=SYNCEXIT or MODE=ASYNCEXIT to pass any data you choose to the complete exit. The exit will get control only if the request is processed asynchronously.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the data to be passed to the complete exit.

**,REQECB=reqecb**

Use this output parameter with either MODE=SYNCECB or MODE=ASYNCECB to specify the address of an ECB, which is to be posted when the request completes if the request was processed asynchronously.

Before coding REQECB, you must ensure that:

- You initialize the ECB before you issue the request.
- The ECB resides in either common storage or the home address space where IXLCONN was issued.
- Any tasks that wait for the ECB to be posted reside in the home address space where IXLCONN was issued.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that contains the address of the ECB to be posted when the request completes. The ECB must be aligned on a fullword boundary.

**,REQID=NO\_REQID**

**,REQID=reqid**

Use this input parameter to specify a user-defined request identifier to be associated with the request. You can specify this request identifier on the IXPURGE macro to cancel a request that has not yet been processed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the user-defined request identifier.

**,REQTOKEN=reqtoken**

Use this output parameter with either MODE=SYNCTOKEN or MODE=ASYNCTOKEN to specify the address of a storage area to receive the request token that is returned when the request will be processed asynchronously. This token, which uniquely identifies the request, must be used as input to the IXLFCOMP macro, which you use to determine if the request has completed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field where the system will put the request token.

**,RESTOKEN=NO RESTOKEN**

**,RESTOKEN=restoken**

Use this input parameter to specify a restart token that can be used to resume processing of a RESET\_REFBIT request that exceeded the model-dependent time-out criteria. The restart token is returned in the answer area and should be specified on the next RESET\_REFBIT request to resume processing with the next directory entry to be processed.

If the request does not exceed the model-dependent time-out criteria, the returned token will not be provided.

**Note:**

1. Specifying a restart token of all zeros causes cache services to treat all of the directory entries as unprocessed.
2. Do not specify a restart token other than the one returned in the answer area or one set to all zeros, because results will be unpredictable.

Requestors that specify or default to IXLCONN ALLOWAUTO=NO must use the standard restart token. Requestors that specify IXLCONN ALLOWAUTO=YES must use the extended restart token (EXTRESTOKEN).

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-byte field that contains the restart token.

**,RETCODE=retcode**

Use this output parameter to specify a field to contain the return code. (The return code is also returned in register 15.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the return code when the request has completed.

**,RSNCODE=rsncode**

Use this output parameter to specify a field to contain the reason code returned, if applicable. (The reason code is also returned in register 0.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the reason code (if any) when the request has completed.

## ABEND Codes

---

Abend X'026' (See *z/OS MVS System Codes* for more information on this abend.)

## Return and Reason Codes

---

When the system returns control to your program:

- GPR 15 (and RETCODE, if specified) contains a return code.
- If the return code is not zero, GPR 0 (and RSNCODE, if specified) contains a reason code.

**Note:** The return and reason codes will also be put into the answer area (mapped by IXLYCAA), if there is one, when the request completes.

The IXLYCON macro provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

<b>0</b>	IXLRETCODEOK
<b>4</b>	IXLRETCODEWARNING
<b>8</b>	IXLRETCODEPARMERROR
<b>C</b>	IXLRETCODEENVERROR
<b>10</b>	IXLRETCODECOMPERROR

The following table contains hexadecimal return and reason codes, the equate symbols associated with each reason code, and the meaning and suggested action for each return and reason code.

Table 42. Return and Reason Codes for the IXLCACHE REQUEST=RESET_REFBIT Macro		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0	None.	<p><b>Meaning:</b> If you specified a MODE value of ASYNCECB, ASYNCEXIT, or ASYNCTOKEN, the request has been successfully initiated. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed. For any other MODE value, the request has successfully completed.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=ASYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=ASYNCEXIT, the connection's complete exit will be given control when the request has completed.</li> <li>• If you specified MODE=ASYNCTOKEN, use REQTOKEN when calling IXLFComp to determine when the request has completed.</li> </ul>

Table 42. Return and Reason Codes for the IXLCACHE REQUEST=RESET\_REFBIT Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	xxxx0402	<p><b>Equate Symbol:</b> IXLSRNCODEASYNCH</p> <p><b>Meaning:</b> The request could not be processed synchronously. It will be processed asynchronously. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed.</p> <ul style="list-style-type: none"> <li>• If you specified MODE=SYNCECB, the ECB specified in REQECB will be posted when the request has finished.</li> <li>• If you specified MODE=SYNCEXIT, the connection's complete exit will be given control when the request has finished.</li> <li>• If you specified MODE=SYNCTOKEN, the REQTOKEN contains an asynchronous request token that may be used on the IXLFCOMP macro to determine when the request has completed.</li> </ul> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=SYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=SYNCEXIT, the connection's completion exit will be given control when the request has completed.</li> <li>• If you specified MODE=SYNCTOKEN, use REQTOKEN when calling IXLFCOMP to determine when the request has completed.</li> </ul>
4	xxxx0409	<p><b>Equate Symbol:</b> IXLSRNCODETIMEOUT</p> <p><b>Meaning:</b> The request has completed prematurely because the model-dependent time-out criteria of the coupling facility has been exceeded. A token for restarting the request has been returned in the answer area (field CAARESTOKEN or CAAEXTRESTOKEN). The number of processed directory entries that initially had the reference bit set (field CAAREFCOUNT), and the number of directory entries processed (field CAADIRCOUNT) are returned in the answer area.</p> <p><b>Action:</b> Reissue the request using the appropriate restart token. For more information about premature request completion, see <i>z/OS MVS Programming: Sysplex Services Guide</i>.</p>

Table 42. Return and Reason Codes for the IXLCACHE REQUEST=RESET\_REFBIT Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0801	<p><b>Equate Symbol:</b> IXLRSNCODEBADPARMLIST</p> <p><b>Meaning:</b> The parameter list (mfctrl) for this request is not addressable. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the parameter list address.</li> <li>• The parameter list must be addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are issuing this macro while disabled, the parameter list must reside in either page-fixed or DREF storage.</li> <li>• If you are issuing this macro in AR-mode and you specified the parameter list address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are issuing this macro in AR-mode, SYSSTATE ASCENV=AR must be issued before this macro.</li> </ul>
8	xxxx0804	<p><b>Equate Symbol:</b> IXLRSNCODEBADVERSION#</p> <p><b>Meaning:</b> The version number in the macro parameter list is not compatible with the level of XES currently being used. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that your program did not overlay the parameter list storage.</li> <li>• Verify that your program was assembled with the correct macro library for the release of MVS on which your program is running.</li> <li>• Verify that your program is running on an MVS system that supports the version of the macro you are using.</li> </ul>



Table 42. Return and Reason Codes for the IXLCACHE REQUEST=RESET\_REFBIT Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx080A	<p><b>Equate Symbol:</b> IXLSRNCODEBADCONTOKEN</p> <p><b>Meaning:</b> Program error. The specified connect token (CONTOKEN) was not valid for one of the following reasons:</p> <ol style="list-style-type: none"> <li>1. The user with the connection identifier represented by the token has disconnected from the structure.</li> <li>2. The connector's task (the task that issued IXLCONN) ended.</li> <li>3. The specified token is not the token that was returned from IXLCONN.</li> <li>4. The request was issued from an address space other than the address space in which IXLCONN was issued.</li> <li>5. The connect token was invalidated during rebuild.</li> <li>6. The connect token was invalidated by XES.</li> </ol> <p><b>Note:</b> The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Verify that the CONTOKEN value specified was valid and for the correct structure. The numbers on the following actions correspond to the numbers above.</p> <ol style="list-style-type: none"> <li>1. Discontinue use. Perform recovery and cleanup for the structure and check protocol for use of structure to determine why you did not know of the disconnection.</li> <li>2. Discontinue use. Perform recovery and cleanup for the structure and check protocol for use of structure to determine why you did not know that the task ended.</li> <li>3. Issue IXLCONN to connect to the structure. The contoken is returned in the CONACONTOKEN field of the answer area specified on the IXLCONN request.</li> <li>4. Issue your request from the same address space the IXLCONN was issued.</li> <li>5. Participate in the rebuild. When it is complete, try again.</li> <li>6. Discontinue use. Perform recovery and cleanup for the structure and check protocol for use of structure.</li> </ol> <p>You may want to issue IXCQUERY to get more information about the structure.</p>
8	xxxx0824	<p><b>Equate Symbol:</b> IXLSRNCODEWRONGSTRTYPE</p> <p><b>Meaning:</b> The connect token specified by CONTOKEN is not to a cache structure.</p> <p><b>Action:</b> Verify the connect token for this cache structure.</p> <p><b>Note:</b> The connect token can be found in the IXLCONN answer area. Verify the value specified on the TYPE= parameter of the IXLCONN request for this structure.</p>

Table 42. Return and Reason Codes for the IXLCACHE REQUEST=RESET\_REFBIT Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0838	<p><b>Equate Symbol:</b> IXLRNCODEBADANSAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by ANSAREA is not addressable.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the ANSAREA address.</li> <li>• ANSAREA must be addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are calling IXLCACHE while disabled, ANSAREA must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLCACHE in AR-mode and you specified the ANSAREA address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are calling IXLCACHE in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLCACHE macro.</li> </ul>
8	xxxx0839	<p><b>Equate Symbol:</b> IXLRNCODEBADREQTOKENAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by REQTOKEN is not addressable.</p> <p><b>Action:</b> Ensure that the request token area specified by REQTOKEN is valid:</p> <ul style="list-style-type: none"> <li>• Verify the REQTOKEN address.</li> <li>• If you are calling IXLCACHE while disabled, REQTOKEN must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLCACHE in AR-mode and you specified the REQTOKEN address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are calling IXLCACHE in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLCACHE macro.</li> </ul>
8	xxxx083D	<p><b>Equate Symbol:</b> IXLRNCODEBADANSLEN</p> <p><b>Meaning:</b> Program error. The length of the answer area, as specified by ANSLEN, is not sufficient to contain answer area information. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Increase the size of the answer area provided for the request, and rerun your program. Depending on the macro version number, specify the length of the answer area as follows:</p> <ul style="list-style-type: none"> <li>• When the connection specified ASYNCXI=1 on the IXLCONN invocation and AXIOVERRIDE=0 was specified or defaulted to for the IXLCACHE request, the answer area length must be a minimum value of CAALEVEL2LEN</li> <li>• When the value of the macro version number (PLISTVER) is 4 or more, the minimum answer area length is CAALEVEL1LEN.</li> <li>• When the value of the macro version number (PLISTVER) is 0-3, the minimum answer area length is CAALEVEL0LEN.</li> </ul>

Table 42. Return and Reason Codes for the IXLCACHE REQUEST=RESET\_REFBIT Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0849	<p><b>Equate Symbol:</b> IXLRNCODEBADRESTOKEN</p> <p><b>Meaning:</b> Program error. The restart token specified by RESTOKEN is not valid.</p> <p><b>Action:</b> Determine why the restart token cannot be used to resume the request. Possible causes are:</p> <ul style="list-style-type: none"> <li>• The specified token does not correspond to the restart token returned in the answer area of the previous request.</li> <li>• The user specified RESTOKEN when EXTRESTOKEN was required.</li> <li>• The user specified EXTRESTOKEN when RESTOKEN was required.</li> </ul> <p>For more information about premature request completion, see <a href="#">z/OS MVS Programming: Sysplex Services Guide</a>.</p>
8	xxxx0851	<p><b>Equate Symbol:</b> IXLRNCODENOSUSPENDISABLE</p> <p><b>Meaning:</b> Program error. The request failed because MODE=SYNCSUSPEND was specified, but the caller is disabled and cannot be suspended.</p> <p><b>Action:</b> Either specify another MODE value or become enabled (release the CPU lock); then reissue the request.</p>
8	xxxx0887	<p><b>Equate Symbol:</b> IXLRNCODEBADEXTRESTOKEN</p> <p><b>Meaning:</b> Program error. The extended restart token specified by EXTRESTOKEN is not valid. The specified token refers to an older instance of the target structure. A system-managed process occurred between the time a request returned the extended restart token and the time the connector tried to continue the request using that token.</p> <p><b>Action:</b> Discard the results of the initial request and reissue the request with an EXTRESTOKEN value of zero. For more information about restarting requests, see <a href="#">z/OS MVS Programming: Sysplex Services Guide</a>.</p>
C	xxxx0C06	<p><b>Equate Symbol:</b> IXLRNCODENOCONN</p> <p><b>Meaning:</b> Environmental error. No connectivity to the cache structure exists. This may occur due to operator commands such as VARY PATH,OFFLINE or CONFIG CHP,OFFLINE or hardware errors such as facility or path failures. The CONTOKEN is invalidated. The request fails.</p> <p><b>Action:</b> Either disconnect from the structure (using IXLDISC) or initiate a rebuild of the structure, if allowed (using IXLREBLD).</p>

Table 42. Return and Reason Codes for the IXLCACHE REQUEST=RESET\_REFBIT Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C13	<p><b>Equate Symbol:</b> IXLRSNCODEREQPURGED</p> <p><b>Meaning:</b> Environmental error. The request was purged prior to completion of the request. Possible reasons include:</p> <ul style="list-style-type: none"> <li>• The connector failed.</li> <li>• The connector disconnected.</li> <li>• The requestor failed.</li> <li>• The request was purged by IXLPURGE.</li> <li>• Requests were purged when the connector provided an IXLEERSP response for the Rebuild Quiesce, Rebuild Stop, or Rebuild Cleanup event.</li> <li>• The connector invoked IXLREBLD REQUEST=COMPLETE.</li> <li>• The secondary address space was no longer valid.</li> </ul> <p><b>Action:</b> None if this is expected. Otherwise, determine why the connector failed.</p>
C	xxxx0C14	<p><b>Equate Symbol:</b> IXLRSNCODESTATUSUNKNOWN</p> <p><b>Meaning:</b> Environmental error. The IXLCACHE request has completed, but the final disposition of the request cannot be determined. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Verify the validity of your data by comparing the expected results with what is in the coupling facility.</p>
C	xxxx0C25	<p><b>Equate Symbol:</b> IXLRSNCODESTRFAILURE</p> <p><b>Meaning:</b> Environmental error. The cache structure failed prior to completion of the request.</p> <p><b>Action:</b> Attempt to rebuild the structure using IXLREBLD, or disconnect from the structure using IXLDISC.</p>
C	xxxx0CA0	<p><b>Equate Symbol:</b> IXLRSNCODEQUIESCEDSUSPENDFAIL</p> <p><b>Meaning:</b> Environmental error. The request is failed because the structure is quiesced for a system-managed process and SUSPEND=FAIL is specified on the IXLCONN.</p> <p><b>Action:</b> None, if this is expected.</p>
C	xxxxFFFF	<p><b>Equate Symbol:</b> IXLRSNCODENOTAVAILABLE</p> <p><b>Meaning:</b> Environmental error. XES functions are not available. The coupling facility hardware might not be present.</p> <p><b>Action:</b> XES may not be used when XCF is running local mode. XES requires that the coupling facility be installed.</p>

Table 42. Return and Reason Codes for the IXLCACHE REQUEST=RESET\_REFBIT Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
10	xxxx10xx	<p><b>Equate Symbol:</b> IXLRSNCODECOMPERROR</p> <p><b>Meaning:</b> System error. XES processing failure. The state of the structure and the disposition of the request are unpredictable. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Save the reason code information, and contact the IBM support center.</p>



## Chapter 45. IXLCACHE REQUEST=SET\_RECLVCTR

### Description

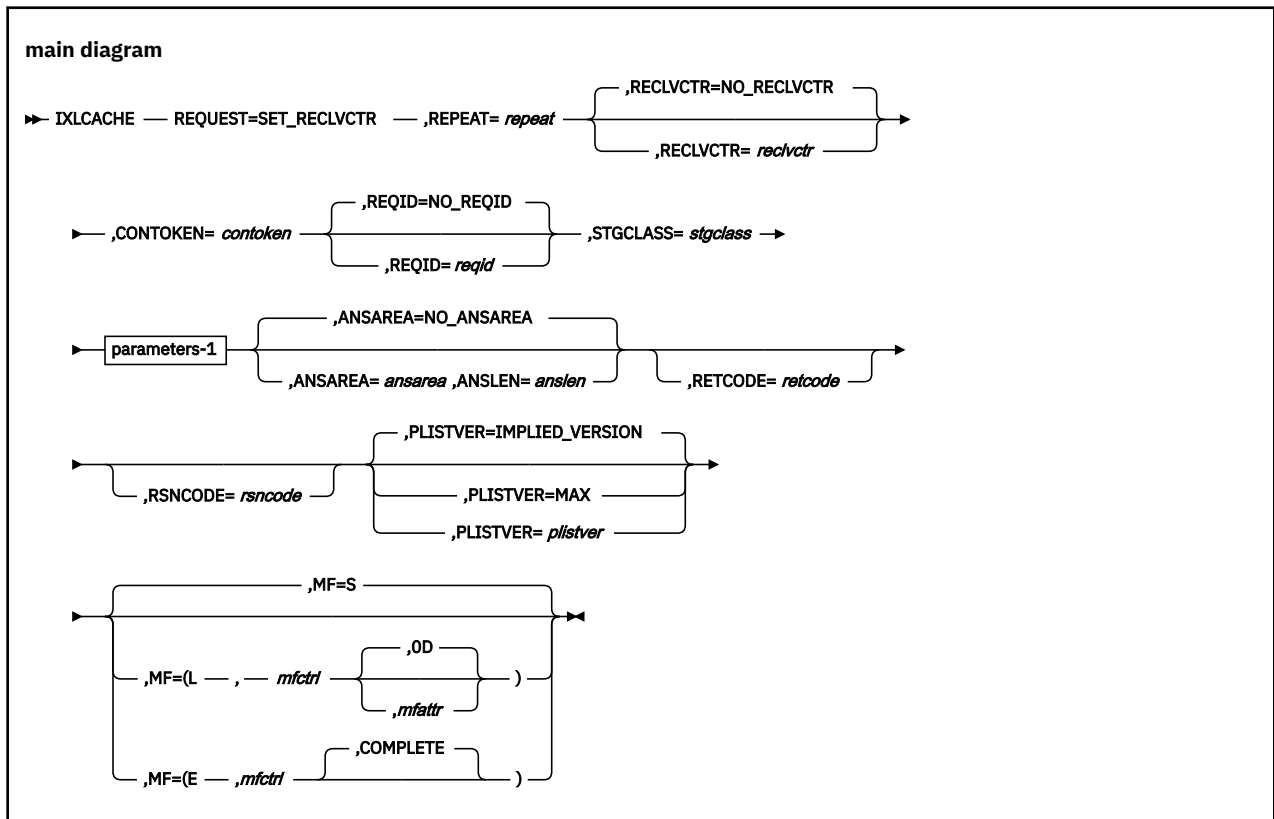
A SET\_RECLVCTR request allows you to define, activate, or deactivate a reclaiming vector (RECLVCTR) for a specified storage class (STGCLASS). Once defined, the reclaiming vector specifies how many reclaims are allowed against each storage class defined in the structure for requests targeted to the storage class to which the reclaim vector pertains. The reclaiming vector is associated with the specified storage class (STGCLASS) and will be used to satisfy subsequent IXLCACHE requests that require resources in that storage class.

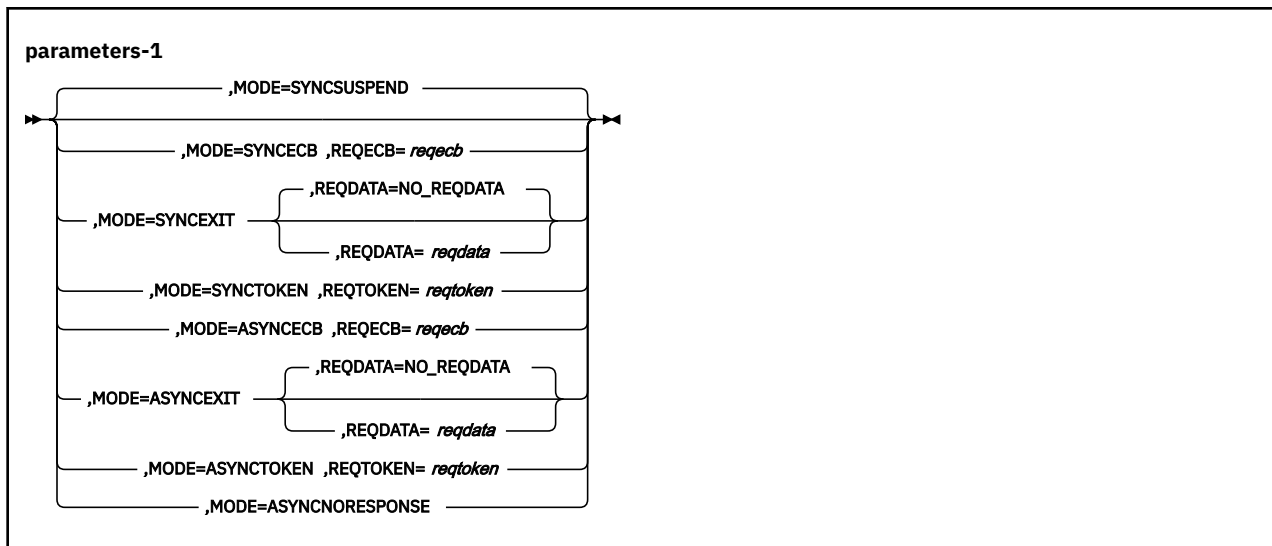
You can also specify the number of passes (REPEAT) through the reclaim vector before reclaim processing reverts to the default reclaim algorithm. (By default, the system attempts to reclaim the least recently used resources belonging to data items in the target storage class.) The reclaiming vector you specify remains active for the specified number of passes, or until it is deactivated. Specifying a zero on the REPEAT parameter deactivates the reclaiming vector.

When the system processes an IXLALTER request for a cache structure, all active reclaim vectors associated with all storage classes for the structure are deactivated. While the alter process continues, the system rejects any attempt to activate a reclaim vector. When the alter process completes, the system does not automatically reactivate any reclaim vectors that were deactivated when the structure alter was initiated. It is the responsibility of the user to activate any new or still-needed reclaim vectors.

### Syntax Diagram

The syntax diagram for IXLCACHE REQUEST=SET\_RECLVCTR is as follows:





**Note:** When MODE=SYNCTOKEN or MODE=ASYNCTOKEN is specified, ANSAREA= *ansarea* and ANSLEN= *anslen* are required.

## Parameter Descriptions

The parameter descriptions for REQUEST=SET\_RECLVCTR are listed in alphabetical order. Default values are underlined:

### REQUEST=SET\_RECLVCTR

Use this input parameter to specify that the reclaiming vector (RECLVCTR) for the specified storage class (STGCLASS) be activated or deactivated.

### **,ANSAREA=NO\_ANSAREA**

### **,ANSAREA=ansarea**

Use this output parameter to specify an answer area to contain information returned from the request. The format of the answer area is described by the IXLYCAA mapping macro.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an area (with a length of ANSLEN) where the information returned by the request will be put.

### **,ANSLEN=anslen**

Use this inppaameter to specify the size of the storage area specified by ANSAREA.

Use either CAALEVEL0LEN, CAALEVEL1LEN or CAALEVEL2LEN of the IXLYCAA mapping macro to determine the minimum size of the answer area. The answer area length must be at least large enough to accommodate the level of the IXLYCAA mapping appropriate to the requested function.

- When the connection specified ASYNCXI=1 on the IXLCONN invocation and AXIOVERRIDE=0 was specified or defaulted to for the IXLCACHE request, the answer area length is a required parameter and must be a minimum value of CAALEVEL2LEN to contain a returned asynchronous cross-invalidation sequence number (CAAASYNXISEQNUM).
- When the value of PLISTVER is 4 or above, the minimum answer area length is CAALEVEL1LEN.
- When the value of PLISTVER is 0 - 3, the minimum answer area length is CAALEVEL0LEN.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12 2 to ) of a 2-byte field that contains the length of the answer area (ANSAREA).

### **,CONTO=conken**

Use this input parameter to specify the connect token that was returned by the IXLCONN service in the IXLCONN answer area, which is mapped which is by IXLYCONA. The connect token uniquely identifies your connection to the cache structure, and must be specified on each IXLCACHE invocation.



**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field that contains the connect token.

**,MF=S**  
**,MF=(L,mfctrl)**  
**,MF=(L,mfctrl,mfattr)**  
**,MF=(L,mfctrl,0D)**  
**,MF=(M,mfctrl)**  
**,MF=(M,mfctrl,COMPLETE)**  
**,MF=(M,mfctrl,NOCHECK)**  
**,MF=(E,mfctrl)**  
**,MF=(E,mfctrl,COMPLETE)**  
**,MF=(E,mfctrl,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

**,mfctrl**

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

**,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

**,COMPLETE**

**,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

**COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if SMILE=*var* were an optional parameter and the default is SMILE=NO\_SMILE then it would not be documented. However, if the default was SMILE=-), then it would be documented because a value would be the default.

**NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

,MODE=SYNCSUSPEND  
 ,MODE=SYNCECB  
 ,MODE=SYNCEXIT  
 ,MODE=SYNCTOKEN  
 ,MODE=ASYNCECB  
 ,MODE=ASYNCEXIT  
 ,MODE=ASYNCTOKEN  
 DEASYNCSNORESPONSE

Use this input parameter to specify:

- Whether the request is to be performed synchronously or asynchronously
- How you want to be notified of request completion if the request is processed asynchronously.

See *z/OS MVS Programming: Sysplex Services Guide* for more information on understanding synchronous and asynchronous cache operations.

#### **MODE=SYNCSUSPEND**

The request processes synchronously. If necessary, the request is suspended until it can complete synchronously. To use this option, your program must be enabled for I/O and external interrupts.

#### **MODE=SYNCECB**

The request processes synchronously if possible. If the request processes asynchronously, the ECB specified by REQECB is posted when the request completes.

#### **MODE=SYNCEXIT**

The request processes synchronously if possible. If the request processes asynchronously, your complete exit is given control when the request completes.

#### **MODE=SYNCTOKEN**

The request processes synchronously if possible. If the request processes asynchronously, an asynchronous request token is returned to the area specified by REQTOKEN. Use the returned request token on the IXLFCOMP macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=SYNCTOKEN is specified.

#### **MODE=ASYNCECB**

The request processes asynchronously. The ECB specified by REQECB is posted when the request completes.

#### **MODE=ASYNCEXIT**

The request processes asynchronously. Your complete exit is given control when the request completes.

#### **MODE=ASYNCTOKEN**

The request processes asynchronously. An asynchronous request token is returned to the area specified by REQTOKEN. Use the returned request token on the IXLFCOMP macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=ASYNCTOKEN is specified.

#### **MODE=ASYNCSNORESPONSE**

The request processes asynchronously. No notification of request completion is provided.

,PLISTVER=IMPLIED\_VERSION

,PLISTVER=MAX

,PLISTVER=*plistver*

Use this input parameter to specify the version of the macro. See [“Understanding IXLCACHE Version Support” on page 461](#) for a description of the options available with PLISTVER.

,RECLVCTR=NO\_RECLVCTR

,RECLVCTR=*reclvctr*

Use this input parameter to define and activate a reclaiming vector. The RECLVCTR area must be arranged as follows:

- The area must contain, starting at offset zero, a fully initialized array of halfwords, each corresponding to a storage class. (There must be one halfword for each storage class defined for

the structure.) The relative position of a halfword in the array determines which storage class that halfword is associated with. For instance, the first halfword in the array is associated with storage class one, the second halfword is associated with storage class two, and so forth.

- Each halfword in the array must contain the reclaiming count for its associated storage class. You can avoid reclaims from a storage class by placing zero in the appropriate halfword.

If the specified reclaiming vector is not already active and a REPEAT factor of zero is specified, the REPEAT factor is ignored. If the reclaiming vector is already active, a zero REPEAT factor deactivates the reclaiming vector.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a storage area (with a length of double the number of storage classes defined for the structure on the IXLCONN macro, keyword NUMSTGCLASS) that contains the reclaiming vectors.

#### **,REPEAT=repeat**

Use this input parameter to specify the repeat factor for the reclaiming vector being activated (or deactivated). The repeat factor defines the number of passes through the reclaiming vector before the vector deactivates. The specified value must fall within the range 0 to 65535, inclusive. Specifying a repeat factor of zero causes the active reclaiming vector for the specified storage class to be deactivated.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 2-byte field that contains the repeat factor.

#### **,REQDATA=NO\_REQDATA**

#### **,REQDATA=reqdata**

Use this input parameter with MODE=SYNCEXIT or MODE=ASYNCEXIT to pass any data you choose to the complete exit. The exit will get control only if the request is processed asynchronously.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the data to be passed to the complete exit.

#### **,REQECB=reqecb**

Use this output parameter with either MODE=SYNCECB or MODE=ASYNCECB to specify the address of an ECB, which is to be posted when the request completes if the request was processed asynchronously.

Before coding REQECB, you must ensure that:

- You initialize the ECB before you issue the request.
- The ECB resides in either common storage or the home address space where IXLCONN was issued.
- Any tasks that wait for the ECB to be posted reside in the home address space where IXLCONN was issued.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that contains the address of the ECB to be posted when the request completes. The ECB must be aligned on a fullword boundary.

#### **,REQID=NO\_REQID**

#### **,REQID=reqid**

Use this input parameter to specify a user-defined request identifier to be associated with the request. You can specify this request identifier on the IXPURGE macro to cancel a request that has not yet been processed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the user-defined request identifier.

#### **,REQTOKEN=reqtoken**

Use this output parameter with either MODE=SYNCTOKEN or MODE=ASYNCTOKEN to specify the address of a storage area to receive the request token that is returned when the request will be processed asynchronously. This token, which uniquely identifies the request, must be used as input to the IXLFComp macro, which you use to determine if the request has completed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field where the system will put the request token.

**,RETCODE=retcode**

Use this output parameter to specify a field to contain the return code. (The return code is also returned in register 15.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the return code when the request has completed.

**,RSNCODE=rsncode**

Use this output parameter to specify a field to contain the reason code returned, if applicable. (The reason code is also returned in register 0.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the reason code (if any) when the request has completed.

**,STGCLASS=stgclass**

Use this input parameter to specify the storage class for which a reclaim vector is to be activated or deactivated.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 1-byte field that contains the storage class.

## ABEND Codes

---

Abend X'026' (See [z/OS MVS System Codes](#) for more information on this abend.)

## Return and Reason Codes

---

When the system returns control to your program:

- GPR 15 (and RETCODE, if specified) contains a return code.
- If the return code is not zero, GPR 0 (and RSNCODE, if specified) contains a reason code.

**Note:** The return and reason codes will also be put into the answer area (mapped by IXLYCAA), if there is one, when the request completes.

The IXLYCON macro provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

<b>0</b>	IXLRETCODEOK
<b>4</b>	IXLRETCODEWARNING
<b>8</b>	IXLRETCODEPARMERROR
<b>C</b>	IXLRETCODEENVERROR
<b>10</b>	IXLRETCODECOMPERROR

The following table contains hexadecimal return and reason codes, the equate symbols associated with each reason code, and the meaning and suggested action for each return and reason code.

Table 43. Return and Reason Codes for the IXLCACHE REQUEST=SET\_RECLVCTR Macro

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0	None.	<p><b>Meaning:</b> If you specified a MODE value of ASYNCECB, ASYNCEXIT, ASYNCTOKEN, or ASYNCNORESPONSE, the request has been successfully initiated. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed. For any other MODE value, the request has successfully completed.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=ASYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=ASYNCEXIT, the connection's complete exit will be given control when the request has completed.</li> <li>• If you specified MODE=ASYNCTOKEN, use REQTOKEN when calling IXLFComp to determine when the request has completed.</li> <li>• If you specified MODE=ASYNCNORESPONSE, no action is required. You will not be notified when the request completes.</li> </ul>
4	xxxx0402	<p><b>Equate Symbol:</b> IXLRNCODEASYNCH</p> <p><b>Meaning:</b> The request could not be processed synchronously. It will be processed asynchronously. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed.</p> <ul style="list-style-type: none"> <li>• If you specified MODE=SYNCECB, the ECB specified in REQECB will be posted when the request has finished.</li> <li>• If you specified MODE=SYNCEXIT, the connection's complete exit will be given control when the request has finished.</li> <li>• If you specified MODE=SYNCTOKEN, the REQTOKEN contains an asynchronous request token that may be used on the IXLFComp macro to determine when the request has completed.</li> </ul> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=SYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=SYNCEXIT, the connection's completion exit will be given control when the request has completed.</li> <li>• If you specified MODE=SYNCTOKEN, use REQTOKEN when calling IXLFComp to determine when the request has completed.</li> </ul>
4	xxxx0414	<p><b>Equate Symbol:</b> IXLRNCODERCLVCTRNOTSET</p> <p><b>Meaning:</b> The reclaim vector was not set because either the structure size or the entry-to-element ratio is being changed through IXLALTER.</p> <p><b>Action:</b> Set the reclaim vector when notified that the alter processing for the structure is complete.</p>

Table 43. Return and Reason Codes for the IXLCACHE REQUEST=SET\_RECLVCTR Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0801	<p><b>Equate Symbol:</b> IXLRSNCODEBADPARMLIST</p> <p><b>Meaning:</b> The parameter list (mfctrl) for this request is not addressable. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the parameter list address.</li> <li>• The parameter list must be addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are issuing this macro while disabled, the parameter list must reside in either page-fixed or DREF storage.</li> <li>• If you are issuing this macro in AR-mode and you specified the parameter list address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are issuing this macro in AR-mode, SYSSTATE ASCENV=AR must be issued before this macro.</li> </ul>
8	xxxx0804	<p><b>Equate Symbol:</b> IXLRSNCODEBADVERSION#</p> <p><b>Meaning:</b> The version number in the macro parameter list is not compatible with the level of XES currently being used. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that your program did not overlay the parameter list storage.</li> <li>• Verify that your program was assembled with the correct macro library for the release of MVS on which your program is running.</li> <li>• Verify that your program is running on an MVS system that supports the version of the macro you are using.</li> </ul>

Table 43. Return and Reason Codes for the IXLCACHE REQUEST=SET\_RECLVCTR Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx080A	<p><b>Equate Symbol:</b> IXLRNCODEBADCONTOKEN</p> <p><b>Meaning:</b> Program error. The specified connect token (CONTOKEN) was not valid for one of the following reasons:</p> <ol style="list-style-type: none"> <li>1. The user with the connection identifier represented by the token has disconnected from the structure.</li> <li>2. The connector's task (the task that issued IXLCONN) ended.</li> <li>3. The specified token is not the token that was returned from IXLCONN.</li> <li>4. The request was issued from an address space other than the address space in which IXLCONN was issued.</li> <li>5. The connect token was invalidated during rebuild.</li> <li>6. The connect token was invalidated by XES.</li> </ol> <p><b>Note:</b> The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Verify that the CONTOKEN value specified was valid and for the correct structure. The numbers on the following actions correspond to the numbers above.</p> <ol style="list-style-type: none"> <li>1. Discontinue use. Perform recovery and cleanup for the structure and check protocol for use of structure to determine why you did not know of the disconnection.</li> <li>2. Discontinue use. Perform recovery and cleanup for the structure and check protocol for use of structure to determine why you did not know that the task ended.</li> <li>3. Issue IXLCONN to connect to the structure. The contoken is returned in the CONACONTOKEN field of the answer area specified on the IXLCONN request.</li> <li>4. Issue your request from the same address space the IXLCONN was issued.</li> <li>5. Participate in the rebuild. When it is complete, try again.</li> <li>6. Discontinue use. Perform recovery and cleanup for the structure and check protocol for use of structure.</li> </ol> <p>You may want to issue IXCQUERY to get more information about the structure.</p>
8	xxxx0824	<p><b>Equate Symbol:</b> IXLRNCODEWRONGSTRTYPE</p> <p><b>Meaning:</b> The connect token specified by CONTOKEN is not to a cache structure.</p> <p><b>Action:</b> Verify the connect token for this cache structure.</p> <p><b>Note:</b> The connect token can be found in the IXLCONN answer area. Verify the value specified on the TYPE= parameter of the IXLCONN request for this structure.</p>

Table 43. Return and Reason Codes for the IXLCACHE REQUEST=SET\_RECLVCTR Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx082D	<p><b>Equate Symbol:</b> IXLRNCODEBADSTGCLASS</p> <p><b>Meaning:</b> Program error. The storage class specified by STGCLASS exceeds the maximum number of storage classes defined for the cache structure.</p> <p><b>Action:</b> The number of storage classes allowed for a structure are defined on the IXLCONN macro by the NUMSTGCLASS parameter. Correct the STGCLASS parameter to specify a valid storage class.</p>
8	xxxx0832	<p><b>Equate Symbol:</b> IXLRNCODENORCLVCTR</p> <p><b>Meaning:</b> Program error. The REPEAT value specified is greater than zero, but the reclaim vector storage area (RECLVCTR) was not specified.</p> <p><b>Action:</b> If you intend to deactivate an active reclaim vector, specify zero for REPEAT. If you intend to specify a repeat factor for a reclaim vector you are activating, specify a valid reclaim vector as described under the RECLVCTR parameter.</p>
8	xxxx0838	<p><b>Equate Symbol:</b> IXLRNCODEBADANSAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by ANSAREA is not addressable.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the ANSAREA address.</li> <li>• ANSAREA must be addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are calling IXLCACHE while disabled, ANSAREA must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLCACHE in AR-mode and you specified the ANSAREA address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are calling IXLCACHE in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLCACHE macro.</li> </ul>
8	xxxx0839	<p><b>Equate Symbol:</b> IXLRNCODEBADREQTOKENAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by REQTOKEN is not addressable.</p> <p><b>Action:</b> Ensure that the request token area specified by REQTOKEN is valid:</p> <ul style="list-style-type: none"> <li>• Verify the REQTOKEN address.</li> <li>• If you are calling IXLCACHE while disabled, REQTOKEN must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLCACHE in AR-mode and you specified the REQTOKEN address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are calling IXLCACHE in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLCACHE macro.</li> </ul>



Table 43. Return and Reason Codes for the IXLCACHE REQUEST=SET\_RECLVCTR Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx083D	<p><b>Equate Symbol:</b> IXLRNCODEBADANSLEN</p> <p><b>Meaning:</b> Program error. The length of the answer area, as specified by ANSLEN, is not sufficient to contain answer area information. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Increase the size of the answer area provided for the request, and rerun your program. Depending on the macro version number, specify the length of the answer area as follows:</p> <ul style="list-style-type: none"> <li>• When the connection specified ASYNCXI=1 on the IXLCONN invocation and AXIOVERRIDE=0 was specified or defaulted to for the IXLCACHE request, the answer area length must be a minimum value of CAALEVEL2LEN</li> <li>• When the value of the macro version number (PLISTVER) is 4 or more, the minimum answer area length is CAALEVEL1LEN.</li> <li>• When the value of the macro version number (PLISTVER) is 0-3, the minimum answer area length is CAALEVEL0LEN.</li> </ul>
8	xxxx0851	<p><b>Equate Symbol:</b> IXLRNCODENOSUSPENDISABLE</p> <p><b>Meaning:</b> Program error. The request failed because MODE=SYNCSUSPEND was specified, but the caller is disabled and cannot be suspended.</p> <p><b>Action:</b> Either specify another MODE value or become enabled (release the CPU lock); then reissue the request.</p>
8	xxxx0868	<p><b>Equate Symbol:</b> IXLRNCODEBADRECLVCTR</p> <p><b>Meaning:</b> Program error. The storage area specified by RECLVCTR is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The address specified by RECLVCTR is valid.</li> <li>• The RECLVCTR area was not previously freed.</li> <li>• If the caller is running in AR-mode and the RECLVCTR parameter was specified using explicit register notation, ensure that the corresponding access register was updated appropriately.</li> <li>• If the caller is disabled, then RECLVCTR must reside in either nonpageable or disabled reference storage.</li> <li>• If your program is running in AR mode, you specified SYSSTATE ASCENV=AR before issuing the IXLCACHE macro.</li> </ul>

Table 43. Return and Reason Codes for the IXLCACHE REQUEST=SET\_RECLVCTR Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C06	<p><b>Equate Symbol:</b> IXLRSNCODENOCNN</p> <p><b>Meaning:</b> Environmental error. No connectivity to the cache structure exists. This may occur due to operator commands such as VARY PATH,OFFLINE or CONFIG CHP,OFFLINE or hardware errors such as facility or path failures. The CONTOKEN is invalidated. The request fails.</p> <p><b>Action:</b> Either disconnect from the structure (using IXLDISC) or initiate a rebuild of the structure, if allowed (using IXLREBLD).</p>
C	xxxx0C13	<p><b>Equate Symbol:</b> IXLRSNCODEREQPURGED</p> <p><b>Meaning:</b> Environmental error. The request was purged prior to completion of the request. Possible reasons include:</p> <ul style="list-style-type: none"> <li>• The connector failed.</li> <li>• The connector disconnected.</li> <li>• The requestor failed.</li> <li>• The request was purged by IXLPURGE.</li> <li>• Requests were purged when the connector provided an IXLEERSP response for the Rebuild Quiesce, Rebuild Stop, or Rebuild Cleanup event.</li> <li>• The connector invoked IXLREBLD REQUEST=COMPLETE.</li> <li>• The secondary address space was no longer valid.</li> </ul> <p><b>Action:</b> None if this is expected. Otherwise, determine why the connector failed.</p>
C	xxxx0C14	<p><b>Equate Symbol:</b> IXLRSNCODESTATUSUNKNOWN</p> <p><b>Meaning:</b> Environmental error. The IXLCACHE request has completed, but the final disposition of the request cannot be determined. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Verify the validity of your data by comparing the expected results with what is in the coupling facility.</p>
C	xxxx0C25	<p><b>Equate Symbol:</b> IXLRSNCODESTRFAILURE</p> <p><b>Meaning:</b> Environmental error. The cache structure failed prior to completion of the request.</p> <p><b>Action:</b> Attempt to rebuild the structure using IXLREBLD, or disconnect from the structure using IXLDISC.</p>
C	xxxx0CA0	<p><b>Equate Symbol:</b> IXLRSNCODEQUIESCEDSUSPENDFAIL</p> <p><b>Meaning:</b> Environmental error. The request is failed because the structure is quiesced for a system-managed process and SUSPEND=FAIL is specified on the IXLCONN.</p> <p><b>Action:</b> None, if this is expected.</p>

Table 43. Return and Reason Codes for the IXLCACHE REQUEST=SET\_RECLVCTR Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxxFFFF	<b>Equate Symbol:</b> IXLRSNCODENOTAVAILABLE <b>Meaning:</b> Environmental error. XES functions are not available. The coupling facility hardware might not be present. <b>Action:</b> XES may not be used when XCF is running local mode. XES requires that the coupling facility be installed.
10	xxxx10xx	<b>Equate Symbol:</b> IXLRSNCODECOMPERROR <b>Meaning:</b> System error. XES processing failure. The state of the structure and the disposition of the request are unpredictable. The answer area (ANSAREA) fields do not contain valid information. <b>Action:</b> Save the reason code information, and contact the IBM support center.



## Chapter 46. IXLCACHE REQUEST=UNLOCK\_CASTOUT

### Description

An UNLOCK\_CASTOUT request allows you to release the cast-out locks for multiple data items that are named in the storage areas specified by either BUFFER or BUFLIST. The data items listed by name in the buffer(s) are referenced by an index into the buffer(s), and are processed sequentially beginning with the data item identified by the first index (FIRSTNAME) and ending with the data item identified by the last index (LASTNAME). Besides releasing cast-out locks, the request updates the parity and user data in the directory entry for each data item named in the list. The parity and user data for each data item should also be supplied in the buffer(s). (See mapping macro IXLYCUNB for how the buffer(s) should be arranged.)

Once the cast-out lock is released for a data item, the data item might or might not remain associated with its cast-out class:

- If the data entry contains unchanged data, the data item is disassociated from any previously specified cast-out class.
- If the data entry contains changed data (the entry could have had changed data written to it while the cast-out lock was held), the data item is left associated with the last specified cast-out class.

You can also specify that a currently unchanged data item remain associated with its cast-out class by overriding its changed indicator in the directory entry for the data item to indicate the data item is changed. This information should also be supplied in the buffer(s).

An UNLOCK\_CASTOUT request can either fail or complete prematurely for the following reasons:

- A data item named in the buffer(s) does not exist in the structure. The index of the failing name is returned in the answer area (ANSAREA). All names preceding the failing name are processed.
- You do not hold the cast-out lock for a data item named in the buffer(s). The index of the failing name and the value of the cast-out lock are returned in the answer area because of either a user or PROCESSID mismatch. All names preceding the failing name are processed.
- The request has exceeded the model-dependent time-out criteria. The index of the next name to be processed is returned in the answer area. All names preceding this name are processed.

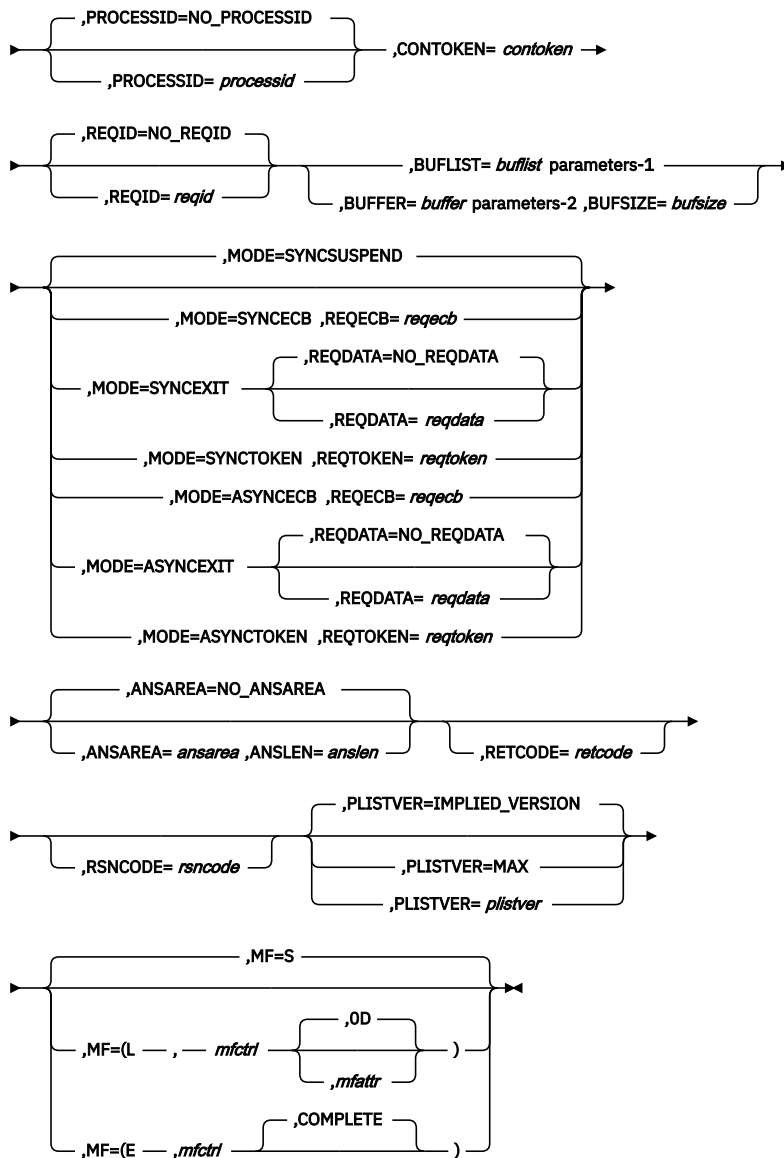
The index value of the data item name that caused the problem is returned in CAAULINDEX. You may update FIRSTNAME to point after CAAULINDEX, and reissue the request.

### Syntax Diagram

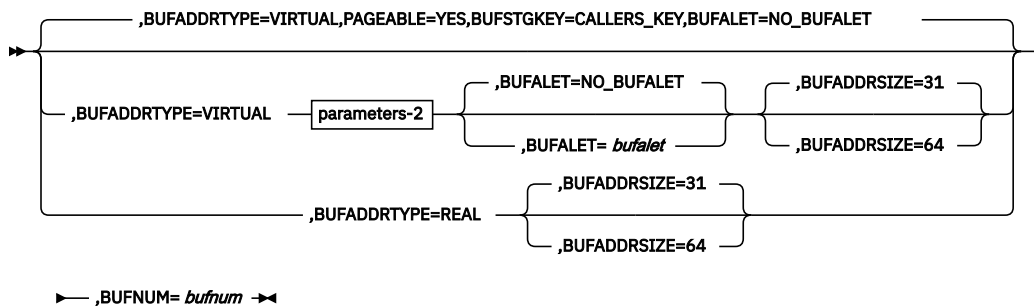
The syntax diagram for the IXLCACHE REQUEST=UNLOCK\_CASTOUT is as follows:

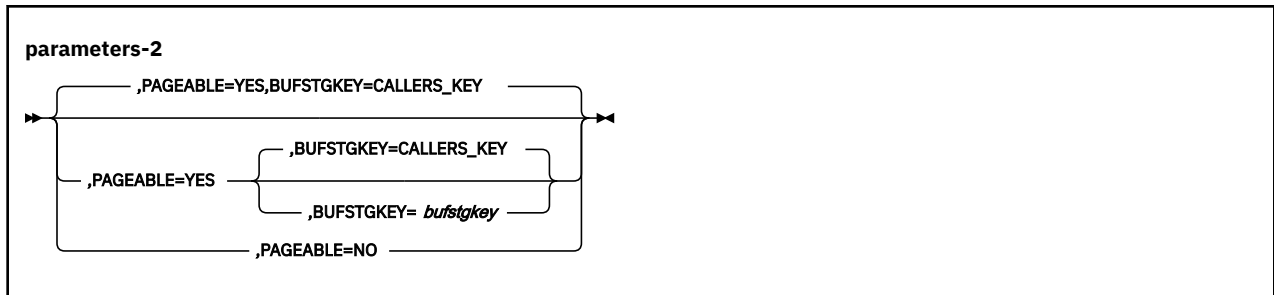
## main diagram

► IXLCACHE — REQUEST=UNLOCK\_CASTOUT — ,FIRSTNAME= *firstname* — ,LASTNAME= *lastname* →



## parameters-1





**Note:** If you specify `MODE=SYNCTOKEN` or `MODE=ASYNCTOKEN`, then you must also specify `ANSAREA=ansarea`, `ANSLEN=anslen`.

## Parameter Descriptions

The parameter descriptions for `REQUEST=UNLOCK_CASTOUT` are listed in alphabetical order. Default values are underlined:

### **REQUEST=UNLOCK\_CASTOUT**

Use this input parameter to specify that the cast-out locks held by your connection for the data items named in the storage areas specified by `BUFFER` or `BUFLIST` be released. Additionally, the user data and parity supplied in the buffer(s) updates the existing directory entry information for each data item named in the buffer(s).

#### **,ANSAREA=NO ,ANSAREA**

#### **,ANSAREA=ansarea**

Use this output parameter to specify an answer area to contain information returned from the request if the request does not complete successfully. See the return and reason code descriptions for this request to determine what fields in the answer area are valid for non-zero return codes. The format of the answer area is described by the `IXLYCAA` mapping macro.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an area (with a length of `ANSLEN`) where the information returned from the request will be put.

#### **,ANSLEN=anslen**

Use this inppaameter to specify the size of the storage area specified by `ANSAREA`.

Use either `CAALEVELOLEN`, `CAALEVEL1LEN` or `CAALEVEL2LEN` of the `IXLYCAA` mapping macro to determine the minimum size of the answer area. The answer area length must be at least large enough to accommodate the level of the `IXLYCAA` mapping appropriate to the requested function.

- When the connection specified `ASYNCXI=1` on the `IXLCONN` invocation and `AXIOVERRIDE=0` was specified or defaulted to for the `IXLCACHE` request, the answer area length is a required parameter and must be a minimum value of `CAALEVEL2LEN` to contain a returned asynchronous cross-invalidation sequence number (`CAASYNXISEQNUM`).
- When the value of `PLISTVER` is 4 or above, the minimum answer area length is `CAALEVEL1LEN`.
- When the value of `PLISTVER` is 0 - 3, the minimum answer area length is `CAALEVELOLEN`.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12 2 to ) of a 2-byte field that contains the length of the answer area (`ANSAREA`).

#### **,BUFADDRSIZE=31**

#### **,BUFADDRSIZE=64**

Use this input parameter to specify whether a 31-bit or a 64-bit address is specified by a `BUFLIST` entry.

#### **31**

The entry in `BUFLIST` is 31 bits in size.

#### **64**

The entry in `BUFLIST` is 64 bits in size.

**,BUFADDRTYPE=VIRTUAL****,BUFADDRTYPE=REAL**

Use this input parameter to specify whether the buffer addresses specified in the BUFLIST list are virtual storage or real storage addresses.

**VIRTUAL**

The buffer addresses are virtual storage addresses. The virtual storage can be pageable or nonpageable. See the PAGEABLE parameter for information about managing storage binds when specifying virtual storage addresses.

**REAL**

The buffer addresses are real storage addresses.

It is the caller's responsibility to manage the binds between the data buffer virtual storage and the real storage addresses provided. The caller must ensure that the data buffer virtual storage remains bound to the real storage addresses provided until the request completes.

**,BUFALET=NO\_BUFALET****,BUFALET=bufalet**

Use this input parameter to specify an access list entry token (ALET) to be used in referencing all of the buffers specified by BUFLIST.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 4-byte field that contains the ALET.

**,BUFFER=buffer**

Use this input parameter to specify a buffer area to contain a list of entries for which cast-out locks should be released.

Only 31-bit addressable virtual storage areas (below 2GB) are supported by the BUFFER specification. High virtual storage areas (above 2GB) can only be specified via the BUFLIST specification.

You must ensure that the storage area specified by BUFFER:

- Is a multiple of 4096 bytes.
- Is less than or equal to 65536 bytes.
- Starts on a 4096-byte boundary.
- Does not start below storage address 512.
- Consists of 32-byte elements, starting at offset zero. See the IXLYCUNB mapping macro for a mapping of the 32-byte element for an UNLOCK\_CASTOUT request.

See the BUFSIZE description for specifying the size of the buffer.

See [z/OS MVS Programming: Sysplex Services Guide](#) for more information on buffers.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an area (with a length of BUFSIZE) that contains the entry names to be processed.

**,BUFLIST=buflist**

Use this input parameter to specify a list of buffers to hold a list of entries for which cast-out locks should be released. BUFLIST specifies a 128-byte storage area that consists of a list of 0 to 16 buffer addresses.

Either 31-bit addressable (below 2GB) or 64-bit addressable (above 2GB) real or virtual storage areas are supported for the BUFLIST specification, depending on the specification for the BUFADDRTYPE and BUFADDRSIZE keywords. However, pageable high shared virtual storage areas (above 2GB) may not be used.

The format of the list is a set of 8-byte elements. The BUFADDRSIZE keyword denotes whether four or eight bytes of the element are used.

- If BUFADDRSIZE=31 is specified, then the first four bytes of each element are reserved space and the last four bytes contain the address of the buffer.
- If BUFADDRSIZE=64 is specified, then the full eight bytes specify the address of the buffer.



**The BUFLIST buffers must:**

- Reside in the same address space or same data space.
- Be 4096 bytes.
- Start on a 4096-byte boundary.
- Not start below storage address 512.
- Consist of 32-byte elements, starting at offset zero. See the IXLYCUNB mapping macro for a mapping of the 32-byte element for an UNLOCK\_CASTOUT request.

**Note:** The buffers do not have to be contiguous in storage. Cache services treat BUFLIST buffers as a single buffer even if the buffers are not contiguous.

See the BUFNUM parameter to specify the number of buffers in the buffer list.

See *z/OS MVS Programming: Sysplex Services Guide* for more information on buffers.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 128-byte storage area that contains a list of buffer addresses.

**,BUFNUM=*bufnum***

Use this input parameter to specify the number of buffers in the BUFLIST list. Valid BUFNUM values are from 0 to 16. A value of zero indicates that no cast-out locks will be released.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 1-byte field that contains the number of buffers (0-16) in the buffer list.

**,BUFSIZE=*bufsize***

Use this input parameter to specify the size of the BUFFER area. See the BUFFER parameter description for valid buffer sizes.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a fullword field that contains the size of the buffer (BUFFER) in bytes.

**,BUFSTGKEY=CALLERS\_KEY****,BUFSTGKEY=*bufstgkey***

Use this input parameter to specify a storage key that you define and use when referencing the buffers specified by BUFLIST or the buffer that is specified by BUFFER.

If you do not specify BUFSTGKEY, or if you specify BUFSTGKEY=CALLERS\_KEY, all references to one or more buffers are performed by using the caller's PSW key.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of an 8-bit field that contains the storage key in the format B'kkkkxxxx', where kkkk is the key and xxxx is ignored.

**,CONTO=*conken***

Use this input parameter to specify the connect token that was returned by the IXLCONN service in the IXLCONN answer area, which is mapped which is by IXLYCONA. The connect token uniquely identifies your connection to the cache structure, and must be specified on each IXLCACHE invocation.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field that contains the connect token.

**,FIRSTNAME=*firstname***

Use this input parameter to specify an index into the list of names stored in the BUFFER area or BUFLIST buffers. The index you select identifies the first data item to be processed.

FIRSTNAME can be used on a subsequent UNLOCK\_CASTOUT request to identify the data item with which processing should resume should the previous request complete prematurely.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 2-byte field that contains the index.

**,LASTNAME=*lastname***

Use this input parameter to specify an index into the list of names stored in the BUFFER area or BUFLIST buffers. The index you select identifies the last data item to be processed, and must be greater than, or equal to, the index specified for FIRSTNAME.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 2-byte field that contains the index.

**,MF=S****,MF=(L,*mfctrl*)****,MF=(L,*mfctrl*,*mfattr*)****,MF=(L,*mfctrl*,0D)****,MF=(M,*mfctrl*)****,MF=(M,*mfctrl*,COMPLETE)****,MF=(M,*mfctrl*,NOCHECK)****,MF=(E,*mfctrl*)****,MF=(E,*mfctrl*,COMPLETE)****,MF=(E,*mfctrl*,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

**,*mfctrl***

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

**,*mfattr***

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

**,COMPLETE****,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

**COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if SMILE=*var* were an optional parameter and the default is SMILE=NO\_SMILE then it would not be documented. However, if the default was SMILE=-), then it would be documented because a value would be the default.

**NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

**,MODE=SYNCSUSPEND**  
**,MODE=SYNCECB**  
**,MODE=SYNCEXIT**  
**,MODE=SYNCTOKEN**  
**,MODE=ASYNCECB**  
**,MODE=ASYNCEXIT**  
**,MODE=ASYNCTOKEN**

Use this input parameter to specify:

- Whether the request is to be performed synchronously or asynchronously
- How you want to be notified of request completion if the request is processed asynchronously.

See *z/OS MVS Programming: Sysplex Services Guide* for more information on understanding synchronous and asynchronous cache operations.

#### **SYNCSUSPEND**

The request processes synchronously. If necessary, the request is suspended until it can complete synchronously. To use this option, your program must be enabled for I/O and external interrupts.

#### **SYNCECB**

The request processes synchronously if possible. If the request processes asynchronously, the ECB specified by REQECB is posted when the request completes.

#### **SYNCEXIT**

The request processes synchronously if possible. If the request processes asynchronously, your complete exit is given control when the request completes.

#### **SYNCTOKEN**

The request processes synchronously if possible. If the request processes asynchronously, an asynchronous request token is returned to the area specified by REQTOKEN. Use the returned request token on the IXLFCOMP macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=SYNCTOKEN is specified.

#### **ASYNCECB**

The request processes asynchronously. The ECB specified by REQECB is posted when the request completes.

#### **ASYNCEXIT**

The request processes asynchronously. Your complete exit is given control when the request completes.

#### **ASYNCTOKEN**

The request processes asynchronously. An asynchronous request token is returned to the area specified by REQTOKEN. Use the returned request token on the IXLFCOMP macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=ASYNCTOKEN is specified.

**,PAGEABLE=YES**

**,PAGEABLE=NO**

Use this input parameter to identify whether the storage areas specified by BUFFER or BUFLIST are in pageable or potentially pageable storage.

#### **YES**

Specify this option to indicate that the BUFFER or BUFLIST buffers reside in pageable virtual storage. XES performs the required page fixing to fix the buffers in real storage while the cache or list request transfers data to or from the coupling facility.

This includes storage obtained from pageable subpools, disabled reference (DREF) storage, and may include storage that has the potential to become pageable during the processing of a request. (An example is address space storage owned by any swappable address space, for which a PGSER FIX has been successfully processed, but for which the owning address space gets swapped during processing of a cache or list request.) This does not include implicitly non-pageable storage (for example, storage obtained from non-pageable subpools).

The system takes responsibility for managing binds to central storage for the duration of the cache or list request, regardless of what address space owns the storage or whether the storage-owning address space is swappable or nonswappable. The storage can be owned by any address space.

High shared virtual storage areas (above 2GB) may not be used.

## NO

Specify this option to indicate that the BUFFER or BUFLIST buffers reside in non-pageable virtual storage. XES does not page fix the buffers in real storage.

This includes implicitly non-pageable storage areas (for example, storage obtained from non-pageable subpools), and may include storage that has the potential to become pageable during the processing of a request (An example is address space storage owned by any swappable address space, for which a PGSER FIX has been successfully processed, but for which the owning address space gets swapped-out during processing of a cache or list request.)

The system takes responsibility for managing binds to central storage for the duration of the cache or list request, if and only if the non-pageable storage is owned by either the requester's address space or the connector's address space. If the storage is owned by any other address space, then the invoker is responsible for ensuring that the virtual storage remains non-pageable for the duration of the request (including the case in which the storage is owned by a swappable address space that is swapped during processing of an IXLCACHE or IXLLIST request). Subject to this consideration, the storage can be owned by any address space. See *z/OS MVS Programming: Sysplex Services Guide*.

## **,PLISTVER=IMPLIED\_VERSION**

## **,PLISTVER=MAX**

## **,PLISTVER=plistver**

Use this input parameter to specify the version of the macro. See “[Understanding IXLCACHE Version Support](#)” on page 461 for a description of the options available with PLISTVER.

## **,PROCESSID=NO\_PROCESSID**

## **,PROCESSID=processid**

Use this input parameter to specify a user-defined process identifier to be compared with the cast-out lock along with the connection identifier.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12 2 to 12) of a 1-byte field that contains the user-defined process identifier.

## **,REQDATA=NO\_REQDATA**

## **,REQDATA=reqdata**

Use this input parameter with MODE=SYNCEXIT or MODE=ASYNCEXIT to pass any data you choose to the complete exit. The exit will get control only if the request is processed asynchronously.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the data to be passed to the complete exit.

## **,REQECB=reqecb**

Use this output parameter with either MODE=SYNCECB or MODE=ASYNCECB to specify the address of an ECB, which is to be posted when the request completes if the request was processed asynchronously.

Before coding REQECB, you must ensure that:

- You initialize the ECB before you issue the request.
- The ECB resides in either common storage or the home address space where IXLCONN was issued.
- Any tasks that wait for the ECB to be posted reside in the home address space where IXLCONN was issued.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that contains the address of the ECB to be posted when the request completes. The ECB must be aligned on a fullword boundary.

**,REQID=NO\_REQID****,REQID=reqid**

Use this input parameter to specify a user-defined request identifier to be associated with the request. You can specify this request identifier on the IXLPURGE macro to cancel a request that has not yet been processed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the user-defined request identifier.

**,REQTOKEN=reqtoken**

Use this output parameter with either MODE=SYNCTOKEN or MODE=ASYNCTOKEN to specify the address of a storage area to receive the request token that is returned when the request will be processed asynchronously. This token, which uniquely identifies the request, must be used as input to the IXLFCOMP macro, which you use to determine if the request has completed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field where the system will put the request token.

**,RETCODE=retcode**

Use this output parameter to specify a field to contain the return code. (The return code is also returned in register 15.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the return code when the request has completed.

**,RSNCODE=rsncode**

Use this output parameter to specify a field to contain the reason code returned, if applicable. (The reason code is also returned in register 0.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the reason code (if any) when the request has completed.

## ABEND Codes

---

Abend X'026' (See [z/OS MVS Programming: Sysplex Services Guide](#) for more information on this abend.)

## Return and Reason Codes

---

When the system returns control to your program:

- GPR 15 (and RETCODE, if specified) contains a return code.
- If the return code is not zero, GPR 0 (and RSNCODE, if specified) contains a reason code.

**Note:** The return and reason codes will also be put into the answer area (mapped by IXLYCAA), if there is one, when the request completes.

The IXLYCON macro provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

**0**

IXLRETCODEOK

**4**

IXLRETCODEWARNING

**8**

IXLRETCODEPARMERROR

**C**

IXLRETCODEENVERROR

**10**

IXLRETCODECOMPERROR

The following table contains hexadecimal return and reason codes, the equate symbols associated with each reason code, and the meaning and suggested action for each return and reason code.

Table 44. Return and Reason Codes for the IXLCACHE REQUEST=UNLOCK\_CASTOUT Macro

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0	None.	<p><b>Meaning:</b> If you specified a MODE value of ASYNCECB, ASYNCEXIT, or ASYNCTOKEN, the request has been successfully initiated. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed. For any other MODE value, the request has successfully completed.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=ASYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=ASYNCEXIT, the connection's complete exit will be given control when the request has completed.</li> <li>• If you specified MODE=ASYNCTOKEN, use REQTOKEN when calling IXLFComp to determine when the request has completed.</li> </ul>
4	xxxx0402	<p><b>Equate Symbol:</b> IXLRNCodeASYNCH</p> <p><b>Meaning:</b> The request could not be processed synchronously. It will be processed asynchronously. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed.</p> <ul style="list-style-type: none"> <li>• If you specified MODE=SYNCECB, the ECB specified in REQECB will be posted when the request has finished.</li> <li>• If you specified MODE=SYNCEXIT, the connection's complete exit will be given control when the request has finished.</li> <li>• If you specified MODE=SYNCTOKEN, the REQTOKEN contains an asynchronous request token that may be used on the IXLFComp macro to determine when the request has completed.</li> </ul> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=SYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=SYNCEXIT, the connection's completion exit will be given control when the request has completed.</li> <li>• If you specified MODE=SYNCTOKEN, use REQTOKEN when calling IXLFComp to determine when the request has completed.</li> </ul>

Table 44. Return and Reason Codes for the IXLCACHE REQUEST=UNLOCK\_CASTOUT Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	xxxx0409	<p><b>Equate Symbol:</b> IXLRSNCODETIMEOUT</p> <p><b>Meaning:</b> The request has completed prematurely because the model-dependent time-out criteria of the coupling facility has been exceeded. The index of the next name element to be processed has been returned in the answer area (field CAAULINDEX).</p> <p><b>Action:</b> Reissue the request. Update FIRSTNAME (from CAAULINDEX) with the index of the next name element to be processed.</p> <p>Be sure to process the information returned from this request before reissuing the request. The data returned from this request will be overwritten if you specify the same buffer address. Continue to reissue the request until the return code indicates that all processing has completed.</p> <p>For more information about premature request completion, see <a href="#">z/OS MVS Programming: Sysplex Services Guide</a>.</p>
8	xxxx0801	<p><b>Equate Symbol:</b> IXLRSNCODEBADPARMLIST</p> <p><b>Meaning:</b> The parameter list (mfctrl) for this request is not addressable. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the parameter list address.</li> <li>• The parameter list must be addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are issuing this macro while disabled, the parameter list must reside in either page-fixed or DREF storage.</li> <li>• If you are issuing this macro in AR-mode and you specified the parameter list address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are issuing this macro in AR-mode, SYSSTATE ASCENV=AR must be issued before this macro.</li> </ul>
8	xxxx0804	<p><b>Equate Symbol:</b> IXLRSNCODEBADVERSION#</p> <p><b>Meaning:</b> The version number in the macro parameter list is not compatible with the level of XES currently being used. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that your program did not overlay the parameter list storage.</li> <li>• Verify that your program was assembled with the correct macro library for the release of MVS on which your program is running.</li> <li>• Verify that your program is running on an MVS system that supports the version of the macro you are using.</li> </ul>

Table 44. Return and Reason Codes for the IXLCACHE REQUEST=UNLOCK\_CASTOUT Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx080A	<p><b>Equate Symbol:</b> IXLRSNCODEBADCONTOKEN</p> <p><b>Meaning:</b> Program error. The specified connect token (CONTOKEN) was not valid for one of the following reasons:</p> <ol style="list-style-type: none"> <li>1. The user with the connection identifier represented by the token has disconnected from the structure.</li> <li>2. The connector's task (the task that issued IXLCONN) ended.</li> <li>3. The specified token is not the token that was returned from IXLCONN.</li> <li>4. The request was issued from an address space other than the address space in which IXLCONN was issued.</li> <li>5. The connect token was invalidated during rebuild.</li> <li>6. The connect token was invalidated by XES.</li> </ol> <p><b>Note:</b> The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Verify that the CONTOKEN value specified was valid and for the correct structure. The numbers on the following actions correspond to the numbers above.</p> <ol style="list-style-type: none"> <li>1. Discontinue use. Perform recovery and cleanup for the structure and check protocol for use of structure to determine why you did not know of the disconnection.</li> <li>2. Discontinue use. Perform recovery and cleanup for the structure and check protocol for use of structure to determine why you did not know that the task ended.</li> <li>3. Issue IXLCONN to connect to the structure. The contoken is returned in the CONACONTOKEN field of the answer area specified on the IXLCONN request.</li> <li>4. Issue your request from the same address space the IXLCONN was issued.</li> <li>5. Participate in the rebuild. When it is complete, try again.</li> <li>6. Discontinue use. Perform recovery and cleanup for the structure and check protocol for use of structure.</li> </ol> <p>You may want to issue IXCQUERY to get more information about the structure.</p>



Table 44. Return and Reason Codes for the IXLCACHE REQUEST=UNLOCK\_CASTOUT Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx081E	<p><b>Equate Symbol:</b> IXLRSNCODEBADCOLOCKSTATE</p> <p><b>Meaning:</b> You requested that an entry in BUFFER or BUFLIST have its change bit overridden to indicate that the entry contains changed data, but this option is incompatible with the current cast-out lock state value, "write with castout." The "write with castout" state is entered when the cast-out lock is obtained by a WRITE_DATA request specifying GETCOLOCK=YES. The following fields are returned in the answer area:</p> <ul style="list-style-type: none"> <li>• CAACLOCKSTATE</li> <li>• CAAULINDEX</li> <li>• CAACLOCKVAL</li> </ul> <p>For a description of cast-out lock state values, see mapping macro IXLYCAA.</p> <p><b>Action:</b> The lock state and value are returned in the answer area (ANSAREA).</p> <ul style="list-style-type: none"> <li>• Determine who holds the lock for this data item.</li> <li>• Verify the names in the buffer you have passed.</li> <li>• Verify the FIRSTNAME and LASTNAME indexes.</li> <li>• Determine why this entry was in your list.</li> </ul> <p>You can bypass the current entry by updating FIRSTNAME with the index after the value in CAAULINDEX, and resubmitting your request.</p>
8	xxxx0824	<p><b>Equate Symbol:</b> IXLRSNCODEWRONGSTRTYPE</p> <p><b>Meaning:</b> The connect token specified by CONTOKEN is not to a cache structure.</p> <p><b>Action:</b> Verify the connect token for this cache structure.</p> <p><b>Note:</b> The connect token can be found in the IXLCONN answer area. Verify the value specified on the TYPE= parameter of the IXLCONN request for this structure.</p>
8	xxxx0825	<p><b>Equate Symbol:</b> IXLRSNCODENOENTRY</p> <p><b>Meaning:</b> Program error. The request failed because a name element specified an entry name that is not in the cache structure. All name elements before the offending name were processed; no name elements beyond the offending name were processed. The index of the offending name is returned in the answer area (field CAAULINDEX).</p> <p><b>Action:</b> Reissue the request specifying for FIRSTNAME the next valid name element to be processed.</p>

Table 44. Return and Reason Codes for the IXLCACHE REQUEST=UNLOCK\_CASTOUT Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0829	<p><b>Equate Symbol:</b> IXLRSNCODEBADUNLOCKVAL</p> <p><b>Meaning:</b> Program error. The UNLOCK_CASTOUT request encountered a name element for which the cast-out lock was not held for the connection specified by CONTOKEN and/or the PROCESSID specified. Processing was halted with this name element. The lock state (field CAACOLCKSTATE) and lock value (field CAACOLCKVAL) for the element as well as the index value (field CAAULINDEX) for the name element that caused the failure are returned in the answer area (ANSAREA).</p> <p><b>Action:</b></p> <p>Specify for FIRSTNAME the index of the next name element (CAAULINDEX + 1) for which the cast-out lock is held, and reissue the request.</p>
8	xxxx082B	<p><b>Equate Symbol:</b> IXLRSNCODEBADIDINDEX</p> <p><b>Meaning:</b> Program error. The name index specified by either FIRSTNAME or LASTNAME is not valid. Some elements may have been processed. CAAULINDEX contains the index of the name that caused the failure.</p> <p><b>Action:</b> Ensure that FIRSTNAME and LASTNAME specify valid indexes into the elements stored in BUFLIST or BUFFER.</p>
8	xxxx082F	<p><b>Equate Symbol:</b> IXLRSNCODEBADPARITY</p> <p><b>Meaning:</b> Program error. The parity bits in a name element are not valid.</p> <p><b>Action:</b> The buffers specified in the request contained parity bits that were not valid. The answer area field CAAULINDEX, contains the index of the name element that caused the failure. The value for the parity bits may be 00, 01, or 11 in bits 2 and 3 of the specification (xxpp xxxx where pp is the parity bits). See mapping macro IXLYCUNB to find the parity bits in the buffer.</p>
8	xxxx0833	<p><b>Equate Symbol:</b> IXLRSNCODEBADPGBLATTR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list is specified as being pageable (PAGEABLE=YES) but is not.</p> <p><b>Action:</b> Change the buffer area(s) to pageable storage, or specify PAGEABLE=NO. See the PAGEABLE parameter description for specification instructions.</p>

Table 44. Return and Reason Codes for the IXLCACHE REQUEST=UNLOCK\_CASTOUT Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0834	<p><b>Equate Symbol:</b> IXLRNCODEBADNONPGBLATTR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list is specified as being nonpageable (PAGEABLE=NO) but is either pageable or not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The type of storage in the buffer area corresponds to the value you specified on the PAGEABLE parameter. See the PAGEABLE parameter description for specification instructions.</li> <li>• The correct buffer address was used.</li> <li>• The buffer area(s) were not previously freed.</li> <li>• If BUFLIST was specified and your program is running in AR mode, ensure that: <ul style="list-style-type: none"> <li>– The BUFALET specification is correct.</li> <li>– You specified SYSSTATE ASCENV=AR before issuing the macro.</li> </ul> </li> <li>• If the caller is running in AR-mode and the BUFFER or BUFLIST parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> </ul>
8	xxxx0835	<p><b>Equate Symbol:</b> IXLRNCODEBADDATAADDR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list, is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The correct buffer address was used for BUFFER or for a buffer within the BUFLIST.</li> <li>• The buffer area(s) were not previously freed.</li> <li>• The buffer area(s) were allocated in a storage key that matches the key specified using the BUFSTGKEY parameter, or, if BUFSTGKEY is omitted, the caller's PSW key.</li> <li>• If the caller is running in AR-mode and the BUFFER or BUFLIST parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If the caller is running in AR-mode, SYSSTATE ASCENV=AR must be specified before issuing this macro.</li> <li>• If BUFLIST was specified and your program is running in AR mode the BUFALET specification is correct.</li> </ul>

Table 44. Return and Reason Codes for the IXLCACHE REQUEST=UNLOCK\_CASTOUT Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0836	<p><b>Equate Symbol:</b> IXLRNCODEBADREALADDR</p> <p><b>Meaning:</b> Program error. Real storage addresses were provided in a BUFLIST list, but one of the buffers is not addressable in central storage.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that BUFADDRTYPE was specified as you intended.</li> <li>• Ensure that the real buffer addresses specified by BUFLIST are valid.</li> </ul>
8	xxxx0838	<p><b>Equate Symbol:</b> IXLRNCODEBADANSAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by ANSAREA is not addressable.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the ANSAREA address.</li> <li>• ANSAREA must be addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are calling IXLCACHE while disabled, ANSAREA must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLCACHE in AR-mode and you specified the ANSAREA address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are calling IXLCACHE in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLCACHE macro.</li> </ul>
8	xxxx0839	<p><b>Equate Symbol:</b> IXLRNCODEBADREQTOKENAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by REQTOKEN is not addressable.</p> <p><b>Action:</b> Ensure that the request token area specified by REQTOKEN is valid:</p> <ul style="list-style-type: none"> <li>• Verify the REQTOKEN address.</li> <li>• If you are calling IXLCACHE while disabled, REQTOKEN must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLCACHE in AR-mode and you specified the REQTOKEN address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are calling IXLCACHE in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLCACHE macro.</li> </ul>

Table 44. Return and Reason Codes for the IXLCACHE REQUEST=UNLOCK\_CASTOUT Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx083D	<p><b>Equate Symbol:</b> IXLRSNCODEBADANSLEN</p> <p><b>Meaning:</b> Program error. The length of the answer area, as specified by ANSLEN, is not sufficient to contain answer area information. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Increase the size of the answer area provided for the request, and rerun your program. Depending on the macro version number, specify the length of the answer area as follows:</p> <ul style="list-style-type: none"> <li>• When the connection specified ASYNCXI=1 on the IXLCONN invocation and AXIOVERRIDE=0 was specified or defaulted to for the IXLCACHE request, the answer area length must be a minimum value of CAALEVEL2LEN</li> <li>• When the value of the macro version number (PLISTVER) is 4 or more, the minimum answer area length is CAALEVEL1LEN.</li> <li>• When the value of the macro version number (PLISTVER) is 0-3, the minimum answer area length is CAALEVEL0LEN.</li> </ul>
8	xxxx0851	<p><b>Equate Symbol:</b> IXLRSNCODENOSUSPENDISABLE</p> <p><b>Meaning:</b> Program error. The request failed because MODE=SYNCSUSPEND was specified, but the caller is disabled and cannot be suspended.</p> <p><b>Action:</b> Either specify another MODE value or become enabled (release the CPU lock); then reissue the request.</p>
8	xxxx0865	<p><b>Equate Symbol:</b> IXLRSNCODEBADBUFSPEC</p> <p><b>Meaning:</b> Program error. There is an error in the buffer specification.</p> <p><b>Action:</b> Check the following:</p> <ul style="list-style-type: none"> <li>• If BUFLIST was specified, check the requirements for BUFLIST and BUFNUM.</li> <li>• If BUFFER was specified, check the requirements for BUFFER and BUFSIZE.</li> <li>• Buffer pointer(s) in BUFLIST.</li> <li>• Buffer boundaries.</li> </ul>

Table 44. Return and Reason Codes for the IXLCACHE REQUEST=UNLOCK\_CASTOUT Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0866	<p><b>Equate Symbol:</b> IXLRSNCODEBADBUFKEY</p> <p><b>Meaning:</b> Program error. The buffer storage key specified by BUFSTGKEY is incorrect or BUFSTGKEY was not specified and the caller's PSW key does not match the key of the buffers.</p> <p>The data cannot be fetched from the specified buffer area.</p> <p><b>Action:</b> Check the following:</p> <ul style="list-style-type: none"> <li>• Determine if the key of the storage being used for the buffers is different from the PSW key.</li> <li>• If BUFSTGKEY was specified, verify that the key was put in the correct bits (see the explanation of this parameter for more information).</li> </ul>
8	xxxx0867	<p><b>Equate Symbol:</b> IXLRSNCODEBADBUFLIST</p> <p><b>Meaning:</b> Program error. The 128-byte storage area specified by BUFLIST is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The correct BUFLIST address was used.</li> <li>• The BUFLIST area was not previously freed.</li> <li>• If the caller is running in AR-mode and the BUFLIST parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If the caller is disabled, then the BUFLIST must reside in either nonpageable or disabled reference storage.</li> <li>• If your program is running in AR mode, ensure that you specified SYSSTATE ASCENV=AR before issuing the macro.</li> </ul>
8	xxxx08AD	<p><b>Equate Symbol:</b> IXLRSNCODEBADHIGHSHAREDVIRT</p> <p><b>Meaning:</b> Program error. The request specified a high shared virtual storage area (above 2GB).</p> <p><b>Action:</b> None required.</p>
C	xxxx0C06	<p><b>Equate Symbol:</b> IXLRSNCODENOCNN</p> <p><b>Meaning:</b> Environmental error. No connectivity to the cache structure exists. This may occur due to operator commands such as VARY PATH,OFFLINE or CONFIG CHP,OFFLINE or hardware errors such as facility or path failures. The CONTOKEN is invalidated. The request fails.</p> <p><b>Action:</b> Either disconnect from the structure (using IXLDISC) or initiate a rebuild of the structure, if allowed (using IXLREBLD).</p>

Table 44. Return and Reason Codes for the IXLCACHE REQUEST=UNLOCK\_CASTOUT Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C13	<p><b>Equate Symbol:</b> IXLRSNCODEREQPURGED</p> <p><b>Meaning:</b> Environmental error. The request was purged prior to completion of the request. Possible reasons include:</p> <ul style="list-style-type: none"> <li>• The connector failed.</li> <li>• The connector disconnected.</li> <li>• The requestor failed.</li> <li>• The request was purged by IXLPURGE.</li> <li>• Requests were purged when the connector provided an IXLEERSP response for the Rebuild Quiesce, Rebuild Stop, or Rebuild Cleanup event.</li> <li>• The connector invoked IXLREBLD REQUEST=COMPLETE.</li> <li>• The secondary address space was no longer valid.</li> </ul> <p><b>Action:</b> None if this is expected. Otherwise, determine why the connector failed.</p>
C	xxxx0C14	<p><b>Equate Symbol:</b> IXLRSNCODESTATUSUNKNOWN</p> <p><b>Meaning:</b> Environmental error. The IXLCACHE request has completed, but the final disposition of the request cannot be determined. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Verify the validity of your data by comparing the expected results with what is in the coupling facility.</p>
C	xxxx0C25	<p><b>Equate Symbol:</b> IXLRSNCODESTRFAILURE</p> <p><b>Meaning:</b> Environmental error. The cache structure failed prior to completion of the request.</p> <p><b>Action:</b> Attempt to rebuild the structure using IXLREBLD, or disconnect from the structure using IXLDISC.</p>
C	xxxx0CA0	<p><b>Equate Symbol:</b> IXLRSNCODEQUIESCEDSUSPENDFAIL</p> <p><b>Meaning:</b> Environmental error. The request is failed because the structure is quiesced for a system-managed process and SUSPEND=FAIL is specified on the IXLCONN.</p> <p><b>Action:</b> None, if this is expected.</p>
C	xxxxFFFF	<p><b>Equate Symbol:</b> IXLRSNCODENOTAVAILABLE</p> <p><b>Meaning:</b> Environmental error. XES functions are not available. The coupling facility hardware might not be present.</p> <p><b>Action:</b> XES may not be used when XCF is running local mode. XES requires that the coupling facility be installed.</p>

Table 44. Return and Reason Codes for the IXLCACHE REQUEST=UNLOCK\_CASTOUT Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
10	xxxx10xx	<b>Equate Symbol:</b> IXLRSNCODECOMPERROR <b>Meaning:</b> System error. XES processing failure. The state of the structure and the disposition of the request are unpredictable. The answer area (ANSAREA) fields do not contain valid information. <b>Action:</b> Save the reason code information, and contact the IBM support center.



---

## Chapter 47. IXLCACHE REQUEST=UNLOCK\_CO\_NAME

---

### Description

An UNLOCK\_CO\_NAME request allows you to release a single cast-out lock for a data item named in the storage area specified by CUNBAREA. You can also use the request to update the user data in the directory entry for the data item named. The user data, as well as a changed indicator and a parity value, is supplied in the CUNBAREA storage area. (CUNBAREA is mapped by the macro IXLYCUNB.) REQUEST=UNLOCK\_CO\_NAME is valid only for cache structures allocated in a coupling facility with CFLEVEL=4 or higher.

Once the cast-out lock is released for the data item, the data item might or might not remain associated with its cast-out class:

- If the data entry contains unchanged data, the data item is disassociated from any previously specified cast-out class.
- If the data entry contains changed data (the entry could have had changed data written to it while the cast-out lock was held), the data item is left associated with the last specified cast-out class.

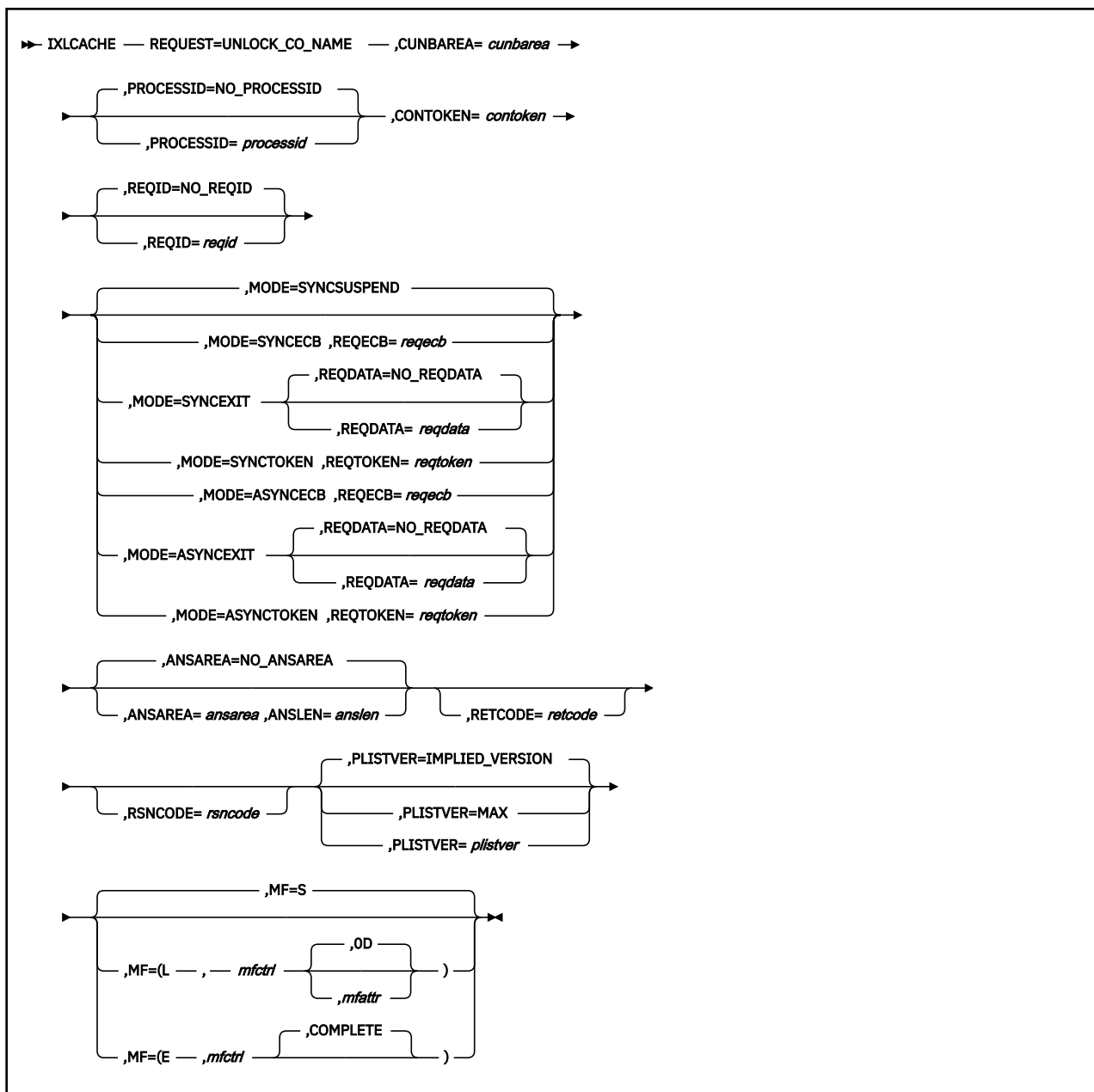
You can specify that a currently unchanged data item remain associated with its cast-out class by overriding its changed indicator in the directory entry for the data item.

Use the UNLOCK\_CO\_NAME request type to reset a single cast-out lock; use the UNLOCK\_CASTOUT request type to reset multiple cast-out locks with a single request.

---

### Syntax Diagram

The syntax diagram for the IXLCACHE REQUEST=UNLOCK\_CO\_NAME is as follows:



**Note:** If you specify MODE=SYNCTOKEN or MODE=ASYNCTOKEN, then you must also specify ANSAREA=*ansarea*, ANSLEN=*anslen*.

## Parameter Descriptions

The parameter descriptions for REQUEST=UNLOCK\_CO\_NAME are listed in alphabetical order. Default values are underlined:

### REQUEST=UNLOCK\_CO\_NAME

Use this input parameter to specify that the cast-out lock held by your connection for the data item named in the storage area specified by CUNBAREA be released. Additionally, the user data supplied in the CUNBAREA updates the existing directory entry information for the data item.

#### ,ANSAREA=NO\_ANSAREA

#### ,ANSAREA=ansarea

Use this output parameter to specify an answer area to contain information returned from the request when it completes. See the return and reason code descriptions for this request to determine what

fields in the answer area are valid for non-zero return codes. The format of the answer area is described by the IXLYCAA mapping macro.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an area (with a length of ANSLEN) where the information returned from the request will be put.

**,ANSLEN=*anslen***

Use this inppaameter to specify the size of the storage area specified by ANSAREA.

Use either CAALEVEL0LEN, CAALEVEL1LEN or CAALEVEL2LEN of the IXLYCAA mapping macro to determine the minimum size of the answer area. The answer area length must be at least large enough to accommodate the level of the IXLYCAA mapping appropriate to the requested function.

- When the connection specified ASYNCXI=1 on the IXLCONN invocation and AXIOVERRIDE=0 was specified or defaulted to for the IXLCACHE request, the answer area length is a required parameter and must be a minimum value of CAALEVEL2LEN to contain a returned asynchronous cross-invalidation sequence number (CAAASYNCXISEQNUM).
- When the value of PLISTVER is 4 or above, the minimum answer area length is CAALEVEL1LEN.
- When the value of PLISTVER is 0 - 3, the minimum answer area length is CAALEVEL0LEN.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12 2 to ) of a 2-byte field that contains the length of the answer area (ANSAREA).

**,CONTO=*conken***

Use this input parameter to specify the connect token that was returned by the IXLCONN service in the IXLCONN answer area, which is mapped which is by IXLYCONA. The connect token uniquely identifies your connection to the cache structure, and must be specified on each IXLCACHE invocation.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field that contains the connect token.

**,CUNBAREA=*cunbarea***

Use this input parameter to specify the 32-byte field containing the structure data name followed by request control information for the cast-out lock that is to be released. The format of the 32-byte area is described by the IXLYCUNB macro.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 32-byte area that contains the structure entry name and control information for the UNLOCK\_CO\_NAME request.

**,MF=S**

**,MF=(L,*mfctrl*)**

**,MF=(L,*mfctrl*,*mfattr*)**

**,MF=(L,*mfctrl*,0D)**

**,MF=(M,*mfctrl*)**

**,MF=(M,*mfctrl*,COMPLETE)**

**,MF=(M,*mfctrl*,NOCHECK)**

**,MF=(E,*mfctrl*)**

**,MF=(E,*mfctrl*,COMPLETE)**

**,MF=(E,*mfctrl*,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into

the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

**,mfctrl**

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

**,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

**,COMPLETE**

**,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

**COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if *SMILE=var* were an optional parameter and the default is *SMILE=NO\_SMILE* then it would not be documented. However, if the default was *SMILE=-*), then it would be documented because a value would be the default.

**NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

**,MODE=SYNCSUSPEND**

**,MODE=SYNCECB**

**,MODE=SYNCEXIT**

**,MODE=SYNCTOKEN**

**,MODE=ASYNCECB**

**,MODE=ASYNCEXIT**

**,MODE=ASYNCTOKEN**

Use this input parameter to specify:

- Whether the request is to be performed synchronously or asynchronously
- How you want to be notified of request completion if the request is processed asynchronously.

See *z/OS MVS Programming: Sysplex Services Guide* for more information on understanding synchronous and asynchronous cache operations.

**SYNCSUSPEND**

The request processes synchronously. If necessary, the request is suspended until it can complete synchronously. To use this option, your program must be enabled for I/O and external interrupts.

**SYNCECB**

The request processes synchronously if possible. If the request processes asynchronously, the ECB specified by REQECB is posted when the request completes.

**SYNCEXIT**

The request processes synchronously if possible. If the request processes asynchronously, your complete exit is given control when the request completes.

**SYNCTOKEN**

The request processes synchronously if possible. If the request processes asynchronously, an asynchronous request token is returned to the area specified by REQTOKEN. Use the returned request token on the IXLFComp macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=SYNCTOKEN is specified.

**ASYNCECB**

The request processes asynchronously. The ECB specified by REQECB is posted when the request completes.

**ASYNCEXIT**

The request processes asynchronously. Your complete exit is given control when the request completes.

**ASYNCTOKEN**

The request processes asynchronously. An asynchronous request token is returned to the area specified by REQTOKEN. Use the returned request token on the IXLFCOMP macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=ASYNCTOKEN is specified.

**,PLISTVER=IMPLIED\_VERSION****,PLISTVER=MAX****,PLISTVER=*plistver***

Use this input parameter to specify the version of the macro. See “[Understanding IXLCACHE Version Support](#)” on page 461 for a description of the options available with PLISTVER.

**,PROCESSID=NO\_PROCESSID****,PROCESSID=*processid***

Use this input parameter to specify a user-defined process identifier to be compared with the cast-out lock along with the connection identifier.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12 2 to 12) of a 1-byte field that contains the user-defined process identifier.

**,REQDATA=NO\_REQDATA****,REQDATA=*reqdata***

Use this input parameter with MODE=SYNCEXIT or MODE=ASYNCEXIT to pass any data you choose to the complete exit. The exit will get control only if the request is processed asynchronously.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the data to be passed to the complete exit.

**,REQECB=*reqecb***

Use this output parameter with either MODE=SYNCECB or MODE=ASYNCECB to specify the address of an ECB, which is to be posted when the request completes if the request was processed asynchronously.

Before coding REQECB, you must ensure that:

- You initialize the ECB before you issue the request.
- The ECB resides in either common storage or the home address space where IXLCONN was issued.
- Any tasks that wait for the ECB to be posted reside in the home address space where IXLCONN was issued.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that contains the address of the ECB to be posted when the request completes. The ECB must be aligned on a fullword boundary.

**,REQID=NO\_REQID****,REQID=*reqid***

Use this input parameter to specify a user-defined request identifier to be associated with the request. You can specify this request identifier on the IXLPURGE macro to cancel a request that has not yet been processed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the user-defined request identifier.

**,REQTOKEN=*reqtoken***

Use this output parameter with either MODE=SYNCTOKEN or MODE=ASYNCTOKEN to specify the address of a storage area to receive the request token that is returned when the request will be

processed asynchronously. This token, which uniquely identifies the request, must be used as input to the IXLFCOMP macro, which you use to determine if the request has completed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field where the system will put the request token.

**,RETCODE=retcode**

Use this output parameter to specify a field to contain the return code. (The return code is also returned in register 15.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the return code when the request has completed.

**,RSNCODE=rsncode**

Use this output parameter to specify a field to contain the reason code returned, if applicable. (The reason code is also returned in register 0.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the reason code (if any) when the request has completed.

## ABEND Codes

---

Abend X'026' (See [z/OS MVS Programming: Sysplex Services Guide](#) for more information on this abend.)

## Return and Reason Codes

---

When the system returns control to your program:

- GPR 15 (and RETCODE, if specified) contains a return code.
- If the return code is not zero, GPR 0 (and RSNCODE, if specified) contains a reason code.

**Note:** The return and reason codes will also be put into the answer area (mapped by IXLYCAA), if there is one, when the request completes.

The IXLYCON macro provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

**0**

IXLRETCODEOK

**4**

IXLRETCODEWARNING

**8**

IXLRETCODEPARMERROR

**C**

IXLRETCODEENVERROR

**10**

IXLRETCODECOMPERROR

The following table contains hexadecimal return and reason codes, the equate symbols associated with each reason code, and the meaning and suggested action for each return and reason code.

Table 45. Return and Reason Codes for the IXLCACHE REQUEST=UNLOCK\_CO\_NAME Macro

Hexadecimal Return Code	Hexadecimal Reason Cod	Equate Symbol Meaning and Action
0	None.	<p><b>Meaning:</b> If you specified a MODE value of ASYNCECB, ASYNCEXIT, or ASYNCTOKEN, the request has been successfully initiated. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed. For any other MODE value, the request has successfully completed.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=ASYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=ASYNCEXIT, the connection's complete exit will be given control when the request has completed.</li> <li>• If you specified MODE=ASYNCTOKEN, use REQTOKEN when calling IXLFComp to determine when the request has completed.</li> </ul>
4	xxxx0402	<p><b>Equate Symbol:</b> IXLRNCODEASYNCH</p> <p><b>Meaning:</b> The request could not be processed synchronously. It will be processed asynchronously. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed.</p> <ul style="list-style-type: none"> <li>• If you specified MODE=SYNCECB, the ECB specified in REQECB will be posted when the request has finished.</li> <li>• If you specified MODE=SYNCEXIT, the connection's complete exit will be given control when the request has finished.</li> <li>• If you specified MODE=SYNCTOKEN, the REQTOKEN contains an asynchronous request token that may be used on the IXLFComp macro to determine when the request has completed.</li> </ul> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=SYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=SYNCEXIT, the connection's completion exit will be given control when the request has completed.</li> <li>• If you specified MODE=SYNCTOKEN, use REQTOKEN when calling IXLFComp to determine when the request has completed.</li> </ul>

Table 45. Return and Reason Codes for the IXLCACHE REQUEST=UNLOCK\_CO\_NAME Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Cod	Equate Symbol Meaning and Action
8	xxxx0801	<p><b>Equate Symbol:</b> IXLRSNCODEBADPARMLIST</p> <p><b>Meaning:</b> The parameter list (mfctrl) for this request is not addressable. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the parameter list address.</li> <li>• The parameter list must be addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are issuing this macro while disabled, the parameter list must reside in either page-fixed or DREF storage.</li> <li>• If you are issuing this macro in AR-mode and you specified the parameter list address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are issuing this macro in AR-mode, SYSSTATE ASCENV=AR must be issued before this macro.</li> </ul>
8	xxxx0804	<p><b>Equate Symbol:</b> IXLRSNCODEBADVERSION#</p> <p><b>Meaning:</b> The version number in the macro parameter list is not compatible with the level of XES currently being used. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that your program did not overlay the parameter list storage.</li> <li>• Verify that your program was assembled with the correct macro library for the release of MVS on which your program is running.</li> <li>• Verify that your program is running on an MVS system that supports the version of the macro you are using.</li> </ul>



Table 45. Return and Reason Codes for the IXLCACHE REQUEST=UNLOCK\_CO\_NAME Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Cod	Equate Symbol Meaning and Action
8	xxxx080A	<p><b>Equate Symbol:</b> IXLRNCODEBADCONTOKEN</p> <p><b>Meaning:</b> Program error. The specified connect token (CONTOKEN) was not valid for one of the following reasons:</p> <ol style="list-style-type: none"> <li>1. The user with the connection identifier represented by the token has disconnected from the structure.</li> <li>2. The connector's task (the task that issued IXLCONN) ended.</li> <li>3. The specified token is not the token that was returned from IXLCONN.</li> <li>4. The request was issued from an address space other than the address space in which IXLCONN was issued.</li> <li>5. The connect token was invalidated during rebuild.</li> <li>6. The connect token was invalidated by XES.</li> </ol> <p><b>Note:</b> The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Verify that the CONTOKEN value specified was valid and for the correct structure. The numbers on the following actions correspond to the numbers above.</p> <ol style="list-style-type: none"> <li>1. Discontinue use. Perform recovery and cleanup for the structure and check protocol for use of structure to determine why you did not know of the disconnection.</li> <li>2. Discontinue use. Perform recovery and cleanup for the structure and check protocol for use of structure to determine why you did not know that the task ended.</li> <li>3. Issue IXLCONN to connect to the structure. The contoken is returned in the CONACONTOKEN field of the answer area specified on the IXLCONN request.</li> <li>4. Issue your request from the same address space the IXLCONN was issued.</li> <li>5. Participate in the rebuild. When it is complete, try again.</li> <li>6. Discontinue use. Perform recovery and cleanup for the structure and check protocol for use of structure.</li> </ol> <p>You may want to issue IXCQUERY to get more information about the structure.</p>

Table 45. Return and Reason Codes for the IXLCACHE REQUEST=UNLOCK\_CO\_NAME Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Cod	Equate Symbol Meaning and Action
8	xxxx081E	<p><b>Equate Symbol:</b> IXLRNCODEBADCOLOCKSTATE</p> <p><b>Meaning:</b> You requested that an entry named in CUNBAREA have its change bit overridden to indicate that the entry contains changed data, but this option is incompatible with the current cast-out lock state value, "write with castout." The "write with castout" state is entered when the cast-out lock is obtained by a WRITE_DATA request specifying GETCOLOCK=YES. The following fields are returned in the answer area:</p> <ul style="list-style-type: none"> <li>• CAACOLCKSTATE</li> <li>• CAACOLCKVAL</li> </ul> <p>For a description of cast-out lock state values, see mapping macro IXLYCAA.</p> <p><b>Action:</b> The lock state and value are returned in the answer area (ANSAREA).</p> <ul style="list-style-type: none"> <li>• Determine who holds the lock for this data item.</li> <li>• Verify the name in the CUNBAREA you have passed.</li> </ul>
8	xxxx0824	<p><b>Equate Symbol:</b> IXLRNCODEWRONGSTRTYPE</p> <p><b>Meaning:</b> The connect token specified by CONTOKEN is not to a cache structure.</p> <p><b>Action:</b> Verify the connect token for this cache structure.</p> <p><b>Note:</b> The connect token can be found in the IXLCONN answer area. Verify the value specified on the TYPE= parameter of the IXLCONN request for this structure.</p>
8	xxxx0825	<p><b>Equate Symbol:</b> IXLRNCODENOENTRY</p> <p><b>Meaning:</b> Program error. The request failed because the entry name identified in CUNBAREA specified an entry name that is not in the cache structure.</p> <p><b>Action:</b> Reissue the request specifying in CUNBAREA the valid entry name to be processed.</p>
8	xxxx0829	<p><b>Equate Symbol:</b> IXLRNCODEBADUNLOCKVAL</p> <p><b>Meaning:</b> Program error. The entry name identified in CUNBAREA did not have the castout lock held for the connection specified by CONTOKEN and/or PROCESSID. The lock state (field CAACOLCKSTATE) and lock value (field CAACOLCKVAL) for the element are returned in the answer area (ANSAREA).</p> <p><b>Action:</b> Verify that the CONTOKEN and PROCESSID specifications are valid.</p>

Table 45. Return and Reason Codes for the IXLCACHE REQUEST=UNLOCK\_CO\_NAME Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Cod	Equate Symbol Meaning and Action
8	xxxx082F	<p><b>Equate Symbol:</b> IXLRNCODEBADPARITY</p> <p><b>Meaning:</b> Program error. The parity value specified in the write-operation-block was not valid. The data is not written. The index of the write-operation-block that failed and the offset in the data block of the data area for the write-operation-block being processed are returned in the answer area (fields CAAWDLINDEX and CAAWDLDATAOFFSET). All prior write-operation-blocks were processed.</p> <p><b>Action:</b> The value specified in the request contained parity bits that were not valid. The value for the parity bits may be 00, 01, or 11 in bits 2 and 3 of the PARITY specification (xxpp xxxx where pp is the parity bits).</p>
8	xxxx0838	<p><b>Equate Symbol:</b> IXLRNCODEBADANSAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by ANSAREA is not addressable.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the ANSAREA address.</li> <li>• ANSAREA must be addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are calling IXLCACHE while disabled, ANSAREA must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLCACHE in AR-mode and you specified the ANSAREA address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are calling IXLCACHE in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLCACHE macro.</li> </ul>
8	xxxx0839	<p><b>Equate Symbol:</b> IXLRNCODEBADREQTOKENAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by REQTOKEN is not addressable.</p> <p><b>Action:</b> Ensure that the request token area specified by REQTOKEN is valid:</p> <ul style="list-style-type: none"> <li>• Verify the REQTOKEN address.</li> <li>• If you are calling IXLCACHE while disabled, REQTOKEN must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLCACHE in AR-mode and you specified the REQTOKEN address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are calling IXLCACHE in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLCACHE macro.</li> </ul>

Table 45. Return and Reason Codes for the IXLCACHE REQUEST=UNLOCK\_CO\_NAME Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Cod	Equate Symbol Meaning and Action
8	xxxx083D	<p><b>Equate Symbol:</b> IXLRSNCODEBADANSLEN</p> <p><b>Meaning:</b> Program error. The length of the answer area, as specified by ANSLEN, is not sufficient to contain answer area information. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Increase the size of the answer area provided for the request, and rerun your program. Depending on the macro version number, specify the length of the answer area as follows:</p> <ul style="list-style-type: none"> <li>• When the connection specified ASYNCXI=1 on the IXLCONN invocation and AXIOVERRIDE=0 was specified or defaulted to for the IXLCACHE request, the answer area length must be a minimum value of CAALEVEL2LEN</li> <li>• When the value of the macro version number (PLISTVER) is 4 or more, the minimum answer area length is CAALEVEL1LEN.</li> <li>• Wehn the value of the macro version number (PLISTVER) is 0-3, the minimum answer area length is CAALEVEL0LEN.</li> </ul>
8	xxxx0851	<p><b>Equate Symbol:</b> IXLRSNCODENOSUSPENDISABLE</p> <p><b>Meaning:</b> Program error. The request failed because MODE=SYNCSUSPEND was specified, but the caller is disabled and cannot be suspended.</p> <p><b>Action:</b> Either specify another MODE value or become enabled (release the CPU lock); then reissue the request.</p>
C	xxxx0C06	<p><b>Equate Symbol:</b> IXLRSNCODENOCONN</p> <p><b>Meaning:</b> Environmental error. No connectivity to the cache structure exists. This may occur due to operator commands such as VARY PATH,OFFLINE or CONFIG CHP,OFFLINE or hardware errors such as facility or path failures. The CONTOKEN is invalidated. The request fails.</p> <p><b>Action:</b> Either disconnect from the structure (using IXLDISC) or initiate a rebuild of the structure, if allowed (using IXLREBLD).</p>

Table 45. Return and Reason Codes for the IXLCACHE REQUEST=UNLOCK\_CO\_NAME Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Cod	Equate Symbol Meaning and Action
C	xxxx0C13	<p><b>Equate Symbol:</b> IXLRSNCODEREQPURGED</p> <p><b>Meaning:</b> Environmental error. The request was purged prior to completion of the request. Possible reasons include:</p> <ul style="list-style-type: none"> <li>• The connector failed.</li> <li>• The connector disconnected.</li> <li>• The requestor failed.</li> <li>• The request was purged by IXLPURGE.</li> <li>• Requests were purged when the connector provided an IXLEERSP response for the Rebuild Quiesce, Rebuild Stop, or Rebuild Cleanup event.</li> <li>• The connector invoked IXLREBLD REQUEST=COMPLETE.</li> <li>• The secondary address space was no longer valid.</li> </ul> <p><b>Action:</b> None if this is expected. Otherwise, determine why the connector failed.</p>
C	xxxx0C14	<p><b>Equate Symbol:</b> IXLRSNCODESTATUSUNKNOWN</p> <p><b>Meaning:</b> Environmental error. The IXLCACHE request has completed, but the final disposition of the request cannot be determined. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Verify the validity of your data by comparing the expected results with what is in the coupling facility.</p>
C	xxxx0C25	<p><b>Equate Symbol:</b> IXLRSNCODESTRFAILURE</p> <p><b>Meaning:</b> Environmental error. The cache structure failed prior to completion of the request.</p> <p><b>Action:</b> Attempt to rebuild the structure using IXLREBLD, or disconnect from the structure using IXLDISC.</p>
C	xxxx0C68	<p><b>Equate Symbol:</b> IXLRSNCODEBADREQCFLEVEL</p> <p><b>Meaning:</b> Environmental error. The request type is not permitted for the level of coupling facility in which the target structure is allocated.</p> <p><b>Action:</b> Either continue processing using an UNLOCK_CASTOUT request to perform the function of the UNLOCK_CO_NAME request or disconnect from the structure (using IXLDISC) and request that the installation provide a coupling facility of the correct CFLEVEL (CFLEVEL=4 or higher).</p>
C	xxxx0CA0	<p><b>Equate Symbol:</b> IXLRSNCODEQUIESCEDSUSPENDFAIL</p> <p><b>Meaning:</b> Environmental error. The request is failed because the structure is quiesced for a system-managed process and SUSPEND=FAIL is specified on the IXLCONN.</p> <p><b>Action:</b> None, if this is expected.</p>

Table 45. Return and Reason Codes for the IXLCACHE REQUEST=UNLOCK\_CO\_NAME Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Cod	Equate Symbol Meaning and Action
C	xxxxFFFF	<p><b>Equate Symbol:</b> IXLRSNCODENOTAVAILABLE</p> <p><b>Meaning:</b> Environmental error. XES functions are not available. The coupling facility hardware might not be present.</p> <p><b>Action:</b> XES may not be used when XCF is running local mode. XES requires that the coupling facility be installed.</p>
10	xxxx10xx	<p><b>Equate Symbol:</b> IXLRSNCODECOMPERROR</p> <p><b>Meaning:</b> System error. XES processing failure. The state of the structure and the disposition of the request are unpredictable. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Save the reason code information, and contact the IBM support center.</p>

## Chapter 48. IXLCACHE REQUEST=WRITE\_DATA

### Description

A WRITE\_DATA request allows you to write data from the storage areas specified by BUFFER or BUFLIST and/or ADJAREA to a data entry in the cache structure. (A data entry is storage in the coupling facility cache structure comprised of one or more data elements where the system stores a data item. For more information, see *z/OS MVS Programming: Sysplex Services Guide*.) The NAME parameter identifies the data item that will be, or has already been, introduced to the structure. Once introduced to the structure, NAME also references the structure resources (the directory entry and data entry) allocated for that data item.

You can use the WRITE\_DATA request to either:

- Create a new data entry to contain the data item. (A directory entry would also be allocated.)
- Update the contents of an existing data entry with the new or changed data item.

To register or update your interest in a data item, you must specify an index into your local cache vector (VECTORINDEX) to be associated with the data item in the structure. This index identifies a vector entry that cache services uses to indicate both your interest in the data item and the validity of your locally cached copy of the data item. Having registered interest in a data item **means** that you have a valid copy of the data item in your local cache buffer. When another user updates the data item in the structure, cache services will update the data item's associated vector entry, thereby deregistering your interest in the data item and invalidating your locally cached copy.

Cache services is responsible for deregistering your interest in a data item whenever another user updates the data item in the structure. You are responsible for maintaining the association between the entry in your local cache vector and the copy of the data item in your local cache buffer. You are also responsible for upholding any serialization protocols and procedures established to manage access to this shared data.

For structures allocated in a coupling facility of CFLEVEL=5 or higher, you can request that interest be deregistered in the entry specified by OLDNAME without registering interest in the entry specified by NAME.

The WHENREG parameter allows you to specify that the data be written only if you have already registered your interest in the data item: (To have a registered interest in the data item, the entry must already exist in the structure.)

- If you specify WHENREG=YES without VECTORINDEX, your interest must already be registered. (To have a registered interest in the data item, the entry must already exist in the structure and the vector index must be the same as that with which you registered interest.)
- If you specify WHENREG=YES with VECTORINDEX (valid only for structures in a CFLEVEL=2 and higher coupling facility), your interest must already be registered **and** the vector index specified must be the same as that with which you are currently registered. When VECTORINDEX is specified and the vector index does not equal the vector index with which you currently have a registered interest in the data item, the system fails the WRITE\_DATA request and returns the vector index with which you are currently registered in the cache answer area.
- If you specify WHENREG=NO, the data item is written regardless of whether your interest is already registered. For a structure llocated in a coupling facility of CFLEVEL=4 or lower:
  - If the entry exists in the structure, your interest is updated.
  - If the entry does not exist in the structure, it is created and your interest is registered.

VECTORINDEX is required when WHENREG=NO is specified for structures allocated in a coupling facility of CFLEVEL=4 or lower.

Optionally, for structures allocated in a coupling facility of CFLEVEL=5 or higher, you can use the REGUSER keyword to specify whether interest in the entry is to be registered or not.

- If REGUSER=YES is specified, interest in the entry is registered. When registration is performed, if connection interest is already registered, the specified VECTORINDEX replaces any previously specified local cache vector index for the entry.

VECTORINDEX is required when REGUSER=YES is either specified or selected by default.

- If REGUSER=NO is specified, interest in the entry will not be registered. Be aware that entries in a data item for which no interest has been registered are higher-priority candidates for reclaim processing than those for which interest is registered.

VECTORINDEX is required when REGUSER=NO is specified along with OLDNAME and is ignored if REGUSER=NO is specified without OLDNAME. When OLDNAME is also specified with VECTORINDEX, interest is deregistered in the entry specified by OLDNAME.

You can write either changed (CHANGED=YES) or unchanged (CHANGED=NO) data to the cache structure. If you write changed data (cache copy is different from permanent storage copy) you must assign the data item to a cast-out class (COCLASS). You must assign both changed and unchanged data items to a storage class (STGCLASS).

For structures allocated in a coupling facility with CFLEVEL=3 or lower, BUFFER or BUFLIST is required and ELEMNUM must be 1 or greater. For structures allocated in a coupling facility with CFLEVEL=4 or higher,

- When CHANGED=YES, BUFFER or BUFLIST is required and ELEMNUM must be 1 or greater.
- When CHANGED=NO, neither BUFFER nor BUFLIST is required and ELEMNUM can be 0 or greater.

You also have the option to invalidate other users' local copies of the data (CROSSINVAL) and to obtain the cast-out lock (GETCOLOCK), if necessary.

For structures allocated in a coupling facility of CFLEVEL=5 or higher, the VERSCOMP keyword allows you to specify a value that is to be compared with the version number associated with the entry being processed. The VERSCOMPTYPE keyword is used to define how the comparison is to be performed. If a version comparison mismatch occurs, processing terminates with no change to the structure. If a version comparison is successful, you can update (or choose not to update) the value of the entry version number by specifying the VERSUPDATE keyword.

For structures allocated in a coupling facility of CFLEVEL=9 or higher, the ASSIGN keyword on an IXLCACHE REQUEST=WRITE\_DATA,WHENREG=NO request allows you to specify that the system is to suppress the creation of a new directory entry when an existing directory entry is not found.

For more information, see *z/OS MVS Programming: Sysplex Services Guide*.

## Syntax Diagram

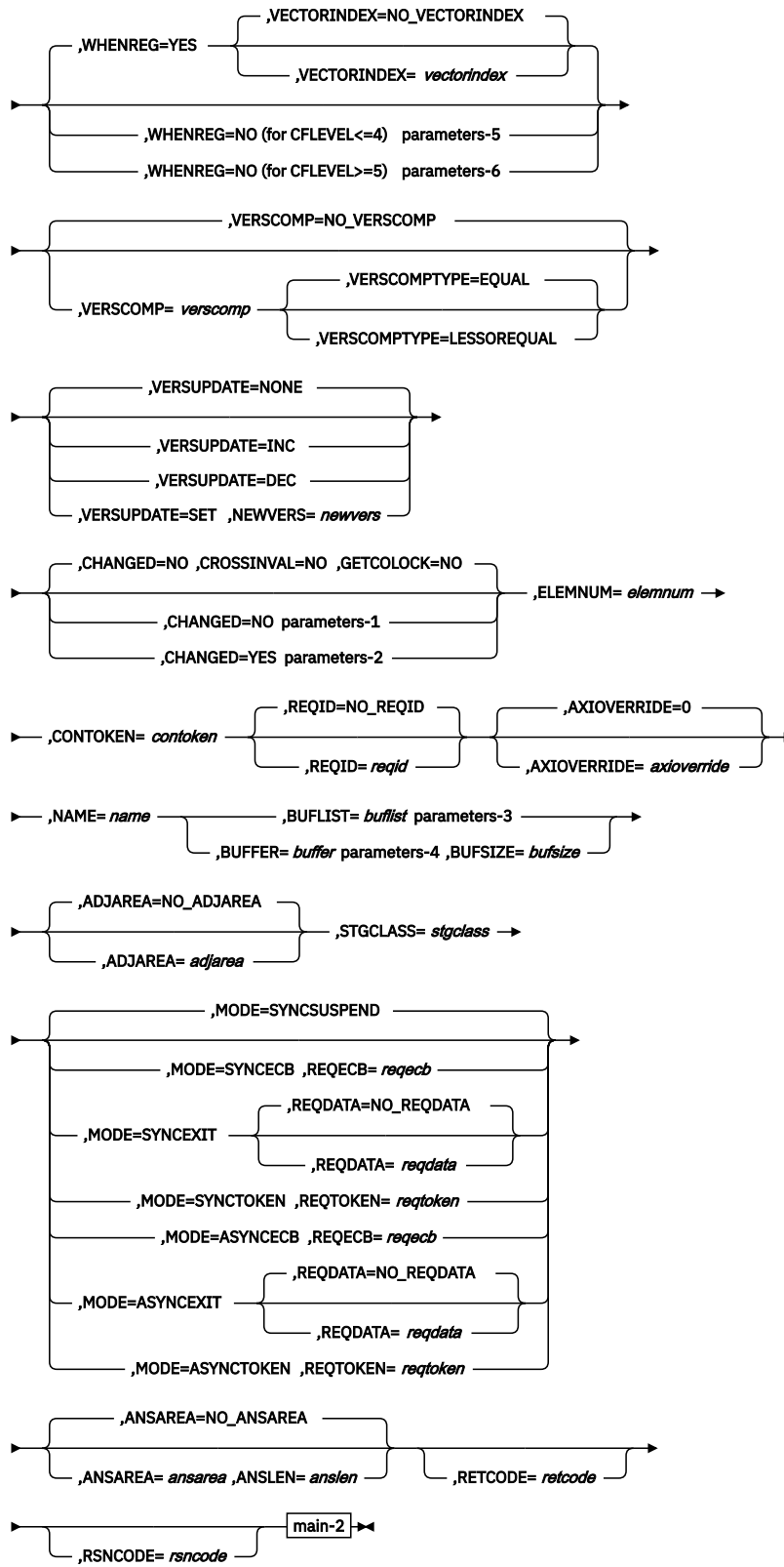
---

The syntax diagram for IXLCACHE REQUEST=WRITE\_DATA is as follows:

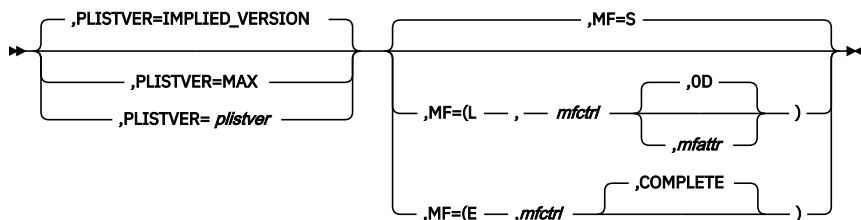


### main diagram

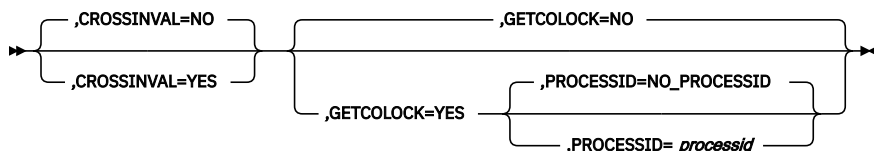
➡ IXLCACHE — REQUEST=WRITE\_DATA ➡



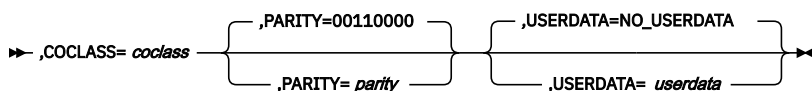
## main-2



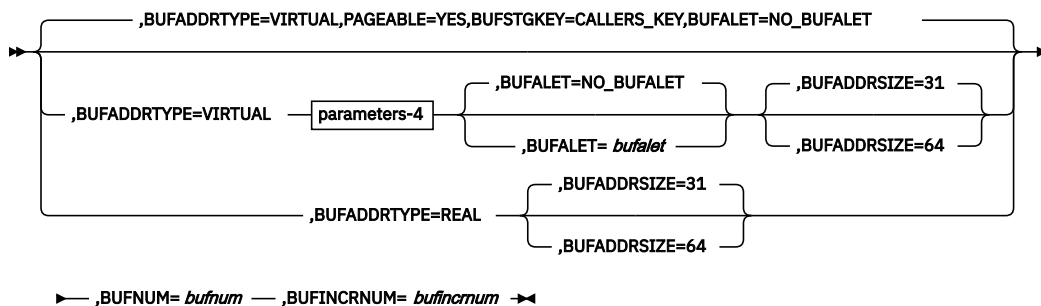
## parameters-1



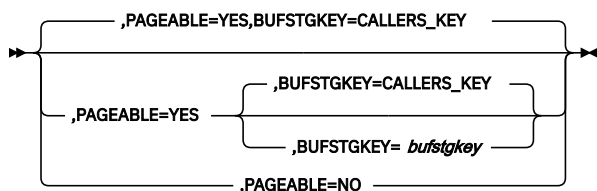
## parameters-2



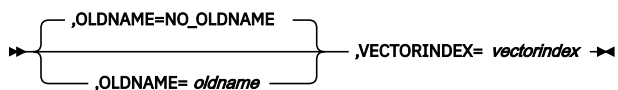
## parameters-3

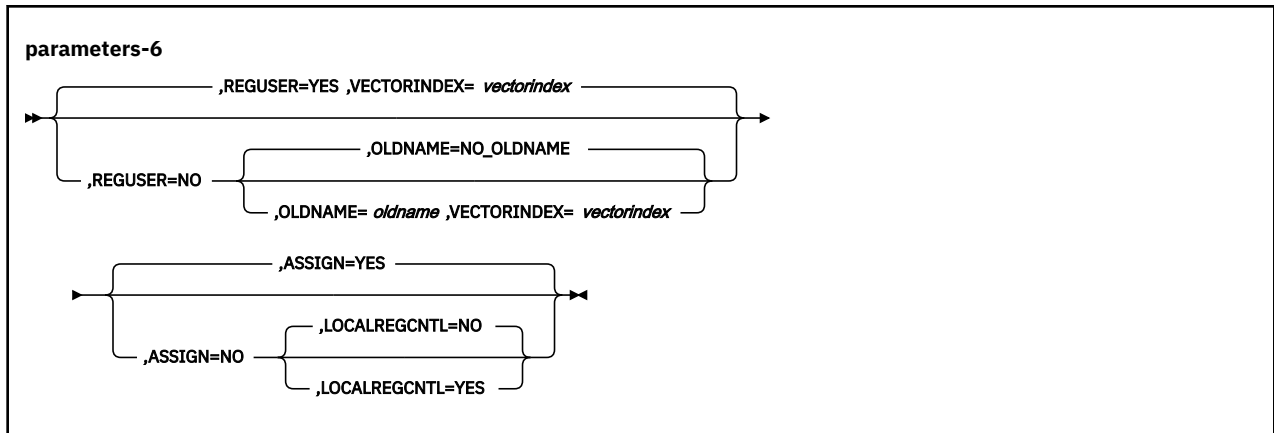


## parameters-4



## parameters-5





**Note:** You must specify `ANSAREA=ansarea`, `ANSLEN=anslen` if you:

- Specify `MODE=SYNCTOKEN` or `MODE=ASYNCTOKEN`, or
- Specify `AXIOVERRIDE=0` (or default to 0) and specified `ASYNCXI=1` on the `IXLCONN` invocation when connecting to the cache structure.

## Parameter Descriptions

The parameter descriptions for `REQUEST=WRITE_DATA` are listed in alphabetical order. Default values are underlined:

### **REQUEST=WRITE\_DATA**

Use this input parameter to specify that the data item identified by `NAME` be written from the areas specified by `BUFFER` or `BUFLIST`, and/or `ADJAREA` to the cache structure.

#### **,ADJAREA=NO\_ADJAREA**

#### **,ADJAREA=adjarea**

Use this input parameter to specify a storage area to contain the adjunct data to be written to an entry in the cache structure. Specify `ADJAREA` only when `ELENUM` is greater than 0.

If the structure supports adjunct data and `ADJAREA` is not specified or `ADJAREA=NO_ADJAREA` is specified, binary zeros are written to the adjunct area. If the structure does not support adjunct data and `ADJAREA` is specified, `ADJAREA` is ignored. (Adjunct areas for a structure are established through the `IXLCONN` macro.)

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 64-byte area that contains the adjunct data you want associated with this data entry.

#### **,ANSAREA=NO\_ANSAREA**

#### **,ANSAREA=ansarea**

Use this output parameter to specify an answer area to contain information returned from the request. The format of the answer area is described by the `IXLYCAA` mapping macro. On a successful completion of a request, the following information is returned in the answer area:

- If changed data is written to the cache, the total number of entries containing either changed or locked-for-cast-out data, that are assigned to the storage class to which the entry was written, is returned (field `CAATOTCHANGED`).
- If changed data is written to the cache, the total number of data items assigned to the cast-out class to which data was just written is returned (field `CAACOCOUNT`).
- If `WHENREG=NO` was specified and the request replaced a previously specified vector index for the entry, the replaced vector index is returned (fields `CAAINVLCVINUM` and `CAAINVLCVI`).

The following additional field is returned in the answer area when the connection specified `ASYNCXI=1` on the `IXLCONN` invocation when connecting to the cache structure and the cache structure is allocated in a `CFLEVEL=23` or higher coupling facility:

- An asynchronous cross-invalidation sequence number (CAAASYNXISEQNUM) if cross-invalidations against local caches for this request were initiated asynchronously to the completion of the request.

See Chapter 29, “IXLAXISN,” on page 449 for a description of how to use the returned CAAASYNXISEQNUM to determine when cross-invalidates of local caches associated with the request have completed.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an area (with a length of ANSLEN) where the information returned from the request may be stored.

**,ANSLEN=anslen**

Use this inppaameter to specify the size of the storage area specified by ANSAREA.

Use either CAALEVELOLEN, CAALEVEL1LEN or CAALEVEL2LEN of the IXLYCAA mapping macro to determine the minimum size of the answer area. The answer area length must be at least large enough to accommodate the level of the IXLYCAA mapping appropriate to the requested function.

- When the connection specified ASYNCXI=1 on the IXLCONN invocation and AXIOVERRIDE=0 was specified or defaulted to for the IXLCACHE request, the answer area length is a required parameter and must be a minimum value of CAALEVEL2LEN to contain a returned asynchronous cross-invalidation sequence number (CAAASYNXISEQNUM).
- When the value of PLISTVER is 4 or above, the minimum answer area length is CAALEVEL1LEN.
- When the value of PLISTVER is 0 - 3, the minimum answer area length is CAALEVELOLEN.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12 2 to ) of a 2-byte field that contains the length of the answer area (ANSAREA).

**,ASSIGN=YES**

**,ASSIGN=NO**

Use this input parameter to specify whether a directory entry will be assigned for NAME if one does not currently exist.

**YES**

A directory entry will be assigned.

**NO**

No directory entry will be assigned.

ASSIGN=NO is meaningful only for cache structures allocated in a coupling facility of CFLEVEL=9 or higher.

**,AXIOVERRIDE=0**

**,AXIOVERRIDE=axioveride**

Use this input parameter to specify whether the asynchronous cross-invalidation control setting of IxlConnAsyncXiYes (1) for the connection identified by CONTOKEN should be overridden for this request. Valid values are 0 (IxlCacheAXiOverrideNo) or 1 (IxlCacheAXiOverrideYes).

The AXIOVERRIDE keyword is meaningful to processing only when the connection specified ASYNCXI=1 on the IXLCONN invocation when connecting to the cache structure and the cache structure is allocated in a CFLEVEL=23 or higher coupling facility.

A value of 0 (IxlCacheAXiOverrideNo) indicates that the asynchronous cross-invalidation control for the connection as specified on the IXLCONN invocation should be used for the request. Cross-invalidates against local caches for this request will preferentially be initiated asynchronously to the completion of the request when asynchronous cross-invalidations are supported by the coupling facility where the cache structure is allocated.

A value of 1 (IxlCacheAXiOverrideYes) indicates that the ASYNCXI specification of IxlConnAsyncXiYes (1) by the connector on the IXLCONN invocation should be overridden for this request only. Cross-invalidations generated by this request will be processed synchronously to the completion of the request.

Any value other than 0 or 1 for AXIOVERRIDE will have the same behavior as specifying a value of 0 (IxlCacheAXiOverrideNo).

If cross-invalidation against local caches for this request were initiated asynchronously to the completion of the request, an asynchronous cross-invalidation sequence number is returned in CAAASYNCXISEQNUM of the cache answer area (ANSAREA).

The asynchronous cross-invalidation sequence number can be used on a subsequent invocation of IXLAXISN to ensure that the asynchronous cross-invalidation associated with this request have completed.

When cross-invalidation is initiated synchronously to the completion of the request or no cross-invalidation occurred for the request, no asynchronous cross-invalidation sequence number is returned.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12 to ) of a one-byte input field that contains the value indicating whether the asynchronous cross-invalidation control setting of the connector should be overridden

**,BUFADDRSIZE=31**

**,BUFADDRSIZE=64**

Use this input parameter to specify whether a 31-bit or a 64-bit address is specified by a BUFLIST entry.

**31**

The entry in BUFLIST is 31 bits in size.

**64**

The entry in BUFLIST is 64 bits in size.

**,BUFADDRTYPE=VIRTUAL**

**,BUFADDRTYPE=REAL**

Use this input parameter to specify whether the buffer addresses specified in the BUFLIST list are virtual storage or real storage addresses.

**VIRTUAL**

The buffer addresses are virtual storage addresses. The virtual storage can be pageable or nonpageable. See the PAGEABLE parameter for information about managing storage binds when specifying virtual storage addresses.

**REAL**

The buffer addresses are real storage addresses.

It is the caller's responsibility to manage the binds between the data buffer virtual storage and the real storage addresses provided. The caller must ensure that the data buffer virtual storage remains bound to the real storage addresses provided until the request completes.

**,BUFALET=NO\_BUFALET**

**,BUFALET=bufalet**

Use this input parameter to specify an access list entry token (ALET) to be used in referencing all of the buffers specified by BUFLIST.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 4-byte field that contains the ALET.

**,BUFFER=buffer**

Use this input parameter to specify a buffer area to hold the entry data to be written to the cache structure. Only 31-bit addressable virtual storage areas (below 2GB) are supported by the BUFFER specification. High virtual storage areas (above 2GB) can only be specified via the BUFLIST specification.

You can define the buffer size to be a total of up to 65536 bytes. Other requirements depend on the size you select:

- If you specify a buffer size of less than or equal to 4096 bytes, you must ensure that the buffer:
  - Is 256, 512, 1024, 2048, or 4096 bytes.
  - Starts on a 256-byte boundary.

- Does not cross a 4096-byte boundary.
- Does not start below storage address 512.
- If you specify a buffer size of greater than 4096 bytes, you must ensure that the buffer:
  - Is a multiple of 4096 bytes.
  - Is less than or equal to 65536 bytes.
  - Starts on a 4096-byte boundary.
  - Does not start below storage address 512.

See the BUFSIZE parameter description for defining the size of the buffer.

See *z/OS MVS Programming: Sysplex Services Guide* for more information on buffers.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an area (with a length of BUFSIZE) to contain the data to be written to the entry in the structure.

**,BUFINCRNUM=bufincrnum**

Use this input parameter to specify the number of 256-byte segments comprising each buffer in the BUFLIST list.

Valid BUFINCRNUM values are 1, 2, 4, 8, or 16, which correspond to BUFLIST buffer sizes of 256, 512, 1024, 2048, and 4096 bytes respectively.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 1-byte field that contains 1, 2, 4, 8, or 16.

**,BUFLIST=buflist**

Use this input parameter to specify a list of buffers to hold the entry data to be written to the cache structure. BUFLIST specifies a 128-byte storage area that consists of a list of 0 to 16 elements.

Either 31-bit addressable (below 2GB) or 64-bit addressable (above 2GB) real or virtual storage areas are supported for the BUFLIST specification, depending on the specifications for the BUFADDRTYPE and BUFADDRSIZE keywords. However, pageable high shared virtual storage areas (above 2GB) may not be used.

The format of the list is a set of 8-byte elements. The BUFADDRSIZE keyword denotes whether four or eight bytes of the element are used.

- If BUFADDRSIZE=31 is specified, then the first four bytes of each element are reserved space and the last four bytes contain the address of the buffer.
- If BUFADDRSIZE=64 is specified, then the full eight bytes specify the address of the buffer.

**The BUFLIST buffers must:**

- All reside in the same address space or data space as defined by BUFALET.
- Be the same size: either 256, 512, 1024, 2048, or 4096 bytes as defined by BUFINCRNUM.
- Start on a 256-byte boundary and not cross a 4096-byte boundary.
- Not start below storage address 512.

**Note:** The buffers do not have to be contiguous in storage. XES treats BUFLIST buffers as a single buffer even if the buffers are not contiguous.

See the BUFNUM and BUFINCRNUM keyword descriptions for specifying the number and size of buffers.

For structures allocated in CFLEVEL=0 through CFLEVEL=3 coupling facilities, one of BUFFER or BUFLIST is required. For structures allocated in CFLEVEL=4 or higher coupling facilities, one of BUFFER or BUFLIST is required when CHANGED=YES and optional when CHANGED=NO.

See *z/OS MVS Programming: Sysplex Services Guide* for more information on buffers.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 128-byte storage area that contains a list of buffer addresses.

**,BUFNUM=*bufnum***

Use this input parameter to specify the number of buffers in the BUFLIST list. Valid BUFNUM values are from 0 to 16. A value of zero indicates that no data is to be written to the entry. If you do not want buffers, but want to specify this parameter, code BUFNUM=NO\_BUFNUM.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 1-byte field that contains the number of buffers (0 to 16) in the buffer list.

**,BUFSIZE=*bufsize***

Use this input parameter to specify the size of the BUFFER area. See the BUFFER parameter description for valid buffer sizes.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a fullword field that contains the size of the buffer (BUFFER) in bytes.

**,BUFSTGKEY=CALLERS\_KEY****,BUFSTGKEY=*bufstgkey***

Use this input parameter to specify a storage key that you define and use when referencing the buffers specified by BUFLIST or the buffer that is specified by BUFFER.

If you do not specify BUFSTGKEY, or if you specify BUFSTGKEY=CALLERS\_KEY, all references to one or more buffers are performed by using the caller's PSW key.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of an 8-bit field that contains the storage key in the format B'kkkkxxxx', where kkkk is the key and xxxx is ignored.

**,CHANGED=NO****,CHANGED=YES**

Use this input parameter to specify whether changed data is to be written to an entry in the cache structure.

**NO**

The data to be written is unchanged:

- The cached copy is the same as the permanent storage copy.
- In a store-through process, the permanent storage copy will be updated (by the user) to match the cached copy, either before the cast-out lock is released (if GETCOLOCK=YES was specified), or before serialization on the named data item is released.

If the entry already exists in the structure, the entry must contain unchanged data or the request will fail. You cannot overwrite changed data with unchanged data.

For structures allocated in CFLEVEL=4 or higher coupling facilities, ELEMNUM=0 and one of BUFFER and BUFLIST are optional when CHANGED=NO is specified.

**YES**

The data to be written is changed. The changed data will be assigned to the specified cast-out class (COCLASS) superseding any previously specified cast-out class for the data. With the exception of your connection, all users with registered interest in the data will have their interest deregistered such that their locally cached copies of the data are invalidated.

**,COCLASS=*coclass***

Use this input parameter to specify the cast-out class to which the data item being written is to be assigned. Any previous assignment is updated to the new specification.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 2-byte field that contains the cast-out class value.

**,CONTO=*conken***

Use this input parameter to specify the connect token that was returned by the IXLCONN service in the IXLCONN answer area, which is mapped which is by IXLYCONA. The connect token uniquely identifies your connection to the cache structure, and must be specified on each IXLCACHE invocation.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field that contains the connect token.

**,CROSSINVAL=NO**

**,CROSSINVAL=YES**

Use this input parameter to specify whether cross-invalidate processing should be performed when writing unchanged data.

**NO**

Cross-invalidate processing is not performed. All users with registered interest in the data item remain registered, and the copies of the data item in their local cache buffers remain valid.

**YES**

Cross-invalidate processing is performed. All users with registered interest in the data item, with the exception of your connection, have their interest deregistered and the copies of the data item in their local cache buffers invalidated.

**,ELEMNUM=*elemnum***

Use this input parameter to specify the number of elements to be allocated to the data entry. Valid ELEMNUM values are from 0 to the MAXELEMNUM value that was returned to the connector in the connect answer area. For structures allocated in a CFLEVEL=0 coupling facility, ELEMNUM values can be in the range of 1 to 16. For structures allocated in coupling facilities with a CFLEVEL between CFLEVEL=1 and CFLEVEL=3, ELEMNUM values can be in the range of 1 to 255. For structures allocated in coupling facilities with a CFLEVEL=4 or higher, ELEMNUM values can be in the range of 0 to 255 where 0 is valid only when CHANGED=NO.

Considerations for specifying ELEMNUM are:

- If this request creates a new entry, the number of elements is set to the ELEMNUM value.
- If the entry already exists, the number of elements is updated to this ELEMNUM value.

If the data passed in BUFFER/BUFLIST does not fill up all the space in the elements, the extra space will be padded with zeros. If the data passed in BUFFER/BUFLIST is greater than the amount of space in the elements, the data will be truncated.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 1-byte field that contains the number of elements.

**,GETCOLOCK=NO**

**,GETCOLOCK=YES**

Use this input parameter to specify whether the cast-out lock should be obtained when writing unchanged data.

**NO**

The cast-out lock is not obtained.

**YES**

The cast-out lock is obtained.

**,LOCALREGCNTLNO**

**,LOCALREGCNTL=YES**

Use this keyword to specify whether the WRITE\_DATA request should be suppressed (write suppression) when the user's connection (local cache) is the only registered interest in the data item identified by NAME in the cache structure, and no subsystem data (data entry) for the data item is cached.

LOCALREGCNTL=YES is only meaningful when write requests are issued to cache structures allocated in a coupling facility that supports write suppression based on local cache registration. If the cache structure is allocated in a coupling facility that does not support write suppression based on local cache registration, the WRITE\_DATA request will be performed as if LOCALREGCNTL=NO was specified.

**NO**

The WRITE\_DATA request will be performed normally without write suppression.



**YES**

Suppress the WRITE\_DATA request when the user's connection is the only registered interest in the directory entry and the data entry does not have cached subsystem data. When the write operation is suppressed, no data is written to the cache structure. The WRITE\_DATA request will complete with a return code IXLRETCODEWARNING, reason code IXLRSNCODELOCALREGWRTSUPPRESS.

If LOCALREGCNTL=YES is requested on a WRITE\_DATA request for a cache structure that is currently in the system-managed Duplex Established phase, the request for write suppression is ignored.

CHANGED=YES is required when LOCALREGCNTL=YES is specified.

When the subsystem data is not written to the cache structure, the local cache buffer contains the only valid copy of the subsystem data. Hardening of the subsystem data to permanent storage should be performed to ensure availability and successful recovery of data in the event of a failure.

When issuing a WRITE\_DATA request with LOCALREGCNTL=YES to write data to a cache structure that is in user-managed duplex mode, the following programming technique should be followed:

- When using registration, assignment or write suppression protocols in the primary structure of a user-managed duplexed structure, unconditional writes of changed data to the secondary structure specifying LOCALREGCNTL=NO should be performed after successful write operations to the primary structure. If the write operation to the primary structure resulted in the write being suppressed, then the write operation should not be issued to the secondary structure.

**,MF=S**

**,MF=(L,mfctrl)**

**,MF=(L,mfctrl,mfattr)**

**,MF=(L,mfctrl,0D)**

**,MF=(M,mfctrl)**

**,MF=(M,mfctrl,COMPLETE)**

**,MF=(M,mfctrl,NOCHECK)**

**,MF=(E,mfctrl)**

**,MF=(E,mfctrl,COMPLETE)**

**,MF=(E,mfctrl,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

**,mfctrl**

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

**,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

**,COMPLETE****,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

**COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if `SMILE=var` were an optional parameter and the default is `SMILE=NO_SMILE` then it would not be documented. However, if the default was `SMILE=:)`, then it would be documented because a value would be the default.

**NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

**,MODE=SYNCSUSPEND****,MODE=SYNCECB****,MODE=SYNCEXIT****,MODE=SYNCTOKEN****,MODE=ASYNCECB****,MODE=ASYNCEXIT****,MODE=ASYNCTOKEN**

Use this input parameter to specify:

- Whether the request is to be performed synchronously or asynchronously
- How you want to be notified of request completion if the request is processed asynchronously.

See *z/OS MVS Programming: Sysplex Services Guide* for more information on understanding synchronous and asynchronous cache operations.

**SYNCSUSPEND**

The request processes synchronously. If necessary, the request is suspended until it can complete synchronously. To use this option, your program must be enabled for I/O and external interrupts.

**SYNCECB**

The request processes synchronously if possible. If the request processes asynchronously, the ECB specified by `REQECB` is posted when the request completes.

**SYNCEXIT**

The request processes synchronously if possible. If the request processes asynchronously, your complete exit is given control when the request completes.

**SYNCTOKEN**

The request processes synchronously if possible. If the request processes asynchronously, an asynchronous request token is returned to the area specified by `REQTOKEN`. Use the returned request token on the `IXLFCOMP` macro to determine whether your request has completed.

**Note:** `ANSAREA` is a required parameter when `MODE=SYNCTOKEN` is specified.

**ASYNCECB**

The request processes asynchronously. The ECB specified by `REQECB` is posted when the request completes.

**ASYNCEXIT**

The request processes asynchronously. Your complete exit is given control when the request completes.

**ASYNCTOKEN**

The request processes asynchronously. An asynchronous request token is returned to the area specified by `REQTOKEN`. Use the returned request token on the `IXLFCOMP` macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=ASYNCTOKEN is specified.

**,NAME=*name***

Use this input parameter to specify the name of the data item to be written to the cache structure.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-byte field that contains the name of the data item.

**,NEWVERS=*newvers***

Use this input parameter to specify the value that is to be assigned to the entry version number.

NEWVERS is meaningful only for structures allocated in a coupling facility of CFLEVEL=5 or higher.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-byte field that contains the value for the entry version number.

**,OLDNAME=NO *OLDNAME***

**,OLDNAME=*oldname***

Use this input parameter to specify the name of the data item for which your interest should be unregistered.

- For structures allocated in a coupling facility of CFLEVEL=4 or lower, you must also specify the name of the data item for which interest is to be registered after the interest is unregistered in OLDNAME. Identify the data item to which interest is to be registered with NAME. The VECTORINDEX currently associated with the entry specified by OLDNAME will be reassigned to the name of the data item specified by NAME.
- For structures allocated in a coupling facility of CFLEVEL=5 or higher, you can specify that interest in the name of the data item specified by OLDNAME is to be unregistered without registering interest in the entry specified by NAME.

In either case, whenever deregistration of interest is requested, VECTORINDEX must be specified.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field that contains the name of the old data item that was associated with VECTORINDEX.

**,PARITY=00110000**

**,PARITY=*parity***

Use this input parameter to update the directory entry parity bits for the data item specified by NAME. The parity bits are updated only when CHANGED=YES is specified and the initial value of the directory entry parity bits is null (that is, B'11'). If these conditions are not met, the PARITY value is ignored.

Bits two and three of the 8-bit PARITY field are the parity bits. Valid parity values are:

- xx00xxxx - Parity equals 0
- xx01xxxx - Parity equals 1
- xx11xxxx - Null (parity is not set)

Note that when a directory entry is created, the parity bits are initialized to the null value.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 1-byte field that contains the parity bits.

**,PAGEABLE=YES**

**,PAGEABLE=NO**

Use this input parameter to identify whether the storage areas specified by BUFFER or BUFLIST are in pageable or potentially pageable storage.

**YES**

Specify this option to indicate that the BUFFER or BUFLIST buffers reside in pageable virtual storage. XES performs the required page fixing to fix the buffers in real storage while the cache or list request transfers data to or from the coupling facility.

This includes storage obtained from pageable subpools, disabled reference (DREF) storage, and may include storage that has the potential to become pageable during the processing of a request. (An example is address space storage owned by any swappable address space, for which a PGSER

FIX has been successfully processed, but for which the owning address space gets swapped during processing of a cache or list request.) This does not include implicitly non-pageable storage (for example, storage obtained from non-pageable subpools).

The system takes responsibility for managing binds to central storage for the duration of the cache or list request, regardless of what address space owns the storage or whether the storage-owning address space is swappable or nonswappable. The storage can be owned by any address space.

High shared virtual storage areas (above 2GB) may not be used.

#### **NO**

Specify this option to indicate that the BUFFER or BUFLIST buffers reside in non-pageable virtual storage. XES does not page fix the buffers in real storage.

This includes implicitly non-pageable storage areas (for example, storage obtained from non-pageable subpools), and may include storage that has the potential to become pageable during the processing of a request (An example is address space storage owned by any swappable address space, for which a PGSER FIX has been successfully processed, but for which the owning address space gets swapped-out during processing of a cache or list request.)

The system takes responsibility for managing binds to central storage for the duration of the cache or list request, if and only if the non-pageable storage is owned by either the requester's address space or the connector's address space. If the storage is owned by any other address space, then the invoker is responsible for ensuring that the virtual storage remains non-pageable for the duration of the request (including the case in which the storage is owned by a swappable address space that is swapped during processing of an IXLCACHE or IXLLIST request). Subject to this consideration, the storage can be owned by any address space. See *z/OS MVS Programming: Sysplex Services Guide*.

#### **,PLISTVER=IMPLIED\_VERSION**

#### **,PLISTVER=MAX**

#### **,PLISTVER=plistver**

Use this input parameter to specify the version of the macro. See “[Understanding IXLCACHE Version Support](#)” on page 461 for a description of the options available with PLISTVER.

#### **,PROCESSID=NO\_PROCESSID**

#### **,PROCESSID=processid**

Use this input parameter to specify a user-defined process identifier to be compared with the cast-out lock along with the connection identifier.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12 2 to 12) of a 1-byte field that contains the user-defined process identifier.

#### **,REGUSER=YES**

#### **,REGUSER=NO**

Use this input parameter to specify, for structures allocated in coupling facilities with CFLEVEL=5 or higher, whether the request should register connection interest in the entry.

#### **YES**

Specify this option to indicate that connection interest registration will be performed.

VECTORINDEX is required when REGUSER=YES is either specified or selected by default.

#### **NO**

Specify this option to indicate that no connection interest will be registered. Be careful when using REGUSER=NO because preference to reclaiming is given to entries that have data but no registered interest.

VECTORINDEX is required when REGUSER=NO is specified along with OLDNAME for the deregistration of interest. VECTORINDEX is ignored when REGUSER=NO is specified without the specification of OLDNAME.

**,REQDATA=NO\_REQDATA****,REQDATA=reqdata**

Use this input parameter with MODE=SYNCEXIT or MODE=ASYNCEXIT to pass any data you choose to the complete exit. The exit will get control only if the request is processed asynchronously.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the data to be passed to the complete exit.

**,REQECB=reqecb**

Use this output parameter with either MODE=SYNCECB or MODE=ASYNCECB to specify the address of an ECB, which is to be posted when the request completes if the request was processed asynchronously.

Before coding REQECB, you must ensure that:

- You initialize the ECB before you issue the request.
- The ECB resides in either common storage or the home address space where IXLCONN was issued.
- Any tasks that wait for the ECB to be posted reside in the home address space where IXLCONN was issued.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that contains the address of the ECB to be posted when the request completes. The ECB must be aligned on a fullword boundary.

**,REQID=NO\_REQID****,REQID=reqid**

Use this input parameter to specify a user-defined request identifier to be associated with the request. You can specify this request identifier on the IXPURGE macro to cancel a request that has not yet been processed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the user-defined request identifier.

**,REQTOKEN=reqtoken**

Use this output parameter with either MODE=SYNCTOKEN or MODE=ASYNCTOKEN to specify the address of a storage area to receive the request token that is returned when the request will be processed asynchronously. This token, which uniquely identifies the request, must be used as input to the IXLFCOMP macro, which you use to determine if the request has completed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field where the system will put the request token.

**,RETCODE=retcode**

Use this output parameter to specify a field to contain the return code. (The return code is also returned in register 15.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the return code when the request has completed.

**,RSNCODE=rsncode**

Use this output parameter to specify a field to contain the reason code returned, if applicable. (The reason code is also returned in register 0.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the reason code (if any) when the request has completed.

**,STGCLASS=stgclass**

Use this input parameter to assign a storage class to the data item being written. Any previous assignment is updated to the new specification.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 1-byte field that contains the storage class.

**,USERDATA=NO\_USERDATA****,USERDATA=userdata**

Use this input parameter to specify user-defined information to be written to the directory entry for the data item specified by NAME. The information is written only if either of the following is true:

- There is no entry data in the structure for NAME.
- There is unchanged entry data in the structure for NAME.

If one of these conditions is not met, USERDATA is ignored. If you do not want to write user data, but want to specify this parameter, code USERDATA=NO\_USERDATA.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-byte field that contains the user data.

**,VECTORINDEX=vectorindex**

Use this input parameter to specify an index into your local cache vector to be associated with the data item specified by NAME. The vector index identifies a vector entry that cache services will use to indicate both your interest in the data item and the validity of the copy of the data item in your local cache buffer.

- With WHENREG=NO, for structures allocated in a coupling facility with CFLEVEL=4 or lower, VECTORINDEX is required. If the vector index you specify is already associated with another data item, you must disassociate the vector index from the old name by specifying OLDNAME before the vector index can be associated with the data item specified by NAME.

For structures allocated in a coupling facility with CFLEVEL=5 or higher, VECTORINDEX is required when either REGUSER=YES or OLDNAME is specified with WHENREG=NO.

- With WHENREG=YES, which is valid only when the cache structure is allocated in a CFLEVEL=2 or higher coupling facility, the vector index you specify must be the same as the vector index with which you currently have a registered interest in the data item. When VECTORINDEX is specified and the vector index does not equal the vector index with which you currently have a registered interest in the data item, the request fails. The vector index with which you are currently registered is returned in the cache answer area.

**To Code:** Specify the RS-type name or address (by using a register from 2 to ) of a 4-byte field that contains the vecr index for NAME.

**,VERSCOMP=NO\_VERSCOMP****,VERSCOMP=verscomp**

Use this input parameter to specify a version number to be compared to the version number of the entry designated by NAME.

VERSCOMP is meaningful only for structures allocated in a coupling facility of CFLEVEL=5 or higher.

If the condition specified by VERSCOMPTYPE is not met, the request is terminated with no resultant change to the structure.

Note that use of VERSCOMP is needed to ensure that updates to the version number requested with the VERSUPDATE keyword are not processed multiple times as a result of internal request redrive logic. When VERSCOMP is requested along with VERSUPDATE to update the version number, then if the initial execution of the request succeeds, any subsequent internal redrive of the request will fail due to a version number miscompare, preventing multiple updates from occurring on the request. Conversely, if the initial execution of the request was unsuccessful, then any subsequent internal redrive of the request will be able to execute successfully and update the version number only once.

In either of these cases, if the request is internally redriven and experiences a version number miscompare on the redrive, a return and reason code of IXLSNCODESTATUSUNKNOWN will be returned. This reflects the fact that it is not known whether the observed version number miscompare resulted from the version number update succeeding on the original issuance of the request (causing the miscompare on the redriven request), or the observed version number miscompare was present all along, or resulted from a version number update made by another request. When this return and reason code is returned, it is up to the user to determine whether or not the requested update has actually occurred, and take the appropriate recovery action.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-byte field that contains the comparative version number.

**,VERSCOMPTYPE=EQUAL**

**,VERSCOMPTYPE=LESSOREQUAL**

Use this input parameter to specify how the structure entry version number comparison is to be performed.

VERSCOMPTYPE is meaningful only for structures allocated in a coupling facility of CFLEVEL=5 or higher.

**VERSCOMPTYPE=EQUAL**

The version number for the structure entry must be equal to the value specified for VERSCOMP.

**VERSCOMPTYPE=LESSOREQUAL**

The version number for the structure entry must be less than or equal to the value specified for VERSCOMP.

**,VERSUPDATE=NONE**

**,VERSUPDATE=INC**

**,VERSUPDATE=DEC**

**,VERSUPDATE=SET**

Use this input parameter to specify how the entry version number will be updated or, for those cases where an entry is created, initialized.

VERSUPDATE is meaningful only for structures allocated in a coupling facility of CFLEVEL=5 or higher.

Note that, depending on the VERSUPDATE specification, internal request redrive logic may cause a request to INCRement the version number more than once, DECrement the version number more than once, or SET the version number more than once. If it is necessary to ensure that this kind of multiple operation does not occur, version number comparison must also be requested. See the description of the VERSCOMP keyword.

**VERSUPDATE=NONE**

The version number is not updated.

On a request that causes an entry to be created, the version number is set to contain all binary zeros.

**VERSUPDATE=INC**

The version number will be incremented.

On a request that causes an entry to be created, the version number for the created entry is set to contain all binary zeros except for the low order bit, which is set to one.

**VERSUPDATE=DEC**

The version number will be decremented.

On a request that causes an entry to be created, the version number for the created number is set to contain binary ones.

**VERSUPDATE=SET**

The version number will be set to the value specified by NEWVERS, including the case where an entry is created.

**,WHENREG=YES**

**,WHENREG=NO**

Use this input parameter to specify whether previous registration of interest by the requesting user in the data item specified by NAME is a prerequisite to this request.

**YES**

The request is processed only if your interest in the data item is already registered. If your interest is not registered, the request will fail.

**NO**

The request is processed regardless of whether your interest in the data item is already registered.

- If the entry already exists in the structure, the data and your interest is updated.
- If the entry does not exist in the structure, a new entry is created, and your interest is registered.

See the VECTORINDEX parameter description for how to register interest in a data item.

## ABEND Codes

Abend X'026' (See *z/OS MVS System Codes* for more information on this abend.)

## Return and Reason Codes

When the system returns control to your program:

- GPR 15 (and RETCODE, if specified) contains a return code.
- If the return code is not zero, GPR 0 (and RSNCODE, if specified) contains a reason code.

**Note:** The return and reason codes will also be put into the answer area (mapped by IXLYCAA), if there is one, when the request completes.

The IXLYCON macro provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

<b>0</b>	IXLRETCODEOK
<b>4</b>	IXLRETCODEWARNING
<b>8</b>	IXLRETCODEPARMERROR
<b>C</b>	IXLRETCODEENVERROR
<b>10</b>	IXLRETCODECOMPERROR

The following table contains hexadecimal return and reason codes, the equate symbols associated with each reason code, and the meaning and suggested action for each return and reason code.

Table 46. Return and Reason Codes for the IXLCACHE REQUEST=WRITE_DATA Macro		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0	None.	<p><b>Meaning:</b> If you specified a MODE value of ASYNCECB, ASYNCEXIT, or ASYNCTOKEN, the request has been successfully initiated. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed. For any other MODE value, the request has successfully completed.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=ASYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=ASYNCEXIT, the connection's complete exit will be given control when the request has completed.</li> <li>• If you specified MODE=ASYNCTOKEN, use REQTOKEN when calling IXLFCOMP to determine when the request has completed.</li> </ul>



Table 46. Return and Reason Codes for the IXLCACHE REQUEST=WRITE\_DATA Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	xxxx0402	<p><b>Equate Symbol:</b> IXLRNCODEASYNCH</p> <p><b>Meaning:</b> The request could not be processed synchronously. It will be processed asynchronously. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed.</p> <ul style="list-style-type: none"> <li>• If you specified MODE=SYNCECB, the ECB specified in REQECB will be posted when the request has finished.</li> <li>• If you specified MODE=SYNCEXIT, the connection's complete exit will be given control when the request has finished.</li> <li>• If you specified MODE=SYNCTOKEN, the REQTOKEN contains an asynchronous request token that may be used on the IXLFCOMP macro to determine when the request has completed.</li> </ul> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=SYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=SYNCEXIT, the connection's completion exit will be given control when the request has completed.</li> <li>• If you specified MODE=SYNCTOKEN, use REQTOKEN when calling IXLFCOMP to determine when the request has completed.</li> </ul>
4	xxxx042C	<p><b>Equate Symbol:</b> IxIRsnCodeLocalRegWrtSuppress</p> <p><b>Meaning:</b> For a WRITE_DATA request with WHENREG=NO, ASSIGN=NO, LOCALREGCNTL=YES and CHANGED=YES specified, the user's connection was the only registered interest in the directory entry for NAME in the cache structure, and no subsystem data for the directory entry is cached. The write operation is suppressed. No data was written for the data item.</p> <p><b>Action:</b> When the subsystem data is not written to the cache structure, the local cache buffer contains the only valid copy of the subsystem data. Hardening of the subsystem data to permanent storage should be performed to ensure availability and successful recovery of data in the event of a failure.</p>

Table 46. Return and Reason Codes for the IXLCACHE REQUEST=WRITE\_DATA Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0801	<p><b>Equate Symbol:</b> IXLRSNCODEBADPARMLIST</p> <p><b>Meaning:</b> The parameter list (mfctrl) for this request is not addressable. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the parameter list address.</li> <li>• The parameter list must be addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are issuing this macro while disabled, the parameter list must reside in either page-fixed or DREF storage.</li> <li>• If you are issuing this macro in AR-mode and you specified the parameter list address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are issuing this macro in AR-mode, SYSSTATE ASCENV=AR must be issued before this macro.</li> </ul>
8	xxxx0804	<p><b>Equate Symbol:</b> IXLRSNCODEBADVERSION#</p> <p><b>Meaning:</b> The version number in the macro parameter list is not compatible with the level of XES currently being used. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that your program did not overlay the parameter list storage.</li> <li>• Verify that your program was assembled with the correct macro library for the release of MVS on which your program is running.</li> <li>• Verify that your program is running on an MVS system that supports the version of the macro you are using.</li> </ul>

Table 46. Return and Reason Codes for the IXLCACHE REQUEST=WRITE\_DATA Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx080A	<p><b>Equate Symbol:</b> IXLRNCODEBADCONTOKEN</p> <p><b>Meaning:</b> Program error. The specified connect token (CONTOKEN) was not valid for one of the following reasons:</p> <ol style="list-style-type: none"> <li>1. The user with the connection identifier represented by the token has disconnected from the structure.</li> <li>2. The connector's task (the task that issued IXLCONN) ended.</li> <li>3. The specified token is not the token that was returned from IXLCONN.</li> <li>4. The request was issued from an address space other than the address space in which IXLCONN was issued.</li> <li>5. The connect token was invalidated during rebuild.</li> <li>6. The connect token was invalidated by XES.</li> </ol> <p><b>Note:</b> The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Verify that the CONTOKEN value specified was valid and for the correct structure. The numbers on the following actions correspond to the numbers above.</p> <ol style="list-style-type: none"> <li>1. Discontinue use. Perform recovery and cleanup for the structure and check protocol for use of structure to determine why you did not know of the disconnection.</li> <li>2. Discontinue use. Perform recovery and cleanup for the structure and check protocol for use of structure to determine why you did not know that the task ended.</li> <li>3. Issue IXLCONN to connect to the structure. The contoken is returned in the CONACONTOKEN field of the answer area specified on the IXLCONN request.</li> <li>4. Issue your request from the same address space the IXLCONN was issued.</li> <li>5. Participate in the rebuild. When it is complete, try again.</li> <li>6. Discontinue use. Perform recovery and cleanup for the structure and check protocol for use of structure.</li> </ol> <p>You may want to issue IXCQUERY to get more information about the structure.</p>
8	xxxx0819	<p><b>Equate Symbol:</b> IXLRNCODEBADVECTOROP</p> <p><b>Meaning:</b> The VECTORINDEX specified is not valid. Request processing was suppressed.</p> <p><b>Action:</b> Specify a vector index that is within the current range of the number of vector entries for the local vector requested for this structure. The number of vector entries is determined by the VECTORLEN parameter specified on the IXLCONN macro and may be changed by issuing the IXLVECTR macro.</p>

Table 46. Return and Reason Codes for the IXLCACHE REQUEST=WRITE\_DATA Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0824	<p><b>Equate Symbol:</b> IXLRNCODEWRONGSTRTYPE</p> <p><b>Meaning:</b> The connect token specified by CONTOKEN is not to a cache structure.</p> <p><b>Action:</b> Verify the connect token for this cache structure.</p> <p><b>Note:</b> The connect token can be found in the IXLCONN answer area. Verify the value specified on the TYPE= parameter of the IXLCONN request for this structure.</p>
8	xxxx0825	<p><b>Equate Symbol:</b> IXLRNCODENOENTRY</p> <p><b>Meaning:</b> Program error.</p> <ul style="list-style-type: none"> <li>The request with WHENREG=YES specified failed for one of the following reasons: <ul style="list-style-type: none"> <li>The entry designated by name is not in the cache structure.</li> <li>The user's connection did not have registered interest in the entry.</li> <li>With CFLEVEL=2, if VECTORINDEX was specified, the vector index did not match the vector index with which the user currently has a registered interest in the data item. The currently-registered vector index is returned in the cache answer area (field CAALCVINUM, which is valid only if CAALCVI is set on).</li> </ul> </li> <li>The request with WHENREG=NO, ASSIGN=NO specified resulted in the suppression of the creation of a new data entry because the entry did not previously exist. (CFLEVEL=9 or higher)</li> </ul> <p><b>Action:</b> If this is expected, then your copy of the entry is invalid and you must get the latest copy. Re-read the data item into your local cache buffer and apply your changes again. Then try the write again.</p> <p>If this is unexpected, try the following:</p> <ul style="list-style-type: none"> <li>Verify that NAME is uncorrupted.</li> <li>Verify that the CONTOKEN is for the correct structure.</li> <li>Verify that the VECTORINDEX is the same as that for which you have previously registered interest in the entry.</li> </ul> <p>You might want to rerun the request with WHENREG=NO to allocate a directory entry for this NAME.</p>

Table 46. Return and Reason Codes for the IXLCACHE REQUEST=WRITE\_DATA Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0826	<p><b>Equate Symbol:</b> IXLRSNCODEINCOMPATSTATE</p> <p><b>Meaning:</b> Program error. The request failed because the state of the named data item is incompatible with the request.</p> <p>The following fields are returned in the answer area:</p> <ul style="list-style-type: none"> <li>• CAACHANGED</li> <li>• CAAINVLCVINUM (if WHENREG=NO was specified)</li> <li>• CAACLOCKSTATE</li> <li>• CAACLOCKVAL</li> </ul> <p><b>Action:</b> Check for the following conditions:</p> <ul style="list-style-type: none"> <li>• GETCOLOCK was specified for an entry that contains changed data (see CAACHANGED, CAACLOCKSTATE, and CAACLOCKVAL to determine if this is the case).</li> <li>• WHENREG=NO was specified and there is registered interest in the data item specified by NAME (see CAAINVLCVINUM).</li> </ul>
8	xxxx082D	<p><b>Equate Symbol:</b> IXLRSNCODEBADSTGCLASS</p> <p><b>Meaning:</b> Program error. The storage class specified by STGCLASS exceeds the maximum number of storage classes defined for the cache structure.</p> <p><b>Action:</b> The number of storage classes allowed for a structure are defined on the IXLCONN macro by the NUMSTGCLASS parameter. Correct the STGCLASS parameter to specify a valid storage class.</p>
8	xxxx082E	<p><b>Equate Symbol:</b> IXLRSNCODEBADCOCLASS</p> <p><b>Meaning:</b> Program error. The cast-out class specified by COCLASS exceeds the maximum number of cast-out classes defined for the cache structure.</p> <p><b>Action:</b> Correct the COCLASS parameter to specify a valid cast-out class. The valid range for COCLASS is from 1 to the maximum value specified by the NUMCOCLASS parameter on the IXLCONN macro.</p>
8	xxxx082F	<p><b>Equate Symbol:</b> IXLRSNCODEBADPARITY</p> <p><b>Meaning:</b> Program error. The parity value specified by PARITY is not valid.</p> <p><b>Action:</b> The value specified in the request contained parity bits that were not valid. The value for the parity bits may be 00, 01, or 11 in bits 2 and 3 of the PARITY specification (xxpp xxxx where pp is the parity bits). See the explanation for the PARITY keyword in this topic.</p>

Table 46. Return and Reason Codes for the IXLCACHE REQUEST=WRITE\_DATA Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0833	<p><b>Equate Symbol:</b> IXLRSNCODEBADPGBLATTR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list is specified as being pageable (PAGEABLE=YES) but is not.</p> <p><b>Action:</b> Change the buffer area(s) to pageable storage, or specify PAGEABLE=NO. See the PAGEABLE parameter description for specification instructions.</p>
8	xxxx0834	<p><b>Equate Symbol:</b> IXLRSNCODEBADNONPGBLATTR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list is specified as being nonpageable (PAGEABLE=NO) but is either pageable or not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The type of storage in the buffer area corresponds to the value you specified on the PAGEABLE parameter. See the PAGEABLE parameter description for specification instructions.</li> <li>• The correct buffer address was used.</li> <li>• The buffer area(s) were not previously freed.</li> <li>• If BUFLIST was specified and your program is running in AR mode, ensure that: <ul style="list-style-type: none"> <li>– The BUFALET specification is correct.</li> <li>– You specified SYSSTATE ASCENV=AR before issuing the macro.</li> </ul> </li> <li>• If the caller is running in AR-mode and the BUFFER or BUFLIST parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> </ul>

Table 46. Return and Reason Codes for the IXLCACHE REQUEST=WRITE\_DATA Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0835	<p><b>Equate Symbol:</b> IXLRNCODEBADDATAADDR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list, is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The correct buffer address was used for BUFFER or for a buffer within the BUFLIST.</li> <li>• The buffer area(s) were not previously freed.</li> <li>• The buffer area(s) were allocated in a storage key that matches the key specified using the BUFSTGKEY parameter, or, if BUFSTGKEY is omitted, the caller's PSW key.</li> <li>• If the caller is running in AR-mode and the BUFFER or BUFLIST parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If the caller is running in AR-mode, SYSSTATE ASCENV=AR must be specified before issuing this macro.</li> <li>• If BUFLIST was specified and your program is running in AR mode the BUFALET specification is correct.</li> </ul>
8	xxxx0836	<p><b>Equate Symbol:</b> IXLRNCODEBADREALADDR</p> <p><b>Meaning:</b> Program error. Real storage addresses were provided in a BUFLIST list, but one of the buffers is not addressable in central storage.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that BUFADDRTYPE was specified as you intended.</li> <li>• Ensure that the real buffer addresses specified by BUFLIST are valid.</li> </ul>
8	xxxx0837	<p><b>Equate Symbol:</b> IXLRNCODEBADWRITEADJDATA</p> <p><b>Meaning:</b> Program error. The request specified that adjunct data was to be written, but the area specified by ADJAREA is not addressable. There was no change to the structure.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the ADJAREA address.</li> <li>• ADJAREA must be addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are calling IXLCACHE while disabled, ADJAREA must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLCACHE in AR-mode and you specified the ADJAREA address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are calling IXLCACHE in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLCACHE macro.</li> </ul>

Table 46. Return and Reason Codes for the IXLCACHE REQUEST=WRITE\_DATA Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0838	<p><b>Equate Symbol:</b> IXLRNCODEBADANSAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by ANSAREA is not addressable.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the ANSAREA address.</li> <li>• ANSAREA must be addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are calling IXLCACHE while disabled, ANSAREA must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLCACHE in AR-mode and you specified the ANSAREA address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are calling IXLCACHE in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLCACHE macro.</li> </ul>
8	xxxx0839	<p><b>Equate Symbol:</b> IXLRNCODEBADREQTOKENAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by REQTOKEN is not addressable.</p> <p><b>Action:</b> Ensure that the request token area specified by REQTOKEN is valid:</p> <ul style="list-style-type: none"> <li>• Verify the REQTOKEN address.</li> <li>• If you are calling IXLCACHE while disabled, REQTOKEN must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLCACHE in AR-mode and you specified the REQTOKEN address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are calling IXLCACHE in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLCACHE macro.</li> </ul>
8	xxxx083D	<p><b>Equate Symbol:</b> IXLRNCODEBADANSLEN</p> <p><b>Meaning:</b> Program error. The length of the answer area, as specified by ANSLEN, is not sufficient to contain answer area information. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Increase the size of the answer area provided for the request, and rerun your program. Depending on the macro version number, specify the length of the answer area as follows:</p> <ul style="list-style-type: none"> <li>• When the connection specified ASYNCCI=1 on the IXLCONN invocation and AXIOVERRIDE=0 was specified or defaulted to for the IXLCACHE request, the answer area length must be a minimum value of CAALEVEL2LEN</li> <li>• When the value of the macro version number (PLISTVER) is 4 or more, the minimum answer area length is CAALEVEL1LEN.</li> <li>• When the value of the macro version number (PLISTVER) is 0-3, the minimum answer area length is CAALEVEL0LEN.</li> </ul>



Table 46. Return and Reason Codes for the IXLCACHE REQUEST=WRITE\_DATA Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx083F	<p><b>Equate Symbol:</b> IXLRNCODEBADENTRYVERSION</p> <p><b>Meaning:</b> Program error. The specified structure entry has a version number which does not meet the criteria specified in the request.</p> <p><b>Action:</b> None necessary.</p>
8	xxxx0851	<p><b>Equate Symbol:</b> IXLRNCODENOSUSPENDISABLE</p> <p><b>Meaning:</b> Program error. The request failed because MODE=SYNCSUSPEND was specified, but the caller is disabled and cannot be suspended.</p> <p><b>Action:</b> Either specify another MODE value or become enabled (release the CPU lock); then reissue the request.</p>
8	xxxx0865	<p><b>Equate Symbol:</b> IXLRNCODEBADBUFSPEC</p> <p><b>Meaning:</b> Program error. There is an error in the buffer specification.</p> <p><b>Action:</b> Check the following:</p> <ul style="list-style-type: none"> <li>• If BUFLIST was specified, check the requirements for BUFLIST, BUFNUM, and BUFINCRNUM.</li> <li>• If BUFFER was specified, check the requirements for BUFFER and BUFSIZE.</li> <li>• Buffer pointer(s) in BUFLIST.</li> <li>• Buffer boundaries.</li> </ul>
8	xxxx0866	<p><b>Equate Symbol:</b> IXLRNCODEBADBUFKEY</p> <p><b>Meaning:</b> Program error. The buffer storage key specified by BUFSTGKEY is incorrect or BUFSTGKEY was not specified and the caller's PSW key does not match the key of the buffers.</p> <p>The data cannot be fetched from the specified buffer area.</p> <p><b>Action:</b> Check the following:</p> <ul style="list-style-type: none"> <li>• Determine if the key of the storage being used for the buffers is different from the PSW key.</li> <li>• If BUFSTGKEY was specified, verify that the key was put in the correct bits (see the explanation of this parameter for more information).</li> </ul>

Table 46. Return and Reason Codes for the IXLCACHE REQUEST=WRITE\_DATA Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0867	<p><b>Equate Symbol:</b> IXLRNCOBEBADBUFLIST</p> <p><b>Meaning:</b> Program error. The 128-byte storage area specified by BUFLIST is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The correct BUFLIST address was used.</li> <li>• The BUFLIST area was not previously freed.</li> <li>• If the caller is running in AR-mode and the BUFLIST parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If the caller is disabled, then the BUFLIST must reside in either nonpageable or disabled reference storage.</li> <li>• If your program is running in AR mode, ensure that you specified SYSSTATE ASCENV=AR before issuing the macro.</li> </ul>
8	xxxx086A	<p><b>Equate Symbol:</b> IXLRNCOBEBADELEMNUM</p> <p><b>Meaning:</b> Program error. The value specified for ELEMNUM is not valid.</p> <p><b>Action:</b> Check the element information specified on the IXLCONN for this structure. The number of elements is affected by MAXELEMNUM on IXLCONN.</p>
8	xxxx08AD	<p><b>Equate Symbol:</b> IXLRNCOBEBADHIGHSHAREDVIRT</p> <p><b>Meaning:</b> Program error. The request specified a high shared virtual storage area (above 2GB).</p> <p><b>Action:</b> None required.</p>
C	xxxx0C06	<p><b>Equate Symbol:</b> IXLRNCOBENOCOCONN</p> <p><b>Meaning:</b> Environmental error. No connectivity to the cache structure exists. This may occur due to operator commands such as VARY PATH,OFFLINE or CONFIG CHP,OFFLINE or hardware errors such as facility or path failures. The CONTOKEN is invalidated. The request fails.</p> <p><b>Action:</b> Either disconnect from the structure (using IXLDISC) or initiate a rebuild of the structure, if allowed (using IXLREBLD).</p>

Table 46. Return and Reason Codes for the IXLCACHE REQUEST=WRITE\_DATA Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx08B9	<p><b>Equate Symbol:</b> IXLRSNCODENOANSAREA</p> <p><b>Meaning:</b> An answer area was not specified when one is required. The requested service determined that conditions exist that require an ANSAREA to complete the request.</p> <p><b>Action:</b> Provide an answer area (ANSAREA) and answer area length (ANSLEN) on the IXLCACHE macro invocation for the request. ANSAREA is required when the connection specified ASYNCXI=1 on the IXLCONN invocation when connecting to the cache structure, AXIOVERRIDE=0 was specified or defaulted to for the IXLCACHE request and the request is one of the following:</p> <ul style="list-style-type: none"> <li>• CROSS_INVALID</li> <li>• CROSS_INVALLIST</li> <li>• DELETE_NAME</li> <li>• DELETE_NAMELIST</li> <li>• READ_DATA</li> <li>• REG_NAMELIST</li> <li>• WRITE_DATA</li> <li>• WRITE_DATA LIST</li> </ul>
C	xxxx0C13	<p><b>Equate Symbol:</b> IXLRSNCODEREQPURGED</p> <p><b>Meaning:</b> Environmental error. The request was purged prior to completion of the request. Possible reasons include:</p> <ul style="list-style-type: none"> <li>• The connector failed.</li> <li>• The connector disconnected.</li> <li>• The requestor failed.</li> <li>• The request was purged by IXLPURGE.</li> <li>• Requests were purged when the connector provided an IXLEERSP response for the Rebuild Quiesce, Rebuild Stop, or Rebuild Cleanup event.</li> <li>• The connector invoked IXLREBLD REQUEST=COMPLETE.</li> <li>• The secondary address space was no longer valid.</li> </ul> <p><b>Action:</b> None if this is expected. Otherwise, determine why the connector failed.</p>
C	xxxx0C14	<p><b>Equate Symbol:</b> IXLRSNCODESTATUSUNKNOWN</p> <p><b>Meaning:</b> Environmental error. The IXLCACHE request has completed, but the final disposition of the request cannot be determined. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Verify the validity of your data by comparing the expected results with what is in the coupling facility.</p>

Table 46. Return and Reason Codes for the IXLCACHE REQUEST=WRITE\_DATA Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C17	<p><b>Equate Symbol:</b> IXLRNCODESTRFULL</p> <p><b>Meaning:</b> Environmental error. Allocation of a directory entry and/or data entry was necessary, but was unavailable or could not be reclaimed. The answer area field, CAASTGCLFULL, contains the storage class from which the reclaiming operation failed.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Determine if any data items may be cast-out to make room for this item.</li> <li>• Check your usage of storage classes to see if some data items can be moved to a different storage class (preferably with a lower priority) so some entries in the structure can be freed.</li> <li>• Determine if a rebuild of the structure is necessary to make room for more data entries/items.</li> </ul>
C	xxxx0C25	<p><b>Equate Symbol:</b> IXLRNCODESTRFAILURE</p> <p><b>Meaning:</b> Environmental error. The cache structure failed prior to completion of the request.</p> <p><b>Action:</b> Attempt to rebuild the structure using IXLREBLD, or disconnect from the structure using IXLDISC.</p>
C	xxxx0CA0	<p><b>Equate Symbol:</b> IXLRNCODEQUIESCEDSUSPENDFAIL</p> <p><b>Meaning:</b> Environmental error. The request is failed because the structure is quiesced for a system-managed process and SUSPEND=FAIL is specified on the IXLCONN.</p> <p><b>Action:</b> None, if this is expected.</p>
C	xxxxFFFF	<p><b>Equate Symbol:</b> IXLRNCODENOTAVAILABLE</p> <p><b>Meaning:</b> Environmental error. XES functions are not available. The coupling facility hardware might not be present.</p> <p><b>Action:</b> XES may not be used when XCF is running local mode. XES requires that the coupling facility be installed.</p>
10	xxxx10xx	<p><b>Equate Symbol:</b> IXLRNCODECOMPERROR</p> <p><b>Meaning:</b> System error. XES processing failure. The state of the structure and the disposition of the request are unpredictable. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Save the reason code information, and contact the IBM support center.</p>

## Chapter 49. IXLCACHE REQUEST=WRITE\_DATALIST

### Description

A WRITE\_DATALIST request allows you to write data to a cache structure for a given set of entries for a connection specified by CONTOKEN. Each entry to be written to the structure is identified by a write-operation-block (a WOB, mapped by IXLYWOB) contained in the BUFFER or BUFLIST storage area. When BUFFER is specified, 1 to 256 write-operation-blocks are allowed. When BUFLIST is specified, 1 to 16 write-operation-blocks are allowed.

The system references the write-operation-blocks by an index into BUFFER or BUFLIST and an offset specifying the location of the data area to be processed. The entries are processed sequentially beginning with the write-operation-block specified by the first index entry (STARTINDEX) and ending with the write-operation-block specified by the last index entry (ENDINDEX). The write-operation-block entries are numbered starting with 1. DATAOFFSET contains the offset in 256-byte increments into the storage area pointed to by BUFFER or BUFLIST. This is where the data resides that coincides with the write-operation-blocks.

Each write-operation-block contains the following information:

- Name of the entry
- Data area size in terms of the number of elements
- Storage class
- Additional information needed to write an entry to the cache structure

As each write-operation-block is processed, information about the request is returned in a write-operation-response-block (WORB). There is a one-to-one correspondence between each WOB in the BUFFER or BUFLIST and a WORB that the system returns in WORBAREA. The contents of each WORB, mapped by IXLYWORB, are as follows:

- The castout count for the castout class to which data was just written
- Total changed count for the storage class to which data was just written
- The invalidated local cache vector index, if any
- Additional information related to writing the data to the cache structure.

Additionally, the answer area (ANSAREA) contains the invalidated local cache validity vector, which is updated for each WOB processed.

When you issue a WRITE\_DATALIST request, based on the contents of the WOB, you can specify whether you do not want to register interest in the entry, whether you want to deregister any outstanding interest for a different entry, whether a directory entry is to be assigned, and, depending on the value of a change control indicator in the WOB, how registered interest and cross-invalidations are to be performed.

- The suppress registration indicator specifies whether the request should register connection interest in the entry.
- The assignment suppression control indicator specifies whether a directory entry should be assigned if one does not currently exist.
- OLDNAME specifies the name of the data item for which your interest should be deregistered. If both OLDNAME and NAME are specified in a WOB and their values are not equal, the name replacement control indicator specifies whether deregistration of interest for OLDNAME should be performed.
- The change control indicator specifies whether changed data is to be written to an entry in the cache structure.

See *z/OS MVS Programming: Sysplex Services Guide* for a complete list of WOB indicators and how their settings affect the processing of each entry.

Processing for a WRITE\_DATALIST request can be discontinued in the following instances:

- Version number comparison fails.
- Cast-out class specified is not valid.
- Parity bits specified are not valid.
- Inconsistency in the changed status of the data.
- Data area size specified is not valid.
- Storage class specified is not valid.
- Inability to obtain resources from the storage class specified.
- Inaccurate element number specified.
- Entry does not exist.

In these cases, all of the prior write-operation-blocks were processed and the index of the failing write-operation-block is returned in the answer area (CAAWDLINDEX). See [z/OS MVS Programming: Sysplex Services Guide](#) for a complete description of the instances listed above.

Processing of an entire WRITE\_DATALIST request can be suppressed in the following instances:

- The write-operation-block contains both the change control indicator and the get cast-out lock control indicator set.
- The write-operation-block contains a local cache vector index that is not valid.
- LOCALREGCNTL=YES is specified for the WRITE\_DATALIST request, but the assignment suppression control indicator and change control indicator are not set in each write-operation block.

In these cases, none of the specified write-operation-blocks was processed and the index of the failing write-operation-block is returned in the answer area (CAAWDLINDEX).

A WRITE\_DATALIST request can complete prematurely because the request exceeds the time-out criteria for the coupling facility (time-out criteria is model-dependent) or because the WORKAREA becomes full before processing is complete for the requested WOBS. When a request completes prematurely, the system returns an index value (CAAWDLINDEX) in the answer area that you can use to restart the WRITE\_DATALIST request. The index value when a WRITE\_DATALIST completes prematurely is the index of the first unprocessed write-operation-block. Also returned in the answer area is the offset in the data block of the data area for the next write-operation-block to be processed and the invalidated local cache validity vector. All prior write-operation-blocks were processed.

To continue processing the remaining write-operation-blocks, reissue the WRITE\_DATALIST request specifying the write-operation-block index and the offset of the next data area returned in the answer area as the STARTINDEX and the DATAOFFSET.

A WRITE\_DATALIST request can be issued only for cache structures allocated in a coupling facility of CFLEVEL=12 or higher. WRITE\_DATALIST requests issued for a cache structure allocated in a coupling facility of CFLEVEL=0 through 11 will fail.

For more information, see [z/OS MVS Programming: Sysplex Services Guide](#).

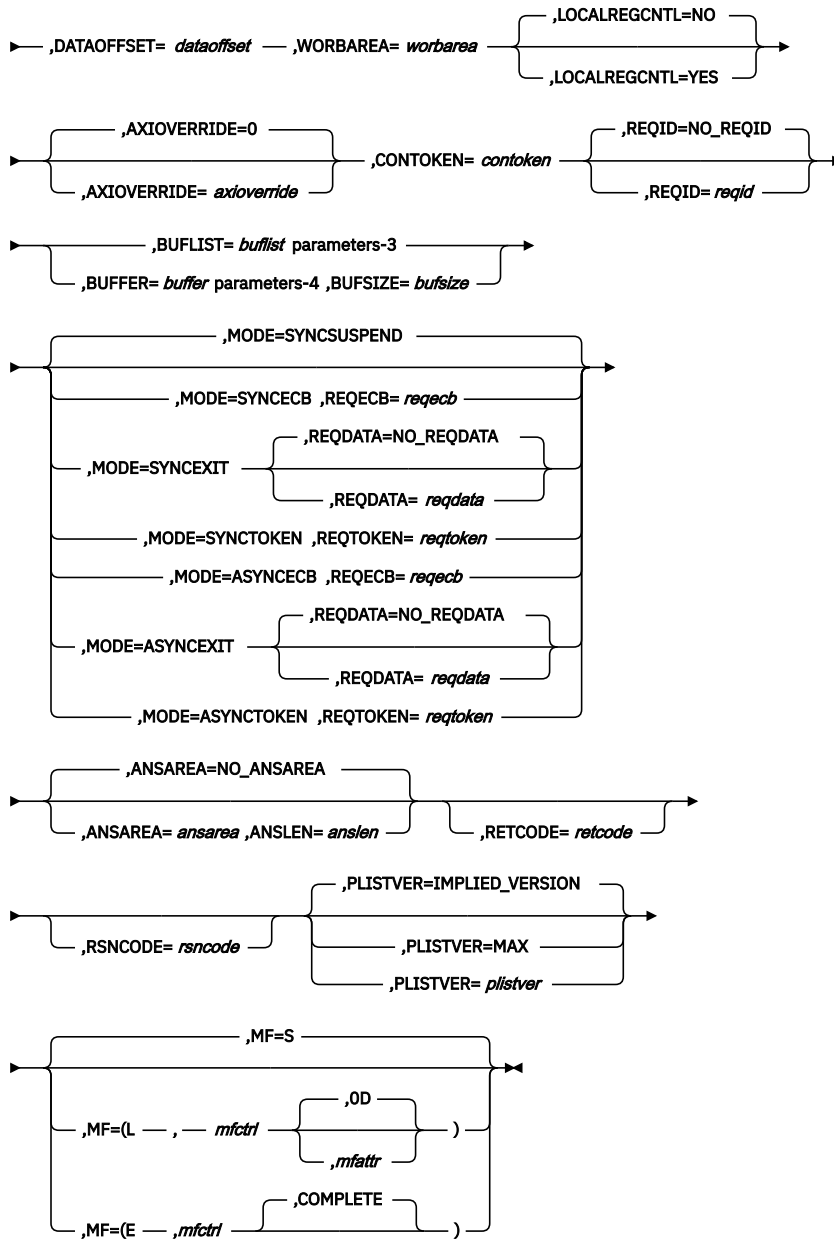
## Syntax Diagram

---

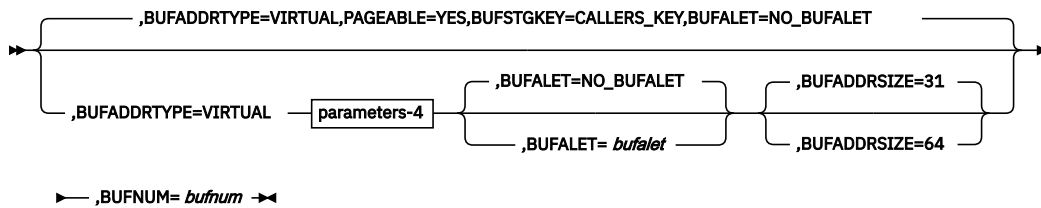
The syntax diagram for IXLCACHE REQUEST=WRITE\_DATALIST is as follows:

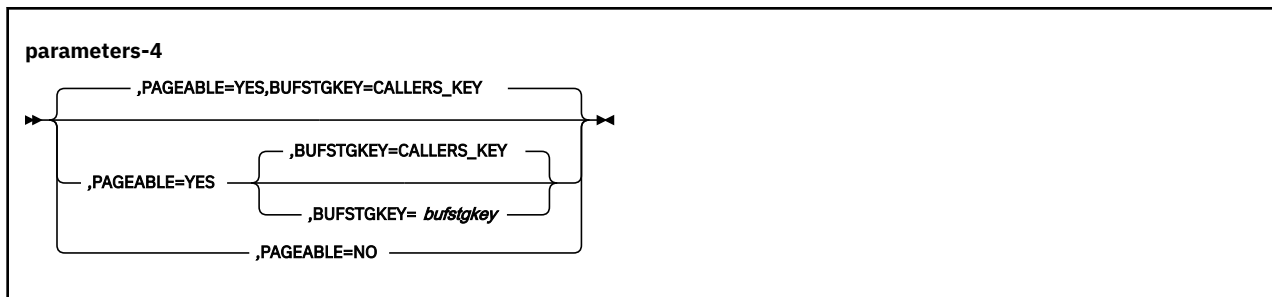
## main diagram

➤ IXLCACHE — REQUEST=WRITE\_DATALIST — ,STARTINDEX= *startindex* — ,ENDINDEX= *endindex* ➤



## parameters-3





**Note:** You must specify `ANSAREA=ansarea`, `ANSLEN=anslen` if you:

- Specify `MODE=SYNCTOKEN` or `MODE=ASYNCTOKEN`, or
- Specify `AXIOVERRIDE=0` (or default to 0) and specified `ASYNCXI=1` on the `IXLCONN` invocation when connecting to the cache structure.

## Parameter Descriptions

The parameter descriptions for `REQUEST=WRITE_DATALIST` are listed in alphabetical order. Default values are underlined:

### **REQUEST=WRITE\_DATALIST**

Use this input parameter to specify that data for a given set of entries for a connection specified by `CONTOKEN` is to be written to the cache structure. Each entry to be written to the cache structure is identified by a write-operation-block, mapped by `IXLYWOB`.

### **,ANSAREA=NO ANSAREA**

### **,ANSAREA=*ansarea***

Use this output parameter to specify an answer area to contain information returned from the request. The format of the answer area is described by the `IXLYCAA` mapping macro. On a successful completion of a request, the following information is returned in the answer area:

- If changed data is written to the cache, the total number of entries containing either changed or locked-for-cast-out data, that are assigned to the storage class to which the entry was written, is returned (field `CAATOTCHANGED`).
- If changed data is written to the cache, the total number of data items assigned to the cast-out class to which data was just written is returned (field `CAACOCOUNT`).

The following additional field is returned in the answer area when the connection specified `ASYNCXI=1` on the `IXLCONN` invocation when connecting to the cache structure and the cache structure is allocated in a `CFLEVEL=23` or higher coupling facility:

- An asynchronous cross-invalidation sequence number (`CAAASYNCXISEQNUM`) if cross-invalidations against local caches for this request were initiated asynchronously to the completion of the request.

See [Chapter 29, "IXLAXISN," on page 449](#) for a description of how to use the returned `CAAASYNCXISEQNUM` to determine when cross-invalidates of local caches associated with the request have completed.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an area (with a length of `ANSLEN`) where the information returned from the request may be stored.

### **,ANSLEN=*anslen***

Use this inppaameter to specify the size of the storage area specified by `ANSAREA`.

Use either `CAALEVEL0LEN`, `CAALEVEL1LEN` or `CAALEVEL2LEN` of the `IXLYCAA` mapping macro to determine the minimum size of the answer area. The answer area length must be at least large enough to accommodate the level of the `IXLYCAA` mapping appropriate to the requested function.

- When the connection specified `ASYNCXI=1` on the `IXLCONN` invocation and `AXIOVERRIDE=0` was specified or defaulted to for the `IXLCACHE` request, the answer area length is a required parameter and must be a minimum value of `CAALEVEL2LEN` to contain a returned asynchronous cross-invalidation sequence number (`CAAASYNCXISEQNUM`).



- When the value of PLISTVER is 4 or above, the minimum answer area length is CAALEVEL1LEN.
- When the value of PLISTVER is 0 - 3, the minimum answer area length is CAALEVEL0LEN.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12 2 to ) of a 2-byte field that contains the length of the answer area (ANSAREA).

**,AXIOVERRIDE=0**

**,AXIOVERRIDE=*axioverride***

Use this input parameter to specify whether the asynchronous cross-invalidation control setting of IxlConnAsyncXiYes (1) for the connection identified by CONTOKEN should be overridden for this request. Valid values are 0 (IxlCacheAXiOverrideNo) or 1 (IxlCacheAXiOverrideYes).

The AXIOVERRIDE keyword is meaningful to processing only when the connection specified ASYNCXI=1 on the IXLCONN invocation when connecting to the cache structure and the cache structure is allocated in a CFLEVEL=23 or higher coupling facility.

A value of 0 (IxlCacheAXiOverrideNo) indicates that the asynchronous cross-invalidation control for the connection as specified on the IXLCONN invocation should be used for the request. Cross-invalidates against local caches for this request will preferentially be initiated asynchronously to the completion of the request when asynchronous cross-invalidations are supported by the coupling facility where the cache structure is allocated.

A value of 1 (IxlCacheAXiOverrideYes) indicates that the ASYNCXI specification of IxlConnAsyncXiYes (1) by the connector on the IXLCONN invocation should be overridden for this request only. Cross-invalidations generated by this request will be processed synchronously to the completion of the request.

Any value other than 0 or 1 for AXIOVERRIDE will have the same behavior as specifying a value of 0 (IxlCacheAXiOverrideNo).

If cross-invalidates against local caches for this request were initiated asynchronously to the completion of the request, an asynchronous cross-invalidation sequence number is returned in CAAASYNCXISEQNUM of the cache answer area (ANSAREA).

The asynchronous cross-invalidation sequence number can be used on a subsequent invocation of IXLAXISN to ensure that the asynchronous cross-invalidations associated with this request have completed.

When cross-invalidations are initiated synchronously to the completion of the request or no cross-invalidations occurred for the request, no asynchronous cross-invalidation sequence number is returned.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12 2 to ) of a one-byte input field that contains the value indicating whether the asynchronous cross-invalidation control setting of the connector should be overridden

**,BUFADDRSIZE=31**

**,BUFADDRSIZE=64**

Use this input parameter to specify whether a 31-bit or a 64-bit address is specified by a BUFLIST entry.

**31**

The entry in BUFLIST is 31 bits in size.

**64**

The entry in BUFLIST is 64 bits in size.

**,BUFADDRTYPE=VIRTUAL**

**,BUFADDRTYPE=REAL**

Use this input parameter to specify whether the buffer addresses specified in the BUFLIST list are virtual storage or real storage addresses.

**VIRTUAL**

The buffer addresses are virtual storage addresses. The virtual storage can be pageable or nonpageable. See the PAGEABLE parameter for information about managing storage binds when specifying virtual storage addresses.

**REAL**

The buffer addresses are real storage addresses.

Note that for WRITE\_DATALIST requests, real storage addresses cannot be used.

**,BUFALET=NO\_BUFALET****,BUFALET=*bufalet***

Use this input parameter to specify an access list entry token (ALET) to be used in referencing all of the buffers specified by BUFLIST.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 4-byte field that contains the ALET.

**,BUFFER=*buffer***

Use this input parameter to specify a buffer area to hold the entry data to be written to the cache structure. Only 31-bit addressable virtual storage areas (below 2GB) are supported by the BUFFER specification. High virtual storage areas (above 2GB) can only be specified via the BUFLIST specification.

You can define the buffer size to be a total of up to 65536 bytes. Other requirements depend on the size you select:

- If you specify a buffer size of less than or equal to 4096 bytes, you must ensure that the buffer:
  - Is 256, 512, 1024, 2048, or 4096 bytes.
  - Starts on a 256-byte boundary.
  - Does not cross a 4096-byte boundary.
  - Does not start below storage address 512.
- If you specify a buffer size of greater than 4096 bytes, you must ensure that the buffer:
  - Is a multiple of 4096 bytes.
  - Is less than or equal to 65536 bytes.
  - Starts on a 4096-byte boundary.
  - Does not start below storage address 512.

See the BUFSIZE parameter description for defining the size of the buffer.

See [z/OS MVS Programming: Sysplex Services Guide](#) for more information on buffers.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an area (with a length of BUFSIZE) to contain the data to be written to the entry in the structure.

**,BUFINCRNUM=16**

For WRITE\_DATALIST requests that specify a BUFLIST parameter, the system assumes a BUFINCRNUM value of 16 that corresponds to a buffer size of 4096.

**,BUFLIST=*buflist***

Use this input parameter to specify a list of buffers to hold information about the entry data to be written to the cache structure.

Either 31-bit addressable (below 2GB) or 64-bit addressable (above 2GB) real or virtual storage areas are supported for the BUFLIST specification, depending on the specifications for the BUFADDRTYPE and BUFADDRSIZE keywords. However, pageable high shared virtual storage areas (above 2GB) may not be used.

BUFLIST specifies a 128-byte storage area that consists of a list of 1 to 16 elements.

The format of the list is a set of 8-byte elements. The BUFADDRSIZE keyword denotes whether four or eight bytes of the element are used.

- If BUFADDRSIZE=31 is specified, then the first four bytes of each element are reserved space and the last four bytes contain the address of the buffer.
- If BUFADDRSIZE=64 is specified, then the full eight bytes specify the address of the buffer.

For WRITE\_DATALIST requests, the first buffer pointed to by the first element in the BUFLIST can contain up to 16 WOBs; the remainder of the buffers in BUFLIST can contain data to be written to the entries specified by the WOBs.

See *z/OS MVS Programming: Sysplex Services Guide* for more information on buffers.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 128-byte storage area that contains a list of buffer addresses.

#### **,BUFNUM=*bufnum***

Use this input parameter to specify the number of buffers in the BUFLIST list. Valid BUFNUM values are from 0 to 16. A value of zero indicates that no data is to be written to the entry. If you do not want buffers, but want to specify this parameter, code BUFNUM=NO\_BUFNUM.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 1-byte field that contains the number of buffers (0 to 16) in the buffer list.

#### **,BUFSIZE=*bufsize***

Use this input parameter to specify the size of the BUFFER area. See the BUFFER parameter description for valid buffer sizes.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a fullword field that contains the size of the buffer (BUFFER) in bytes.

#### **,BUFSTGKEY=CALLERS\_KEY**

#### **,BUFSTGKEY=*bufstgkey***

Use this input parameter to specify a storage key that you define and use when referencing the buffers specified by BUFLIST or the buffer that is specified by BUFFER.

If you do not specify BUFSTGKEY, or if you specify BUFSTGKEY=CALLERS\_KEY, all references to one or more buffers are performed by using the caller's PSW key.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of an 8-bit field that contains the storage key in the format B'kkkkxxxx', where kkkk is the key and xxxx is ignored.

#### **,CONTO=*conken***

Use this input parameter to specify the connect token that was returned by the IXLCONN service in the IXLCONN answer area, which is mapped which is by IXLYCONA. The connect token uniquely identifies your connection to the cache structure, and must be specified on each IXLCACHE invocation.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field that contains the connect token.

#### **,DATAOFFSET=*dataoffset***

Use this input parameter to specify the offset in 256-byte increments into the data block in the storage area specified by BUFFER or BUFLIST of the first data area to be processed (the data block corresponding to the write-operation-block located by STARTINDEX).

A value of 0 implies that there is no actual data being passed in the storage area specified by BUFFER or BUFLIST. When this occurs, all WOBs have to specify an ELEMNUM of 0 to indicate that no data is being passed in to go with those WOBs. If any WOB specifies an ELEMNUM other than 0, the request will fail with return code IXLRETCODEPARMERROR, reason code IXLRSNCODEBADELEMNUM.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a halfword field that contains the offset into the data block.

#### **,ENDINDEX=*endindex***

Use this input parameter to specify the ending index for the last WOB to be processed in the storage area specified by BUFFER or BUFLIST. The index value must be greater than or equal to the value specified for STARTINDEX, but less than or equal to 256.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a halfword field containing the ending index value.

### **,LOCALREGCNTLNO**

### **,LOCALREGCNTL=YES**

Use this keyword to specify whether the write operation requests represented by the write-operation blocks should be suppressed (write suppression) when the user's connection (local cache) is the only registered interest in the data item for WOB\_NAME in the cache structure, and no subsystem data (data entry) for the data item is cached.

LOCALREGCNTL=YES is only meaningful when write operations are issued to cache structures allocated in a coupling facility that supports write suppression based on local cache registration. If the cache structure is allocated in a coupling facility that does not support write suppression based on local cache registration, the write operations will be performed as if LOCALREGCNTL=NO was specified.

### **NO**

The write operations will be performed normally without write suppression.

If a write operation is suppressed due to assignment suppression, the data is not written, the index of the write-operation block that failed, and the offset in the data block of the data area for the write-operation block being processed is returned in the ANSAREA. The WRITE\_DATALIST request fails with a return code IXLRETCODEPARMERROR, reason code IXLRSCODENOENTRY. All prior write-operation blocks were processed.

### **YES**

Suppress the write operations when the user's connection (local cache) is the only registered interest in a data item and the data entry for the data item does not contain cached subsystem data. When the write operation is suppressed, no data is written to the cache structure.

The CAAWRITESUPPRESSVECTOR in the ANSAREA indicates whether write-operation block write requests were suppressed during the WRITE\_DATALIST request. Each bit represents a corresponding write-operation block in the storage area specified by BUFFER or BUFLIST for the WRITE\_DATALIST request that specified LOCALREGCNTL=YES. The CAAWRITESUPPRESSEDVECTOR should be processed whenever the WRITE\_DATALIST request completes with a return code of IXLRETCODEOK or the return and reason code indicates that the WRITE\_DATALIST request completed prematurely and all the write operations prior to the premature completion were processed.

When the subsystem data is not written to the cache structure, the local cache buffer contains the only valid copy of the subsystem data. Hardening of the subsystem data to permanent storage should be performed to ensure availability and successful recovery of data in the event of a failure.

The assignment suppression control indicator and change control indicator must be set in each write-operation block when LOCALREGCNTL=YES is specified.

When issuing a WRITE\_DATALIST request using LOCALREGCNTL=YES to write data to a cache structure that is in user-managed duplex mode, the following programming technique should be followed:

- When using registration, assignment or write suppression protocols in the primary structure of a user-managed duplexed structure, unconditional writes of changed data to the secondary structure specifying LOCALREGCNTL=NO on the WRITE\_DATALIST request and WOB\_ASC = 'O'b in each write-operation block should be performed after successful write operations to the primary structure. If write operations to the primary structure resulted in the writes being suppressed, then those write operations should not be issued to the secondary structure.

The storage area specified by BUFFER or BUFLIST that contains the write-operation blocks should be updated to remove write-operation blocks and corresponding data areas for write operations to the primary structure that were suppressed due to assignment suppression or write suppression conditions. The CaaWriteSuppressVector in the ANSAREA indicates which, if any, write-operation block write requests were suppressed. Each bit represents a

corresponding write-operation block in the storage area specified by BUFFER or BUFLIST for the WRITE\_DATALIST request that specified LOCALREGCNTL=YES.

The WRITE\_DATALIST request should be issued to the secondary structure with the updated BUFFER or BUFLIST and LOCALREGCNTL=NO so that write suppression is ignored.

**,MF=S**  
**,MF=(L,mfctrl)**  
**,MF=(L,mfctrl,mfattr)**  
**,MF=(L,mfctrl,0D)**  
**,MF=(M,mfctrl)**  
**,MF=(M,mfctrl,COMPLETE)**  
**,MF=(M,mfctrl,NOCHECK)**  
**,MF=(E,mfctrl)**  
**,MF=(E,mfctrl,COMPLETE)**  
**,MF=(E,mfctrl,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

**,mfctrl**

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

**,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

**,COMPLETE**

**,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

**COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if SMILE=*var* were an optional parameter and the default is SMILE=NO\_SMILE then it would not be documented. However, if the default was SMILE=:), then it would be documented because a value would be the default.

**NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

**,MODE=SYNCSUSPEND**  
**,MODE=SYNCECB**  
**,MODE=SYNCEXIT**  
**,MODE=SYNCTOKEN**  
**,MODE=ASYNCECB**  
**,MODE=ASYNCEXIT**  
**,MODE=ASYNCTOKEN**

Use this input parameter to specify:

- Whether the request is to be performed synchronously or asynchronously
- How you want to be notified of request completion if the request is processed asynchronously.

See *z/OS MVS Programming: Sysplex Services Guide* for more information on understanding synchronous and asynchronous cache operations.

#### **SYNCSUSPEND**

The request processes synchronously. If necessary, the request is suspended until it can complete synchronously. To use this option, your program must be enabled for I/O and external interrupts.

#### **SYNCECB**

The request processes synchronously if possible. If the request processes asynchronously, the ECB specified by REQECB is posted when the request completes.

#### **SYNCEXIT**

The request processes synchronously if possible. If the request processes asynchronously, your complete exit is given control when the request completes.

#### **SYNCTOKEN**

The request processes synchronously if possible. If the request processes asynchronously, an asynchronous request token is returned to the area specified by REQTOKEN. Use the returned request token on the IXLFCOMP macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=SYNCTOKEN is specified.

#### **ASYNCECB**

The request processes asynchronously. The ECB specified by REQECB is posted when the request completes.

#### **ASYNCEXIT**

The request processes asynchronously. Your complete exit is given control when the request completes.

#### **ASYNCTOKEN**

The request processes asynchronously. An asynchronous request token is returned to the area specified by REQTOKEN. Use the returned request token on the IXLFCOMP macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=ASYNCTOKEN is specified.

**,PAGEABLE=YES**

**,PAGEABLE=NO**

Use this input parameter to identify whether the storage areas specified by BUFFER or BUFLIST are in pageable or potentially pageable storage.

#### **YES**

Specify this option to indicate that the BUFFER or BUFLIST buffers reside in pageable virtual storage. XES performs the required page fixing to fix the buffers in real storage while the cache or list request transfers data to or from the coupling facility.

This includes storage obtained from pageable subpools, disabled reference (DREF) storage, and may include storage that has the potential to become pageable during the processing of a request. (An example is address space storage owned by any swappable address space, for which a PGSER FIX has been successfully processed, but for which the owning address space gets swapped during processing of a cache or list request.) This does not include implicitly non-pageable storage (for example, storage obtained from non-pageable subpools).

The system takes responsibility for managing binds to central storage for the duration of the cache or list request, regardless of what address space owns the storage or whether the storage-owning address space is swappable or nonswappable. The storage can be owned by any address space.

High shared virtual storage areas (above 2GB) may not be used.

## NO

Specify this option to indicate that the BUFFER or BUFLIST buffers reside in non-pageable virtual storage. XES does not page fix the buffers in real storage.

This includes implicitly non-pageable storage areas (for example, storage obtained from non-pageable subpools), and may include storage that has the potential to become pageable during the processing of a request (An example is address space storage owned by any swappable address space, for which a PGSER FIX has been successfully processed, but for which the owning address space gets swapped-out during processing of a cache or list request.)

The system takes responsibility for managing binds to central storage for the duration of the cache or list request, if and only if the non-pageable storage is owned by either the requester's address space or the connector's address space. If the storage is owned by any other address space, then the invoker is responsible for ensuring that the virtual storage remains non-pageable for the duration of the request (including the case in which the storage is owned by a swappable address space that is swapped during processing of an IXLCACHE or IXLLIST request). Subject to this consideration, the storage can be owned by any address space. See *z/OS MVS Programming: Sysplex Services Guide*.

**,PLISTVER=IMPLIED\_VERSION**

**,PLISTVER=MAX**

**,PLISTVER=plistver**

Use this input parameter to specify the version of the macro. See “[Understanding IXLCACHE Version Support](#)” on page 461 for a description of the options available with PLISTVER.

**,REQDATA=NO\_REQDATA**

**,REQDATA=reqdata**

Use this input parameter with MODE=SYNCEXIT or MODE=ASYNCEXIT to pass any data you choose to the complete exit. The exit will get control only if the request is processed asynchronously.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the data to be passed to the complete exit.

**,REQECB=reqecb**

Use this output parameter with either MODE=SYNCECB or MODE=ASYNCECB to specify the address of an ECB, which is to be posted when the request completes if the request was processed asynchronously.

Before coding REQECB, you must ensure that:

- You initialize the ECB before you issue the request.
- The ECB resides in either common storage or the home address space where IXLCONN was issued.
- Any tasks that wait for the ECB to be posted reside in the home address space where IXLCONN was issued.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that contains the address of the ECB to be posted when the request completes. The ECB must be aligned on a fullword boundary.

**,REQID=NO\_REQID**

**,REQID=reqid**

Use this input parameter to specify a user-defined request identifier to be associated with the request. You can specify this request identifier on the IXPURGE macro to cancel a request that has not yet been processed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the user-defined request identifier.

**,REQTOKEN=reqtoken**

Use this output parameter with either MODE=SYNCTOKEN or MODE=ASYNCTOKEN to specify the address of a storage area to receive the request token that is returned when the request will be processed asynchronously. This token, which uniquely identifies the request, must be used as input to the IXLFCOMP macro, which you use to determine if the request has completed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field where the system will put the request token.

**,RETCODE=retcode**

Use this output parameter to specify a field to contain the return code. (The return code is also returned in register 15.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the return code when the request has completed.

**,RSNCODE=rsncode**

Use this output parameter to specify a field to contain the reason code returned, if applicable. (The reason code is also returned in register 0.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the reason code (if any) when the request has completed.

**,STARTINDEX=startindex**

Use this input parameter to specify the index for the first WOB in the storage area specified by BUFFER or BUFLIST to be processed. Valid STARTINDEX values are from 1 to the value of ENDINDEX. The first WOB in the storage area pointed to by the BUFFER or BUFLIST parameter has index number 1.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a halfword field containing the starting index value.

**,WORBAREA=workarea**

Use this output parameter to contain the write-operation response block array.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 192-byte area to contain the write-operation response block data.

## ABEND Codes

---

Abend X'026' (See [z/OS MVS System Codes](#) for more information on this abend.)

## Return and Reason Codes

---

When the system returns control to your program:

- GPR 15 (and RETCODE, if specified) contains a return code.
- If the return code is not zero, GPR 0 (and RSNCODE, if specified) contains a reason code.

**Note:** The return and reason codes will also be put into the answer area (mapped by IXLYCAA), if there is one, when the request completes.

The IXLYCON macro provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

**0**

IXLRETCODEOK

**4**

IXLRETCODEWARNING

**8**

IXLRETCODEPARMERROR



**C**

IXLRETCODEENVERROR

**10**

IXLRETCODECOMPERROR

The following table contains hexadecimal return and reason codes, the equate symbols associated with each reason code, and the meaning and suggested action for each return and reason code.

Table 47. Return and Reason Codes for the IXLCACHE REQUEST=WRITE_DATALIST Macro		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0	None.	<p><b>Meaning:</b> If you specified a MODE value of ASYNCECB, ASYNCEXIT, or ASYNCTOKEN, the request has been successfully initiated. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed. For any other MODE value, the request has successfully completed.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=ASYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=ASYNCEXIT, the connection's complete exit will be given control when the request has completed.</li> <li>• If you specified MODE=ASYNCTOKEN, use REQTOKEN when calling IXLFComp to determine when the request has completed.</li> </ul>
4	xxxx0402	<p><b>Equate Symbol:</b> IXLRSCODEASYNCH</p> <p><b>Meaning:</b> The request could not be processed synchronously. It will be processed asynchronously. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed.</p> <ul style="list-style-type: none"> <li>• If you specified MODE=SYNCECB, the ECB specified in REQECB will be posted when the request has finished.</li> <li>• If you specified MODE=SYNCEXIT, the connection's complete exit will be given control when the request has finished.</li> <li>• If you specified MODE=SYNCTOKEN, the REQTOKEN contains an asynchronous request token that may be used on the IXLFComp macro to determine when the request has completed.</li> </ul> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=SYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=SYNCEXIT, the connection's completion exit will be given control when the request has completed.</li> <li>• If you specified MODE=SYNCTOKEN, use REQTOKEN when calling IXLFComp to determine when the request has completed.</li> </ul>

Table 47. Return and Reason Codes for the IXLCACHE REQUEST=WRITE\_DATALIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	xxxx0409	<p><b>Equate Symbol:</b> IXLRSNCODETIMEOUT</p> <p><b>Meaning:</b> The request has completed prematurely because the model-dependent time-out criteria of the coupling facility has been exceeded. The index of the first unprocessed write-operation-block and the offset of the current data area have been returned in the answer area (fields CAAWDLINDEX and CAAWDLDATAOFFSET). All prior write-operation-blocks have been processed.</p> <p><b>Action:</b> To restart the request, update STARTINDEX with the value of CAAWDLINDEX and DATAOFFSET with the value of CAAWDLDATAOFFSET. For more information about premature request completion, see <a href="#">z/OS MVS Programming: Sysplex Services Guide</a>.</p>
8	xxxx0801	<p><b>Equate Symbol:</b> IXLRSNCODEBADPARMLIST</p> <p><b>Meaning:</b> The parameter list (mfctrl) for this request is not addressable. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the parameter list address.</li> <li>• The parameter list must be addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are issuing this macro while disabled, the parameter list must reside in either page-fixed or DREF storage.</li> <li>• If you are issuing this macro in AR-mode and you specified the parameter list address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are issuing this macro in AR-mode, SYSSTATE ASCENV=AR must be issued before this macro.</li> </ul>
8	xxxx0804	<p><b>Equate Symbol:</b> IXLRSNCODEBADVERSION#</p> <p><b>Meaning:</b> The version number in the macro parameter list is not compatible with the level of XES currently being used. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that your program did not overlay the parameter list storage.</li> <li>• Verify that your program was assembled with the correct macro library for the release of MVS on which your program is running.</li> <li>• Verify that your program is running on an MVS system that supports the version of the macro you are using.</li> </ul>

Table 47. Return and Reason Codes for the IXLCACHE REQUEST=WRITE\_DATALIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx080A	<p><b>Equate Symbol:</b> IXLRNCODEBADCONTOKEN</p> <p><b>Meaning:</b> Program error. The specified connect token (CONTOKEN) was not valid for one of the following reasons:</p> <ol style="list-style-type: none"> <li>1. The user with the connection identifier represented by the token has disconnected from the structure.</li> <li>2. The connector's task (the task that issued IXLCONN) ended.</li> <li>3. The specified token is not the token that was returned from IXLCONN.</li> <li>4. The request was issued from an address space other than the address space in which IXLCONN was issued.</li> <li>5. The connect token was invalidated during rebuild.</li> <li>6. The connect token was invalidated by XES.</li> </ol> <p><b>Note:</b> The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Verify that the CONTOKEN value specified was valid and for the correct structure. The numbers on the following actions correspond to the numbers above.</p> <ol style="list-style-type: none"> <li>1. Discontinue use. Perform recovery and cleanup for the structure and check protocol for use of structure to determine why you did not know of the disconnection.</li> <li>2. Discontinue use. Perform recovery and cleanup for the structure and check protocol for use of structure to determine why you did not know that the task ended.</li> <li>3. Issue IXLCONN to connect to the structure. The contoken is returned in the CONACONTOKEN field of the answer area specified on the IXLCONN request.</li> <li>4. Issue your request from the same address space the IXLCONN was issued.</li> <li>5. Participate in the rebuild. When it is complete, try again.</li> <li>6. Discontinue use. Perform recovery and cleanup for the structure and check protocol for use of structure.</li> </ol> <p>You may want to issue IXCQUERY to get more information about the structure.</p>
8	xxxx0819	<p><b>Equate Symbol:</b> IXLRNCODEBADVECTOROP</p> <p><b>Meaning:</b> The vector index in the write-operation-block indexed by CAAWDLINDEX is not valid. None of the write-operation-blocks have been processed.</p> <p><b>Action:</b> Correct the vector index value and reissue the WRITE_DATALIST request with the same STARTINDEX and ENDINDEX values.</p>

Table 47. Return and Reason Codes for the IXLCACHE REQUEST=WRITE\_DATALIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0824	<p><b>Equate Symbol:</b> IXLRNCODEWRONGSTRTYPE</p> <p><b>Meaning:</b> The connect token specified by CONTOKEN is not to a cache structure.</p> <p><b>Action:</b> Verify the connect token for this cache structure.</p> <p><b>Note:</b> The connect token can be found in the IXLCONN answer area. Verify the value specified on the TYPE= parameter of the IXLCONN request for this structure.</p>
8	xxxx0825	<p><b>Equate Symbol:</b> IXLRNCODENOENTRY</p> <p><b>Meaning:</b> Program error. The request failed because the assignment suppression control indication specified in the write-operation-block was set and the entry did not exist. The data is not written. The index of the failing write-operation-block and the offset in the data block of the data area for the write-operation-block being processed are returned in the answer area (fields CAAWDLINDEX and CAAWDLDATAOFFSET). All prior write-operation-blocks were processed.</p> <p><b>Action:</b> Reissue the request specifying for STARTINDEX the index of the next valid name element to be processed and DATAOFFSET with the offset of the next data area to be processed.</p>

Table 47. Return and Reason Codes for the IXLCACHE REQUEST=WRITE\_DATALIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0826	<p><b>Equate Symbol:</b> IXLRSNCODEINCOMPATSTATE</p> <p><b>Meaning:</b> Program error. The request failed because the state of the named data item is incompatible with the request. This may be due to one of the following conditions:</p> <ul style="list-style-type: none"> <li>• The change control indicator was not set in the write-operation-block, but the data in the cache structure was marked as changed.</li> <li>• The get cast-out lock control was set in the write-operation-block, but the cast-out lock was already held from a previous request.</li> <li>• The get cast-out lock control was set, but the entry was marked as changed or is already locked for castout by a CASTOUT_DATA or CASTOUT_DATALIST request.</li> <li>• The get cast-out lock control was set, but the entry is already locked for castout by another connector.</li> <li>• The change control was not set, but the entry is marked as changed.</li> <li>• The change control was not set, but the entry is locked for castout by a CASTOUT_DATA or CASTOUT_DATALIST request.</li> </ul> <p>The data is not written. The index of the failing write-operation-block (CAAWDLINDEX), the offset in the data block of the data area for the write-operation-block being processed (CAAWDLDATAOFFSET), the changed indication (CAACHANGED), the cast-out lock value (CAACOLCKVAL), the cast-out lock state (CAACOLCKSTATE), and the value of the local cache entry number (CAALCVINUM) are returned in the answer area. All prior write-operation-blocks were processed.</p> <p><b>Action:</b> Check the accuracy of the following values: CAAWDLINDEX, CAAWDLDATAOFFSET, CAACHANGED, CAACOLCKVAL, CAACOLCKSTATE, and CAALCVINUM. Correct the values as necessary and resubmit the request.</p>
8	xxxx082D	<p><b>Equate Symbol:</b> IXLRSNCODEBADSTGCLASS</p> <p><b>Meaning:</b> Program error. The storage class specified in the write-operation-block exceeds the maximum defined storage class for the structure. The data is not written. The index of the write-operation-block that failed and the offset in the data block of the data area for the write-operation-block being processed are returned in the answer area (fields CAAWDLINDEX and CAAWDLDATAOFFSET). All prior write-operation-blocks were processed.</p> <p><b>Action:</b> Correct the storage class value in the write-operation-block indexed by CAAWDLINDEX and reissue the WRITE_DATALIST request starting with that write-operation-block.</p>

Table 47. Return and Reason Codes for the IXLCACHE REQUEST=WRITE\_DATALIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx082E	<p><b>Equate Symbol:</b> IXLRSNCODEBADCOCLASS</p> <p><b>Meaning:</b> Program error. The cast-out class specified in the write-operation-block exceeds the maximum number of cast-out classes defined for the cache structure. The data is not written. The index of the write-operation-block that failed and the offset in the data block of the data area for the write-operation-block being processed are returned in the answer area (fields CAAWDLINDEX and CAAWDLDATAOFFSET). All prior write-operation-blocks were processed.</p> <p><b>Action:</b> Correct the COCLASS value in the write-operation-block to specify a valid cast-out class. The valid range for COCLASS is from 1 to the maximum value specified by the NUMCOCLASS parameter on the IXLCONN macro.</p>
8	xxxx082F	<p><b>Equate Symbol:</b> IXLRSNCODEBADPARITY</p> <p><b>Meaning:</b> Program error. The parity value specified in the write-operation-block was not valid. The data is not written. The index of the write-operation-block that failed and the offset in the data block of the data area for the write-operation-block being processed are returned in the answer area (fields CAAWDLINDEX and CAAWDLDATAOFFSET). All prior write-operation-blocks were processed.</p> <p><b>Action:</b> The value specified in the request contained parity bits that were not valid. The value for the parity bits may be 00, 01, or 11 in bits 2 and 3 of the PARITY specification (xxpp xxxx where pp is the parity bits).</p>
8	xxxx0833	<p><b>Equate Symbol:</b> IXLRSNCODEBADPGBLATTR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list is specified as being pageable (PAGEABLE=YES) but is not.</p> <p><b>Action:</b> Change the buffer area(s) to pageable storage, or specify PAGEABLE=NO. See the PAGEABLE parameter description for specification instructions.</p>

Table 47. Return and Reason Codes for the IXLCACHE REQUEST=WRITE\_DATALIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0834	<p><b>Equate Symbol:</b> IXLRNCODEBADNONPGBLATTR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list is specified as being nonpageable (PAGEABLE=NO) but is either pageable or not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The type of storage in the buffer area corresponds to the value you specified on the PAGEABLE parameter. See the PAGEABLE parameter description for specification instructions.</li> <li>• The correct buffer address was used.</li> <li>• The buffer area(s) were not previously freed.</li> <li>• If BUFLIST was specified and your program is running in AR mode, ensure that: <ul style="list-style-type: none"> <li>– The BUFALET specification is correct.</li> <li>– You specified SYSSTATE ASCENV=AR before issuing the macro.</li> </ul> </li> <li>• If the caller is running in AR-mode and the BUFFER or BUFLIST parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> </ul>
8	xxxx0835	<p><b>Equate Symbol:</b> IXLRNCODEBADDATAADDR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list, is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The correct buffer address was used for BUFFER or for a buffer within the BUFLIST.</li> <li>• The buffer area(s) were not previously freed.</li> <li>• The buffer area(s) were allocated in a storage key that matches the key specified using the BUFSTGKEY parameter, or, if BUFSTGKEY is omitted, the caller's PSW key.</li> <li>• If the caller is running in AR-mode and the BUFFER or BUFLIST parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If the caller is running in AR-mode, SYSSTATE ASCENV=AR must be specified before issuing this macro.</li> <li>• If BUFLIST was specified and your program is running in AR mode the BUFALET specification is correct.</li> </ul>

Table 47. Return and Reason Codes for the IXLCACHE REQUEST=WRITE\_DATALIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0838	<p><b>Equate Symbol:</b> IXLRNCODEBADANSAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by ANSAREA is not addressable.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the ANSAREA address.</li> <li>• ANSAREA must be addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are calling IXLCACHE while disabled, ANSAREA must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLCACHE in AR-mode and you specified the ANSAREA address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are calling IXLCACHE in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLCACHE macro.</li> </ul>
8	xxxx0839	<p><b>Equate Symbol:</b> IXLRNCODEBADREQTOKENAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by REQTOKEN is not addressable.</p> <p><b>Action:</b> Ensure that the request token area specified by REQTOKEN is valid:</p> <ul style="list-style-type: none"> <li>• Verify the REQTOKEN address.</li> <li>• If you are calling IXLCACHE while disabled, REQTOKEN must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLCACHE in AR-mode and you specified the REQTOKEN address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are calling IXLCACHE in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLCACHE macro.</li> </ul>
8	xxxx083D	<p><b>Equate Symbol:</b> IXLRNCODEBADANSLEN</p> <p><b>Meaning:</b> Program error. The length of the answer area, as specified by ANSLEN, is not sufficient to contain answer area information. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Increase the size of the answer area provided for the request, and rerun your program. Depending on the macro version number, specify the length of the answer area as follows:</p> <ul style="list-style-type: none"> <li>• When the connection specified ASYNCXI=1 on the IXLCONN invocation and AXIOVERRIDE=0 was specified or defaulted to for the IXLCACHE request, the answer area length must be a minimum value of CAALEVEL2LEN</li> <li>• When the value of the macro version number (PLISTVER) is 4 or more, the minimum answer area length is CAALEVEL1LEN.</li> <li>• When the value of the macro version number (PLISTVER) is 0-3, the minimum answer area length is CAALEVEL0LEN.</li> </ul>



Table 47. Return and Reason Codes for the IXLCACHE REQUEST=WRITE\_DATALIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx083F	<p><b>Equate Symbol:</b> IXLRNCODEBADENTRYVERSION</p> <p><b>Meaning:</b> Program error. The version number specified in the write-operation-block does not meet the version number criteria specified in the write-operation-block. The data is not written. The index of the write-operation-block that failed, the offset in the data block of the data area for the write-operation-block being processed, and the structure entry version number are returned in the answer area (fields CAAWDLINDEX, CAAWDLDATAOFFSET, and CAAVERSION). All prior write-operation-blocks were processed.</p> <p><b>Action:</b></p>
8	xxxx0851	<p><b>Equate Symbol:</b> IXLRNCODENOSUSPENDISABLE</p> <p><b>Meaning:</b> Program error. The request failed because MODE=SYNCSUSPEND was specified, but the caller is disabled and cannot be suspended.</p> <p><b>Action:</b> Either specify another MODE value or become enabled (release the CPU lock); then reissue the request.</p>
8	xxxx0865	<p><b>Equate Symbol:</b> IXLRNCODEBADBUFSPEC</p> <p><b>Meaning:</b> Program error. There is an error in the buffer specification.</p> <p><b>Action:</b> Check the following:</p> <ul style="list-style-type: none"> <li>• If BUFLIST was specified, check the requirements for BUFLIST, BUFNUM, and BUFINCRNUM.</li> <li>• If BUFFER was specified, check the requirements for BUFFER and BUFSIZE.</li> <li>• Buffer pointer(s) in BUFLIST.</li> <li>• Buffer boundaries.</li> </ul>
8	xxxx0866	<p><b>Equate Symbol:</b> IXLRNCODEBADBUFKEY</p> <p><b>Meaning:</b> Program error. The buffer storage key specified by BUFSTGKEY is incorrect or BUFSTGKEY was not specified and the caller's PSW key does not match the key of the buffers.</p> <p>The data cannot be fetched from the specified buffer area.</p> <p><b>Action:</b> Check the following:</p> <ul style="list-style-type: none"> <li>• Determine if the key of the storage being used for the buffers is different from the PSW key.</li> <li>• If BUFSTGKEY was specified, verify that the key was put in the correct bits (see the explanation of this parameter for more information).</li> </ul>

Table 47. Return and Reason Codes for the IXLCACHE REQUEST=WRITE\_DATALIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0867	<p><b>Equate Symbol:</b> IXLRNCOBEBADBUFLIST</p> <p><b>Meaning:</b> Program error. The 128-byte storage area specified by BUFLIST is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The correct BUFLIST address was used.</li> <li>• The BUFLIST area was not previously freed.</li> <li>• If the caller is running in AR-mode and the BUFLIST parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If the caller is disabled, then the BUFLIST must reside in either nonpageable or disabled reference storage.</li> <li>• If your program is running in AR mode, ensure that you specified SYSSTATE ASCENV=AR before issuing the macro.</li> </ul>
8	xxxx086A	<p><b>Equate Symbol:</b> IXLRNCOBEBADELEMNUM</p> <p><b>Meaning:</b> Program error. The ELEMNUM specified in the write-operation-block is not valid. The data is not written. The index of the write-operation-block that failed and the offset in the data block of the data area for the write-operation-block being processed are returned in the answer area (fields CAAWDLINDEX and CAAWDLDATAOFFSET). All prior write-operation-blocks were processed.</p> <p><b>Action:</b> Check the element information specified on the IXLCONN for this structure. The number of elements is affected by MAXELEMNUM on IXLCONN.</p>
8	xxxx0874	<p><b>Equate Symbol:</b> IXLRNCOBEBADWDLINDEX</p> <p><b>Meaning:</b> Program error. The value specified for either STARTINDEX or ENDINDEX is not valid. No entries are processed.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The values specified by STARTINDEX and ENDINDEX are in the range of 1 to 256.</li> <li>• The value specified by ENDINDEX is greater than or equal to the value of STARTINDEX.</li> <li>• The value specified by ENDINDEX does not imply a larger BUFFER size than was actually specified on the WRITE_DATALIST request.</li> </ul>

Table 47. Return and Reason Codes for the IXLCACHE REQUEST=WRITE\_DATALIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx087D	<p><b>Equate Symbol:</b> IXLRNCODEBADWORBAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by WORBAREA is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The address passed in WORBAREA is valid.</li> <li>• The WORBAREA area was not previously freed.</li> <li>• The WORBAREA area is addressable from the caller's primary address space or from the caller's PASM access list.</li> <li>• If the caller is running in AR-mode and WORBAREA was specified using explicit register notation, ensure that the corresponding access register was updated appropriately.</li> <li>• If the caller is disabled, then WORBAREA must reside in either nonpageable or disabled reference storage.</li> <li>• If your program is running in AR mode, ensure that you specified SYSSTATE ASCENV=AR before issuing the IXLCACHE macro.</li> </ul>
8	xxxx08AA	<p><b>Equate Symbol:</b> IXLRNCODEELEMNUMMISMATCH</p> <p><b>Meaning:</b> Program error. The specified data area size in the write-operation-block does not match the actual size of the corresponding data area in the data block. The data is not written. The index of the write-operation-block that failed and the offset in the data block of the data area for the write-operation-block being processed are returned in the answer area (fields CAAWDLINDEX, and CAAWDLDATAOFFSET). All prior write-operation-blocks were processed.)</p> <p><b>Action:</b> None required.</p>
8	xxxx08AB	<p><b>Equate Symbol:</b> IXLRNCODEBADDATAOFFSET</p> <p><b>Meaning:</b> Program error. A DATAOFFSET was specified that is not valid. No entries are processed and no data is returned.</p> <p><b>Action:</b>None required.</p>
8	xxxx08AC	<p><b>Equate Symbol:</b> IXLRNCODEBADGETCOLOCK</p> <p><b>Meaning:</b> Program error. In the write-operation-block the change control indicator was set and the get castout lock control indicator was also set. The data is not written. The cast-out lock is not obtained and the index of the failing write-operation-block is returned in the answer area (field CAAWDLINDEX). None of the specified write-operation-blocks were processed. Processing of the entire request was suppressed.</p> <p><b>Action:</b> None required.</p>

Table 47. Return and Reason Codes for the IXLCACHE REQUEST=WRITE\_DATALIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx08AD	<p><b>Equate Symbol:</b> IXLRNCOBDBADHIGHSHAREDVIRT</p> <p><b>Meaning:</b> Program error. The request specified a high shared virtual storage area (above 2GB).</p> <p><b>Action:</b> None required.</p>
8	xxxx08AF	<p><b>Equate Symbol:</b> IxlRsnCodeBadWrtSuppressCntl</p> <p><b>Meaning:</b> On a WRITE_DATALIST request, LOCALREGCNTL=YES was specified, but the change control indicator and assignment suppression indicator were not set in a WOB. The data is not written, and the index of the failing write-operation block is returned in the ANSAREA. None of the specified write-operation blocks were processed. Processing of the entire command was suppressed.</p> <p><b>Action:</b> The assignment suppression control indicator and change control indicator must be set in each write-operation block when LOCALREGCNTL=YES is specified. Correct the content of the WOB in error and reissue the request.</p>
C	xxxx0C06	<p><b>Equate Symbol:</b> IXLRNCOBDBADHIGHSHAREDVIRT</p> <p><b>Meaning:</b> Environmental error. No connectivity to the cache structure exists. This may occur due to operator commands such as VARY PATH,OFFLINE or CONFIG CHP,OFFLINE or hardware errors such as facility or path failures. The CONTOKEN is invalidated. The request fails.</p> <p><b>Action:</b> Either disconnect from the structure (using IXLDISC) or initiate a rebuild of the structure, if allowed (using IXLREBLD).</p>
8	xxxx08B9	<p><b>Equate Symbol:</b> IXLRNCOBDBADHIGHSHAREDVIRT</p> <p><b>Meaning:</b> An answer area was not specified when one is required. The requested service determined that conditions exist that require an ANSAREA to complete the request.</p> <p><b>Action:</b> Provide an answer area (ANSAREA) and answer area length (ANSLEN) on the IXLCACHE macro invocation for the request. ANSAREA is required when the connection specified ASYNCCI=1 on the IXLCONN invocation when connecting to the cache structure, AXIOVERRIDE=0 was specified or defaulted to for the IXLCACHE request and the request is one of the following:</p> <ul style="list-style-type: none"> <li>• CROSS_INVAL</li> <li>• CROSS_INVALLIST</li> <li>• DELETE_NAME</li> <li>• DELETE_NAMELIST</li> <li>• READ_DATA</li> <li>• REG_NAMELIST</li> <li>• WRITE_DATA</li> <li>• WRITE_DATALIST</li> </ul>

Table 47. Return and Reason Codes for the IXLCACHE REQUEST=WRITE\_DATALIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C13	<p><b>Equate Symbol:</b> IXLRSNCODEREQPURGED</p> <p><b>Meaning:</b> Environmental error. The request was purged prior to completion of the request. Possible reasons include:</p> <ul style="list-style-type: none"> <li>• The connector failed.</li> <li>• The connector disconnected.</li> <li>• The requestor failed.</li> <li>• The request was purged by IXLPURGE.</li> <li>• Requests were purged when the connector provided an IXLEERSP response for the Rebuild Quiesce, Rebuild Stop, or Rebuild Cleanup event.</li> <li>• The connector invoked IXLREBLD REQUEST=COMPLETE.</li> <li>• The secondary address space was no longer valid.</li> </ul> <p><b>Action:</b> None if this is expected. Otherwise, determine why the connector failed.</p>
C	xxxx0C14	<p><b>Equate Symbol:</b> IXLRSNCODESTATUSUNKNOWN</p> <p><b>Meaning:</b> Environmental error. The IXLCACHE request has completed, but the final disposition of the request cannot be determined. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Verify the validity of your data by comparing the expected results with what is in the coupling facility.</p>
C	xxxx0C17	<p><b>Equate Symbol:</b> IXLRSNCODESTRFULL</p> <p><b>Meaning:</b> Environmental error. Allocation of a directory entry was necessary, but was unavailable or could not be reclaimed. The data was not written. In the answer area, CAASTGCLFULL contains the storage class from which the reclaiming operation failed, CAAWDLINDEX contains the index of the write-operation-block that was being processed when the error occurred, and CAAWDLDATAOFFSET contains the offset in the data block of the data area for the write-operation-block being processed. All prior write-operation-blocks were processed.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Determine if any data items may be cast-out to make room for this item.</li> <li>• Check your usage of storage classes to see if some data items can be moved to a different storage class (preferably with a lower priority) so some entries in the structure can be freed.</li> <li>• Determine if a rebuild or an alter of the structure is necessary to make room for more data entries/items.</li> </ul> <p>After correcting the error, restart the WRITE_DATALIST request starting with the write-operation-block indexed by CAAWDLINDEX.</p>

Table 47. Return and Reason Codes for the IXLCACHE REQUEST=WRITE\_DATA LIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C25	<b>Equate Symbol:</b> IXLRSNCODESTRFAILURE <b>Meaning:</b> Environmental error. The cache structure failed prior to completion of the request. <b>Action:</b> Attempt to rebuild the structure using IXLREBLD, or disconnect from the structure using IXLDISC.
C	xxxx0C68	<b>Equate Symbol:</b> IXLRSNCODEBADREQCFLEVEL <b>Meaning:</b> Environmental error. The request type is not permitted for the level of coupling facility in which the target structure is allocated. <b>Action:</b> Disconnect from the structure (using IXLDISC) and request that the installation provide a coupling facility of the correct CFLEVEL (CFLEVEL=12 or higher).
C	xxxx0CA0	<b>Equate Symbol:</b> IXLRSNCODEQUIESCEDSUSPENDFAIL <b>Meaning:</b> Environmental error. The request is failed because the structure is quiesced for a system-managed process and SUSPEND=FAIL is specified on the IXLCONN. <b>Action:</b> None, if this is expected.
C	xxxxFFFF	<b>Equate Symbol:</b> IXLRSNCODENOTAVAILABLE <b>Meaning:</b> Environmental error. XES functions are not available. The coupling facility hardware might not be present. <b>Action:</b> XES may not be used when XCF is running local mode. XES requires that the coupling facility be installed.
10	xxxx10xx	<b>Equate Symbol:</b> IXLRSNCODECOMPERROR <b>Meaning:</b> System error. XES processing failure. The state of the structure and the disposition of the request are unpredictable. The answer area (ANSAREA) fields do not contain valid information. <b>Action:</b> Save the reason code information, and contact the IBM support center.

## Chapter 50. IXLCONN — Connect to a coupling facility structure

### Description

IXLCONN allows you to allocate and connect to a structure in the coupling facility or to connect to an already allocated structure. The first user to allocate the structure defines the structure attributes. Subsequent users can connect to the allocated structure but cannot change the structure attributes established when the structure was allocated. All connectors, whether the first or subsequent, are informed of the structure attributes through the connect answer area, mapped by IXLYCONA. It is the connector's responsibility to check the structure attributes to verify their acceptability.

#### • Providing Structure Information

Structure attributes include a structure name (STRNAME), a structure disposition (STRDISP), and a structure type (TYPE).

- You identify the structure to which you want to connect by name (STRNAME). For connections to be successful, structure names must be defined in the active CFRM policy, and the system on which you are running must be able to access the CFRM couple data set that contains the active CFRM policy. The system on which you are running also must have connectivity to a coupling facility that is in the preference list of the structure, as defined in the active CFRM policy.
- STRDISP determines whether the structure remains allocated when all users have disconnected from the structure.
- TYPE indicates the kind of structure — a cache, list, or lock structure. You can specify one of these types on an IXLCONN request. A set of attributes specific to each structure type allows you to define structure characteristics (for example, the size of data entries for a cache or list, or the number of lock users).

#### • Providing Connection Information

You also must specify certain information about the connection on IXLCONN, such as the connection disposition (CONDISP), an event exit (EVENTEXIT), and an answer area (ANSAREA) and length (ANSLEN).

- CONDISP determines whether a connection can enter a failed-persistent state which, in the event of a connection failure, would allow you to reconnect to the structure, if possible. Users can specify a connect name (CONNAME) on IXLCONN that they can use to reconnect to the structure if they fail, enter a failed-persistent state, and are subsequently restarted.
- EVENTEXIT specifies the name of the event exit. The event exit gets control whenever an event about the structure or a connection to the structure (such as a failure of the connection) occurs. The system reports an event to the event exit of each connected user. Users establish protocols to process these events.
- IXLCONN returns information about the structure and the connection in an area specified by ANSAREA. Included in this area is a connect token (CONTOKEN) that uniquely identifies the connection within the sysplex. After you have connected to a structure, you specify the CONTOKEN returned from IXLCONN on structure requests like IXLCACHE, IXLLIST, IXLLSTC, IXLLSTE, IXLLSTM, or IXLOCK, as appropriate to the type of structure to which you've connected.

When initially connecting to a structure, you have the option of specifying how long SVC dump holds serialization for a structure when a dump for the connected structure occurs (ACCESSTIME). You also can specify the level of coupling facility that your application desires (CFLEVEL) as well as the minimum CFLEVEL required (MINCFLEVEL).

#### • Providing Modification Information

Your connection can support rebuilding the structure (ALLOWREBLD) and altering the structure (ALLOWALTER). If the connection is to a cache structure, your connection might also support duplexing the structure (ALLOWDUPREBLD).

- Structure rebuild is a process intended for planned reconfiguration and recovery scenarios, in which connectors to a structure allocate another structure with the same name and rebuild data (if applicable) into the new structure.
- Structure alter is a process that allows a structure's size, entry-to-element ratio, and, if applicable, percent of structure storage allocated to event monitor controls to be changed. For changes to structure size and entry-to-element ratio, structure alter requires that the structure be allocated in a coupling facility with CFLEVEL=1 or higher. For event monitor control related changes, the list structure must be allocated in a coupling facility with CFLEVEL=3 or higher for a change in size and entry-to-element ratio, or in a coupling facility with CFLEVEL=4 or higher for a change to the percentage of event monitor controls, and the connectors must be running on a system with the appropriate level of system upgrades.
- Duplexing rebuild is a process that allows duplexing of a cache structure to be managed by the structure's connectors.

Starting with OS/390 Release 8, your connection can also support the use of system-managed processes (ALLOWAUTO). When a process is system-managed, the system (as opposed to the connectors) performs some, if not all, of the significant steps in the process. Release 8 provides system-managed rebuild processing. Its primary use is in a planned reconfiguration scenario, for example when moving structures from one coupling facility to another. Even when an application does not support user-managed rebuild (ALLOWREBLD=NO), system-managed rebuild can be used to accomplish the movement of the structures.

Starting with OS/390 Release 10, the installation can specify through its CFRM policy that system-initiated alter processing is requested for a structure. ALLOWAUTOALT(YES) in the CFRM policy indicates that when the structure reaches a specified percent-full threshold, the system is to automatically begin alter processing to relieve the structure storage shortage and also the reclaiming criterion.

IBM recommends that connectors support the structure alter process (ALLOWALTER=YES) because it allows the installation to specify the automatic alter function for the structure. The system then can monitor and tune the structure size and ratios as needed, within the bounds provided by the installation.

IBM also recommends that connectors that support system-managed processes (ALLOWAUTO=YES) should also support the structure alter process (ALLOWALTER=YES). This gives the system the greatest flexibility in allocating new structures during system-managed processes and provides the other benefits that go along with structure alter in general.

The security administrator might have controlled the use of installation structures through the use of RACF or another security product. If so, ensure that you are authorized to issue the IXLCONN macro for the structure.

For information about connection services, see [z/OS MVS Programming: Sysplex Services Guide](#).

## Environment

The following are the environment requirements for the caller.

Environment	Environment requirement
Authorization:	Supervisor state or PKM keys 0 - 7
Dispatchable unit mode:	Task
Cross memory mode:	PASN=HASN, any SASN
AMODE:	31-bit
ASC mode:	Primary or access register (AR)



Environment	Environment requirement
Interrupt status:	Enabled for I/O and external interrupts
Locks:	No locks held, with no enabled, unlocked task (EUT) FRRs established
Control parameters:	Must be in the primary address space or be in an address/data space that is addressable through a public entry on the caller's dispatchable unit access list (DU-AL)

## Restrictions

- You cannot issue the IXLCONN macro from the master address space.
- The maximum number of connections to a structure in the coupling facility depends on:
  - The number of CONNECT records in the active CFRM policy.  
The maximum number of CONNECT records for a structure is defined by input to the CFRM couple data set format utility, IXCL1DSU. See a description of the ITEM NAME(CONNECT) NUMBER() parameter for the CFRM data type in *z/OS MVS Setting Up a Sysplex*.  
The default number of CONNECT records you can specify is 32; the maximum allowed is 255.
  - The limit imposed by the coupling facility control code.  
This limit is model-dependent. CFLEVEL=0 or 1 for a coupling facility allows up to 32 connections for all structure types. For CFLEVEL values that are in the range CFLEVEL=2 to CFLEVEL=16 including a CFLEVEL=16 coupling facility allows up to 255 connections for a cache structure and 32 connections for a list or lock structure. A CFLEVEL=17 or higher coupling facility allows up to 255 connections for all structure types.
  - The application-specified value of the NUMUSERS or MAXCONN keyword on a request to connect to a list structure.
  - The application-specified value of the NUMUSERS or MAXCONN keywords on a request to connect to a lock structure.
  - The lock table entry size for serialized list and lock structures. The architected z/OS limit on the lock table entry size for serialized list structures limits the number of connectors that can be represented by the lock table entry to 127. The architected z/OS limit on the lock table entry size for lock structures limits the number of connectors that can be represented by the lock table entry to 247.

## Input register information

Before issuing the IXLCONN macro, the caller does not have to place any information into any register unless the caller is using it in register notation for a particular parameter, or using it as a base register.

## Output register information

When control returns to the caller of the IXLCONN macro, the general purpose registers (GPRs) contain:

### Register Contents

- 0** Reason code, if applicable, if GPR 15 return code is nonzero
- 1** Used as work register by the system
- 2-13** Unchanged

**14**

Used as work register by the system

**15**

Return code

When control returns to the caller of the IXLCONN macro, the access registers (ARs) contain:

**Register  
Contents**

**0-1**

Used as work registers by the system

**2-13**

Unchanged

**14-15**

Used as work registers by the system

## Programming requirements

---

If the program is in AR mode, issue the SYSSTATE ASCENV=AR macro before IXLCONN. SYSSTATE ASCENV=AR tells the system to generate code appropriate for AR mode.

The IXLYCONA macro provides the format of the area that the ANSAREA parameter points to. Include the IXLYCONA macro in your program.

## Performance implications

---

IXLCONN connect processing involves updating the CFRM active policy to reflect the new connection, allocating the structure in the coupling facility, and/or connecting the user to the structure in the coupling facility. The IXLCONN invoker is suspended while the new connection is notified of all peer connections, as well as all peer connections being notified of the new connection.

## Understanding IXLCONN version support

---

The IXLCONN macro supports versions in the range of 0 - 10.

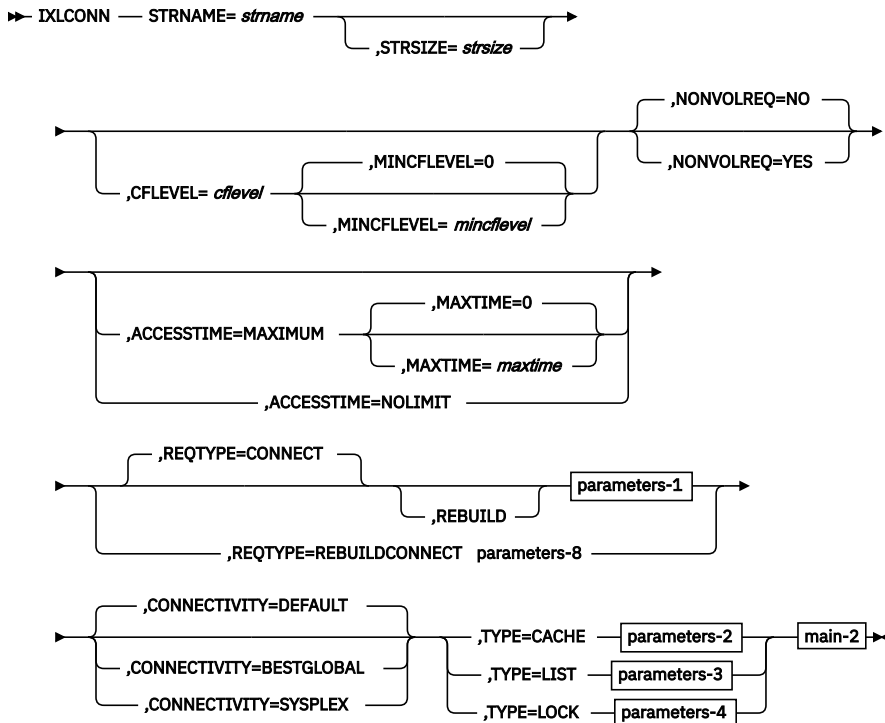
- Keywords not specifically noted here are supported by version 0 and subsequent versions of the IXLCONN macro.
- The following keywords and functions are supported by version 1 and subsequent versions of the IXLCONN macro: ALLOWALTER and MINELEMENT; MINENTRY and RATIO
- The following keyword is supported by version 2 and subsequent versions of the IXLCONN macro: EMCSTGPCT
- The following keywords are supported by version 3 and subsequent versions of the IXLCONN macro: CONNECTIVITY and RNAMELEN
- The following keyword is supported by version 4 and subsequent versions of the IXLCONN macro: MINEMC
- The following keywords are supported by version 5 and subsequent versions of the IXLCONN macro: ALLOWDUPREBLD and UDFORDER
- The following keyword is supported by version 6 and subsequent versions of the IXLCONN macro: NAMECLASSMASK
- The following keywords are supported by version 7 and subsequent versions of the IXLCONN macro: ALLOWAUTO and SUSPEND
- The following keywords are supported by version 8 and subsequent versions of the IXLCONN macro: ENTRYIDTYPE, KEYTYPE, MINCFLEVEL
- The following keywords are supported by version 9 and subsequent versions of the IXLCONN macro: ASYNCDUPLEX, CRITICAL, FUNCTION, PCTENTRYRSV, TERMLEVEL, and ASYNCXI

- The following keywords are supported by version 10 and subsequent versions of the IXLCONN macro:  
MONITOR and MONITORVAL.

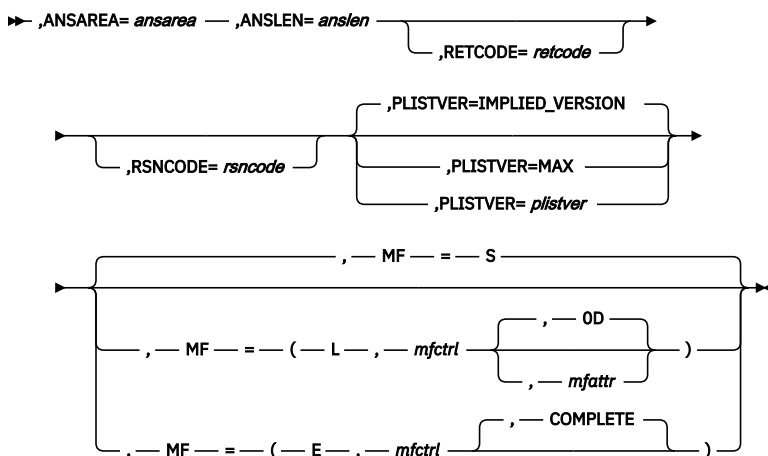
Specify the version of the parameter list that you want generated with the PLISTVER keyword. See Chapter 2, “Specifying a Macro Version Number,” on page 5 for considerations when specifying the version of the parameter list with PLISTVER.

## Syntax diagram

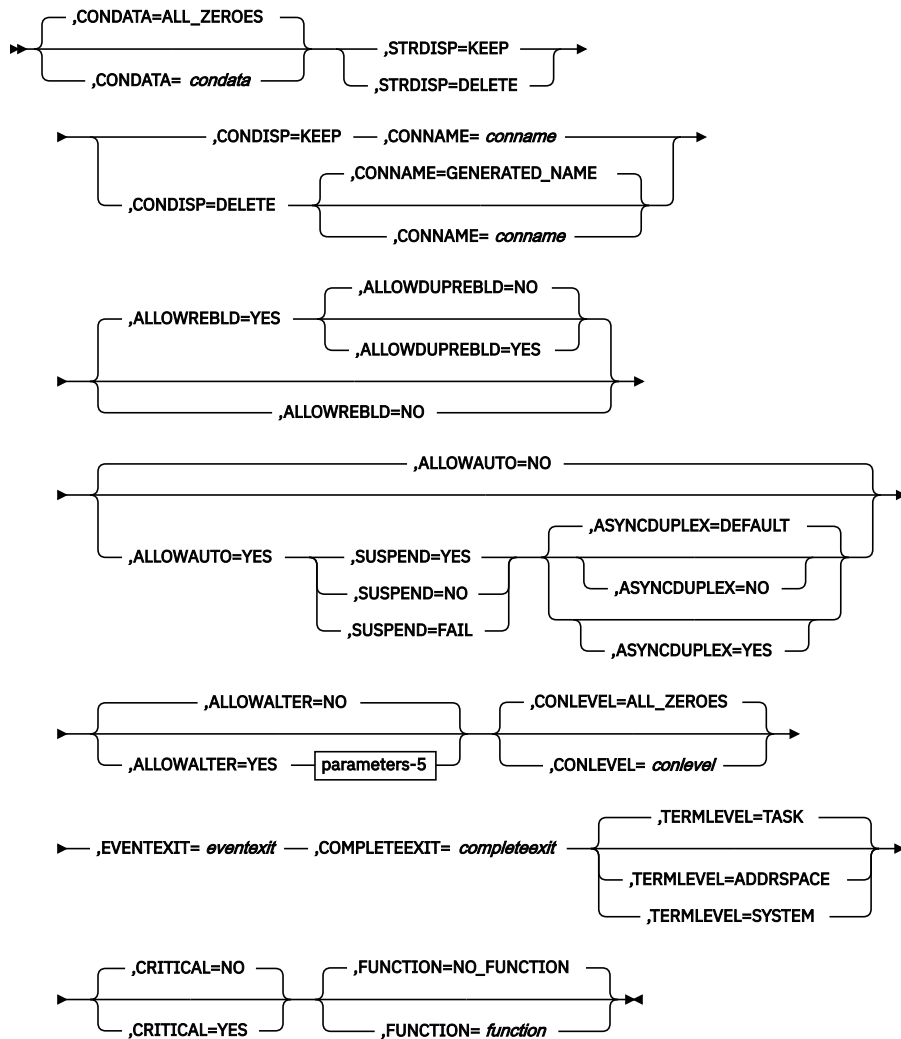
### main diagram



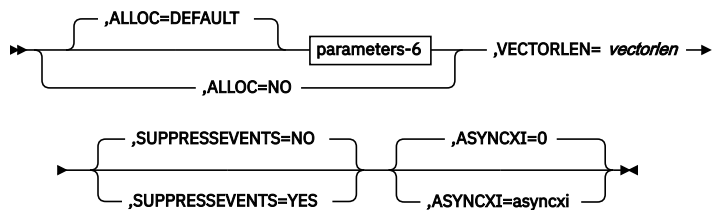
### main-2



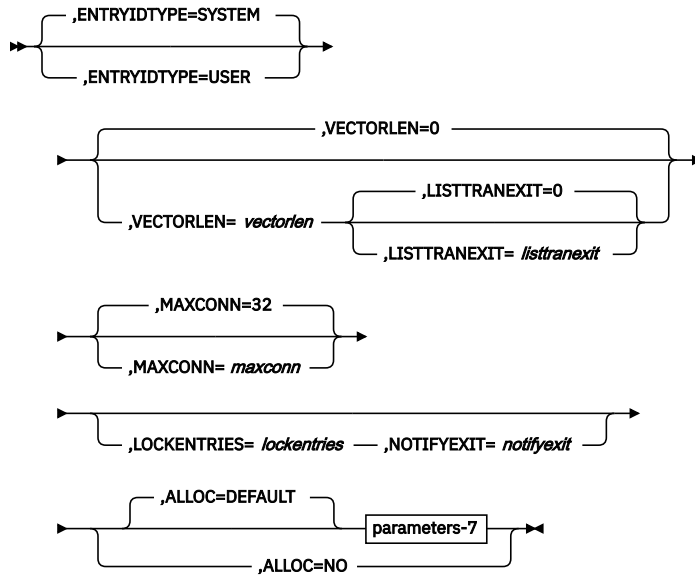
## parameters-1



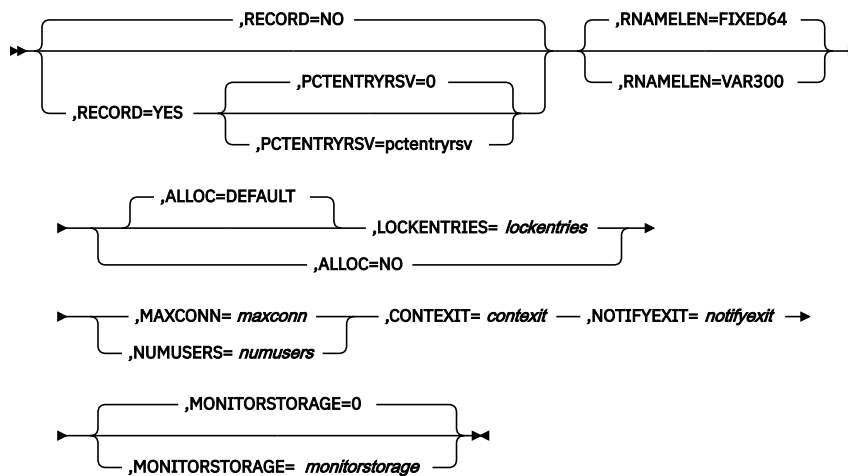
## parameters-2



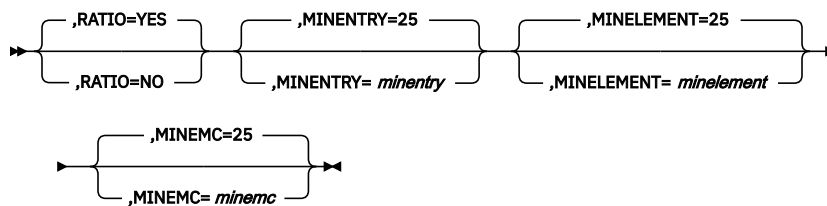
## parameters-3



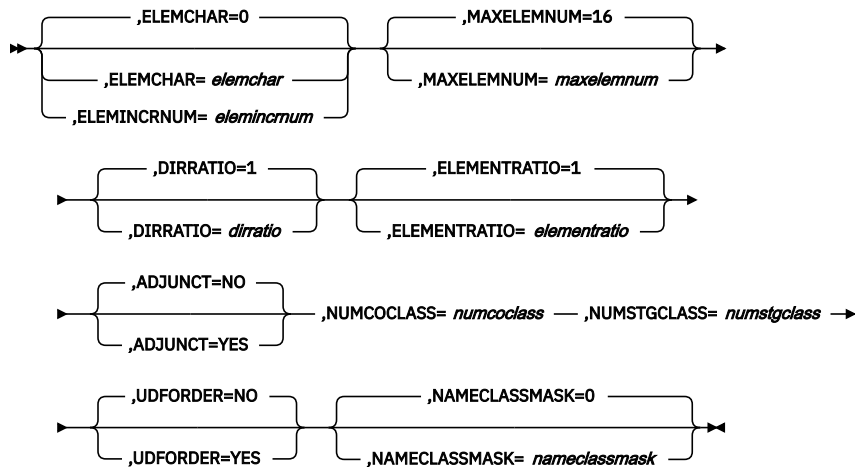
## parameters-4



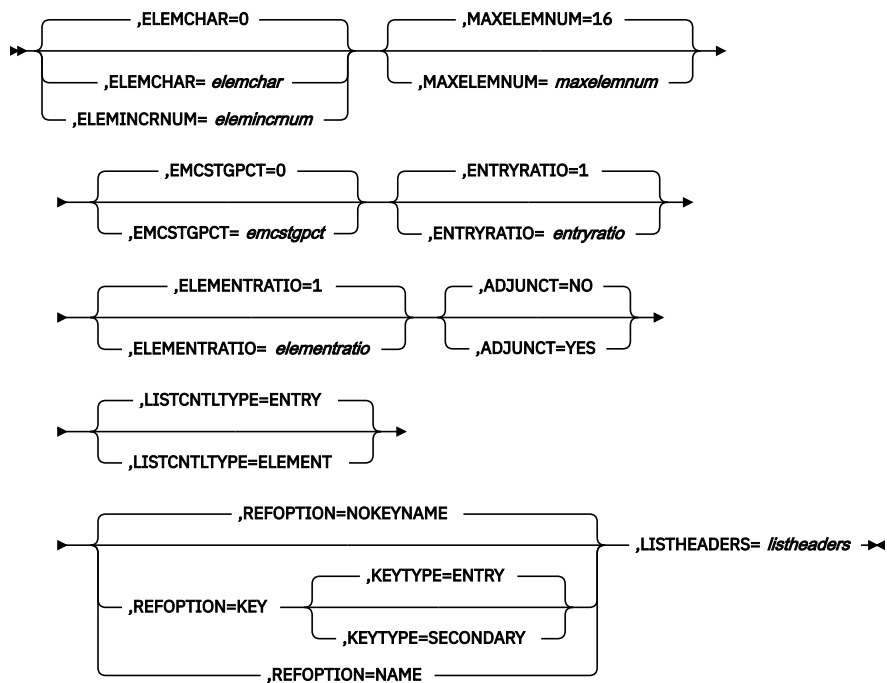
## parameters-5



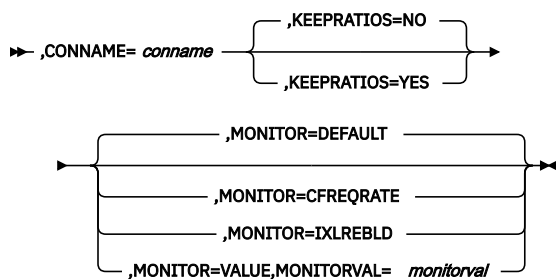
## parameters-6



## parameters-7



## parameters-8



## Parameter descriptions

The parameters that are common to all structure types follow in alphabetical order. The additional parameters that are specific to each structure type are described in “Parameters for TYPE=CACHE” on page 871, “Parameters for TYPE=LIST” on page 874, and “Parameters for TYPE=LOCK” on page 879. Default values are underlined.

### Parameters common to all structure types

#### **,ACCESSTIME=MAXIMUM**

#### **,ACCESSTIME=NOLIMIT**

Use this input parameter to specify the length of time SVC dump holds serialization for the structure when a dump of the structure is requested. This indicates the length of time that a connector can tolerate not having access to the structure. When SVC dump holds serialization on a structure, the system denies new connections to the structure and delays new requests from existing connections to access the structure.

If the user codes ACCESSTIME=MAXIMUM, the maximum time that serialization is held is determined by the first connection that allocates the structure with the value that is specified for MAXTIME.

If the user codes ACCESSTIME=NOLIMIT, serialization is held for as long as it is required to capture the data from SVC dump or until the structure dump or its serialization would be forced with either the IXLFORCE macro or the SETXCF FORCE command.

ACCESSTIME is set by the first connector to the structure. In the IXLYCONA answer area, the fields CONAACCESSTIME AND CONAACCESSTIMENOLIMIT contain the values pertinent to the allocation of the structure.

**Note:** SVC dump is supported for cache, list, and serialized list structures only. ACCESSTIME is not valid for lock structures.

#### **,ALLOC=DEFAULT**

#### **,ALLOC=NO**

Use this input parameter to indicate whether the request can be used to allocate the structure.

#### **DEFAULT**

The request can be used to allocate the structure, if necessary. This is intended to only be used as a default - not actually specified on the IXLCONN invocation.

#### **NO**

The request should not be used to allocate the structure. If the structure is not allocated, the request will fail with IXLRSNCODESTRNOTALLOCATED.

When ALLOC=NO is specified, IXLCONN parameters that are only used for structure allocation are ignored. The following IXLCONN parameters are ignored:

- STRSIZE
- STRDISP
- ACCESSTIME
- MAXTIME
- CONNECTIVITY

With the exception of STRDISP, ignored parameters do not need to be specified.

When running on a system without support for the ALLOC key, specifying ALLOC will cause the request to fail with different reason codes depending upon the structure type:

#### **TYPE=CACHE**

IXLRSNCODEINVALIDSTGCLASS

#### **TYPE=LIST**

IXLRSNCODENOLISTHDRS

**TYPE=LOCK**

IXLRSNCODENOLENTRIES

**,ALLOWALTER=NO****,ALLOWALTER=YES**

Use this input parameter to indicate whether this connection allows structure alter to be initiated against the structure. Structure alter can be used to expand or contract the size of a structure, to reapportion the entry-to-element ratio, and to reapportion the event monitor controls (EMC) storage percentage. One or more of these changes can be requested concurrently.

The structure alter procedure is done in place on the previously allocated structure. Structure alter provides a non-disruptive alternative to rebuild for changes to be made to the structure. Changes other than those listed, such as changing the location of a structure, must be accomplished with the rebuild function.

The MINENTRY, MINELEMENT, and MINEMC parameters are used when a structure alter is initiated to contract a structure or reapportion its ratio and/or its EMC storage, where that request would decrease the amount of storage available for entries, elements, or EMCs. These values prevent a structure alter from making a structure unusable to the application, as might occur if more free entries, elements, or EMC storage were removed or reallocated than the application could tolerate. If a structure alter tries to contract or reapportion more free space than specified by the MIN parameters, the alter will be stopped. The MIN values are not used during requests to expand the structure, or during reapportion requests which increase the amount of storage available for a particular storage type.

IBM recommends that connectors that support system-managed processes (ALLOWAUTO=YES) should also support the structure alter process (ALLOWALTER=YES). This gives the system the greatest flexibility in allocating new structures during system-managed processes.

**NO**

Indicates that this connection does not support structure alter for the structure.

**YES**

Indicates that this connection does support structure alter for the structure. When specifying ALLOWALTER=YES, you must also specify CFLEVEL=1 or higher.

**,ALLOWAUTO=NO****,ALLOWAUTO=YES**

Use this input parameter to indicate whether this connection supports system-managed processes (for example, rebuild). When a system-managed rebuild is in progress, any request to connect to the structure is rejected. Refer to the IXLREBLD documentation for a description of system-managed rebuild and the effect of the ALLOWAUTO keyword on initiating rebuild.

**NO**

Indicates that this connection does not support system-managed processes. An application that does not support structure rebuild (ALLOWREBLD=NO) and specifies or defaults to ALLOWAUTO=NO must provide its own support for planned reconfiguration of the coupling facility. The planned reconfiguration support might include normal shutdown procedures or an operator command interface. The application support must allow the operator to stop an application's use of a structure in a coupling facility.

**YES**

Indicates that this connection supports system-managed processes. The application support for system-managed processes includes the following:

- Processing the Structure Temporarily Unavailable and Structure Available events.
- Using extended restart tokens.
- Evaluating the information presented with the Structure State Change event and reacting appropriately to changes in structure characteristics.
- Using both physical structure version numbers to uniquely identify an instance of a structure. (In IXLCONA, the physical version numbers are identified by CONAPHYSICALSTRUCTUREVERSION



and CONAPHYSICALSTRUCTUREVERSION2; in IXYEEPL, the physical version numbers are identified by EEPLSSCSTRPHYSICALVERSION and EEPLSSCSTRPHYSICALVERSION2).

**,ALLOWDUPREBLD=NO**

**,ALLOWDUPREBLD=YES**

Use this input parameter to indicate whether this connection supports user-managed duplexing rebuild. This keyword applies only to cache structures; the keyword is ignored when the structure type is list or lock. This keyword is used in conjunction with the DUPLEX specification for the structure in the active CFRM policy.

If any active or failed-persistent connection to the structure specified ALLOWDUPREBLD=NO, the system will reject a request to start duplexing the structure.

When a duplexing rebuild is in progress, if any new request to connect to the structure specifies ALLOWDUPREBLD=NO, the system will reject the connection request.

Duplexing rebuild allows connectors to a structure to allocate another structure with the same name as the original structure. It allows connectors to duplex data into both structures. Use the duplexing rebuild procedure for improved availability and usability

If ALLOWDUPREBLD=YES is specified, a protocol for duplexing events must be supported in the event exits of connectors participating in the duplexing rebuild process.

**NO**

Indicates that this connection does not support duplexing rebuild for the structure.

**YES**

Indicates that this connection does support duplexing rebuild for the structure.

**,ALLOWREBLD=YES**

**,ALLOWREBLD=NO**

Use this input parameter to indicate whether this connection allows structure rebuild to be initiated against this structure. Structure rebuild allows connectors to a structure to allocate another structure with the same name as the original structure, and to rebuild data in the new structure, if applicable. Structure rebuild can be used to change the location and certain attributes of a structure. Use the structure rebuild procedure for planned reconfiguration and for recovery.

If any active connection to the structure specified ALLOWREBLD=NO and specified or defaulted to ALLOWAUTO=NO, the system will reject a request to start rebuilding the structure.

If all active connections specified or defaulted to ALLOWREBLD=YES, the specification of ALLOWAUTO is not considered.

Note that if structure rebuild is allowed, a protocol for rebuild events must be supported in the event exits of connectors participating in the rebuild process.

**YES**

Indicates that this connection supports structure rebuild for the structure. Refer to the IXLREBLD documentation for a description of user-managed rebuild and the effect of the ALLOWREBLD keyword on initiating rebuild.

**NO**

Indicates that this connection does not support structure rebuild for the structure. If an application chooses not to support structure rebuild (specifies ALLOWREBLD=NO and specifies or defaults to ALLOWAUTO=NO), then the application must provide its own interfaces for planned reconfiguration of the coupling facility. The planned reconfiguration support might include normal shutdown procedures or an operator command interface. The application support must allow the operator to stop an application's use of a structure in a coupling facility.

**,ANSAREA=ansarea**

Use this output parameter to identify the answer area. When IXLCONN completes, the macro returns information to the answer area. The requester can use the IXLYCONA macro to map the answer area. ANSAREA must begin on a doubleword boundary.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword output field that identifies the answer area.

**,ANSLEN=anslen**

Use this input parameter to specify the length of the answer area. The length should be long enough to accommodate the IXLYCONA mapping of the answer area.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword input field that contains the length of the answer area.

**,ASYNCDUPLEX=DEFAULT****,ASYNCDUPLEX=NO****,ASYNCDUPLEX=YES**

Use this input parameter to indicate whether this connection supports system-managed asynchronous structure duplexing.

**DEFAULT**

For TYPE=LOCK,RECORD=NO, indicates that this connection supports system-managed asynchronous structure duplexing. For other types of structures, indicates that this connection does not support system-managed asynchronous structure duplexing.

**NO**

Indicates that this connection does not support system-managed asynchronous structure duplexing.

**YES**

Indicates that this connection supports system-managed asynchronous structure duplexing. ASYNCDUPLEX=YES is only applicable to lock structures. See the section on working with structures in the async duplex established phase in *z/OS MVS Programming: Sysplex Services Guide* for information about the requirements for this support.

**,CFLEVEL=cflevel**

Use this input parameter to specify that the connector wants the structure to be allocated in a coupling facility that supports (at least) the indicated level. The connector should request the highest CFLEVEL necessary to process the coupling facility requests the connector anticipates submitting against the structure.

Do not request a level based on whether the connector supports system-managed processes (ALLOWAUTO keyword). It is neither necessary nor desirable to request the CFLEVEL required by a particular system-managed process simply because ALLOWAUTO=YES is specified. (For example, do not specify CFLEVEL=8 for the purpose of supporting system-managed rebuild.) If you specify ALLOWAUTO=YES, the system will automatically attempt to allocate your structure in a coupling facility that supports system-managed processes. Therefore, your CFLEVEL specification should reflect only the level required for the requests that you expect to submit against the structure.

Do not request a level that is based on whether the connector supports system-managed asynchronous duplexing (ASYNCDUPLEX keyword). It is neither necessary nor desirable to request the CFLEVEL required by a particular duplexing mode simply because ASYNCDUPLEX=YES is specified. (For example, do not specify CFLEVEL=21 for the purpose of supporting system-managed asynchronous duplexing.) The system will automatically attempt to allocate your structure in a coupling facility that supports system-managed asynchronous duplexing, when appropriate. Therefore, your CFLEVEL specification should reflect only the level required for the requests that you expect to submit against the structure.

On a successful connect, the field CONACFACILITYCFLEVEL indicates the level of operations that can be performed against the structure. CONAMVSRELEASEMAXCFLEVEL indicates the level of operations supported by the operating system on which the connector is running. These coupling facility levels can differ from the requested level, so the connector must verify that the level returned is suitable for its use. The connector must not issue operations that require a coupling facility level greater than the lower of CONAMVSRELEASEMAXCFLEVEL and CONACFACILITYCFLEVEL.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the fullword input field that contains the level of coupling facility required.

**,COMPLETEEXIT=completeexit**

Use this input parameter to identify the complete exit for the requester. The complete exit receives control in SRB mode, enabled, and unlocked.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the complete exit for the requester.

**,CONDATA=ALL\_ZEROES**

**,CONDATA=condata**

Use this input parameter to specify eight bytes of connector data. The data is supplied to all of the connector's exits and has no meaning to the system.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the 8-character input field that contains the connector data.

**,CONDISP=KEEP**

**,CONDISP=DELETE**

Use this input parameter to specify the persistence of the connection after the requester abnormally terminates or disconnects with REASON=FAILURE.

**KEEP**

Indicates that the connection to the structure is to be placed in a failed-persistent state. Peer connectors, notified of the Disconnected/Failed Connection event through their event exits, must respond either with the event exit return code or IXLEERSP.

After all peer connectors have responded to the event, XES uses the CONDISP specification to determine whether to make the connector be failed-persistent or not defined. If peer connectors were able to recover for the failing user, they may override the CONDISP=KEEP parameter to indicate that the connector does not need to be made failed-persistent.

Once a connector is placed in a failed-persistent state, the failed connector can try to reconnect to the structure by reissuing IXLCONN with the same CONNAME specified on the original IXLCONN macro. (See CONNAME.)

If the user is unable to reconnect or chooses not to do so, the following options exist to delete the failed-persistent connection:

- The operator or extended MCS console interface can issue the SETXCF FORCE command to delete the connection.
- A connected user or authorized program can issue the IXLFORCE macro to delete the connection.

**DELETE**

Indicates that the connection to the structure is to be put in the not-defined state after the requester disconnects or terminates abnormally. Prior to being placed in the not-defined state, peer connectors must have been notified of the Disconnected/Failed Connection event and have responded to confirm the failure.

**,CONLEVEL=ALL\_ZEROES**

**,CONLEVEL=conlevel**

Use this input parameter to specify the connector's processing level that is to be communicated to peer connectors through event exit invocation. The level can indicate a version or release level, a functionality level, or any other connector-defined option.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-character input field to contain the connector's processing level.

**,CONNAME=GENERATED\_NAME**

**,CONNAME=conname**

Use this input parameter to specify the unique connection name to identify the user request. CONNAME must be unique for a given structure.

The connection name must be 16 characters long, padded on the right with blanks, if necessary. The name can contain numeric characters, uppercase alphabetic characters, national characters (\$, @, #), or an underscore (\_). Connection names must begin with an uppercase alphabetic character or a national character.

Connection names must be unique for each user connected to a structure. If CONNAME for the request matches the name of another active connection, the system rejects the request.

CONNNAME is required for CONDISP=KEEP. To reconnect, a requester can issue IXLCONN with CONNAME. If the CONNAME matches the name of the failed-persistent connection, then the connector will be reconnected. When a requester reconnects to the structure, the system returns a new connect token to the connect answer area and sets return code X'4' that the user has reconnected. A reason code of IXLRSCODESPECIALCONN also is returned to indicate that the connector should examine IXLYCONA to be aware of any changes that might have been made on the reconnect.

CONNNAME is optional for CONDISP=DELETE. If a user does not specify CONNAME, the system generates a name. Note however, that CONNAME must be specified when IXLCONN REBUILD is issued, even if CONDISP=DELETE. CONNAME must match the connection name returned in IXLYCONA for the original connection.

If an event that the system reports to the event exit is about a connection to the structure, the system reports the connection's CONNAME to the event exit of all connected users.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the 16-character field that contains the unique connection name to identify the user request.

**,CONNECTIVITY=DEFAULT**

**,CONNECTIVITY=BESTGLOBAL**

**,CONNECTIVITY=SYSPLEX**

Use this input parameter to indicate the required scope of system connectivity to the coupling facility in which the structure is to be allocated.

Use this parameter with the REBUILD option of IXLCONN if you want the system to apply the same allocation rules for a structure that is to be rebuilt as for the initial allocation of the structure. During rebuild processing, the system will select a coupling facility based on the current set of active connectors to the old structure.

**CONNECTIVITY=DEFAULT**

Specifies that the system is to use the coupling facility selection algorithm to select the coupling facility in which to allocate the structure. The system considers the amount of system connectivity (as specified in the SFM policy, if active) when selecting a coupling facility, but only to decide between coupling facilities that are equal in all other respects (such as having the requested CFLEVEL and volatility attributes, etc.). If there is no active SFM policy, the system considers each system to be of equal weight.

**CONNECTIVITY=BESTGLOBAL**

Specifies that the system is to use the expanded coupling facility selection algorithm to select the coupling facility in which to allocate the structure. The system considers the amount of system connectivity before examining all other coupling facility attributes that have been requested. The system attempts to allocate the structure in a coupling facility that has the best connectivity to all systems in the sysplex.

**CONNECTIVITY=SYSPLEX**

Specifies that the system is to choose a coupling facility for structure allocation that is connected to all systems in the sysplex, whether other requested structure attributes are available or not. If the system cannot identify a coupling facility with connectivity to all systems in the sysplex, the IXLCONN request fails.

**,CRITICAL=NO**

**,CRITICAL=YES**

Use this input parameter to indicate whether this connector is a critical connector. A critical connector is one whose function is critical to the normal operation of the system.

For a connector to a serialized structure (TYPE=LOCK or TYPE=LIST with LOCKENTRIES), XES invokes IXCJOIN on behalf of the connector to define an XCF group member for the connector and the CRITICAL attribute will be propagated to the IXCJOIN invocation. If the critical XCF group member

appears to be impaired for too long, XCF will terminate the impaired member according to the TERMLEVEL specification.

See the CRITICAL keyword description on the IXCJOIN macro (Chapter 8, “IXCJOIN — Place an XCF Member in the Active State,” on page 79) for additional information about critical XCF group members.

#### **NO**

Indicates that the connector does not designate itself as a critical connector.

#### **YES**

Indicates that the connector designates itself as a critical connector. For serialized structures (TYPE=LOCK or TYPE=LIST with LOCKENTRIES), XCF will monitor the health of the XCF group member that was defined on behalf of the connector. If the member becomes impaired for too long, XCF will terminate the member according to the TERMLEVEL specification.

#### **,EVENTEXIT=*eventexit***

Use this input parameter to identify the event exit for the requester. The event exit receives control in SRB mode, enabled, and unlocked.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the event exit for the requester.

#### **,FUNCTION=NO\_FUNCTION**

#### **,FUNCTION=*function***

Use this input parameter to specify the function associated with the connector.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the 24-character input field that contains the description of the function, service, or application associated with the connector. The string can contain any alphanumeric (A-Z), national (@,#,\$), or special character (underscore or blank). Leading blanks and all blank descriptors are not permitted.

#### **,KEEPRATIOS=NO**

#### **,KEEPRATIOS=YES**

Use this input parameter for rebuild connects to list and cache structures to determine the structure allocation attributes related to the structure object ratios. Do not specify with TYPE=LOCK or ALLOC=NO.

#### **NO**

Indicates that rebuild connect processing is to use the parameter list to determine the structure allocation attributes related to structure object ratios.

#### **YES**

Indicates that rebuild connect processing is to use the current (old structure) attributes to determine the structure allocation attributes related to structure object ratios. When the system has support for this keyword (APAR OA33448), IXLCONN parameters that determine structure allocation attributes related to structure object ratios are to be ignored.

For list structures, the following IXLCONN parameters are ignored:

- ENTRYRATIO
- ELEMENTRATIO
- EMCSTGPCT
- LISTCNTLTYPE
- KEYSIZE
- REFOPTION
- ADJUNCT
- MAXELEMNUM
- ELEMCHAR
- ELEMINCRNUM.

For cache structures, the following IXLCONN parameters are ignored:

- DIRRATIO
- ELEMENTRATIO
- MAXELEMNUM
- ELEMCHAR
- ELEMINCRNUM.

When running on a system at z/OS V1R12 or earlier, use the ignored parameters as you normally would in case the system does not have support for the keyword. For a z/OS V1R12 system or later, with the keyword support you do not need to specify the ignored parameters. In addition to maintaining current structure object ratios, KEEPRATIOS=YES might also make it more likely for equivalent structure object counts to be achieved

**,MAXTIME=0**

**,MAXTIME=***maxtime*

Use this input parameter to specify the length of time in tenths of seconds that SVC dump holds serialization for the structure when the user specifies ACESSTIME=MAXIMUM.

**Note:**

1. MAXTIME is not valid for lock structures.
2. If the user specifies zero, SVC dump does not obtain serialization for the structure and dump data is not reported for the structure.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the halfword input field that contains the length of time that SVC dump holds serialization for the structure when the user specifies ACESSTIME=MAXIMUM.

**,MF=S**

**,MF=(L,***mfctrl***)**

**,MF=(L,***mfctrl***,***mfattr***)**

**,MF=(L,***mfctrl***,0D)**

**,MF=(M,***mfctrl***)**

**,MF=(M,***mfctrl***,COMPLETE)**

**,MF=(M,***mfctrl***,NOCHECK)**

**,MF=(E,***mfctrl***)**

**,MF=(E,***mfctrl***,COMPLETE)**

**,MF=(E,***mfctrl***,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

**,mfctrl**

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

**,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to

force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

**,COMPLETE**

**,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

**COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if *SMILE=var* were an optional parameter and the default is *SMILE=NO\_SMILE* then it would not be documented. However, if the default was *SMILE=-*), then it would be documented because a value would be the default.

**NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

**,MINCFLEVEL=0**

**,MINCFLEVEL=*mincflevel***

Use this input parameter to specify that the connector requires that the structure be allocated in a coupling facility that supports at least the indicated minimum CFLEVEL.

The value specified by MINCFLEVEL must be equal to or less than the value specified by CFLEVEL.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the fullword field that contains the minimum CFLEVEL requested.

**,MINELEMENT=25**

**,MINELEMENT=*minelement***

Use this input parameter to specify the minimum level (as a percent) of “in-use” elements that should exist at the end of the structure alter process. For a list structure, this value is a percent of the “currently-in-use” elements. For a cache structure, this value is a percent of the “currently-in-use-and-changed” elements.

When the specified minimum level is attained, IXLALTER processing will end. If the minimum level cannot be attained, IXLALTER processing will provide a percentage of available elements as close to the target level as possible.

The value of MINELEMENT must be in the range of from 0 to 100.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the one-byte field that contains the percent value.

**,MINEMC=25**

**,MINEMC=*minemc***

Use this input parameter to specify the minimum level (as a percent) of “in-use” EMCs that should exist at the end of the alter process. For a keyed list structure, this value is a percent of “currently in-use” EMCs.

When the specified minimum level is attained, IXLALTER processing will end. If the minimum level cannot be attained, IXLALTER processing will provide a percentage of available EMCs as close to the target level as possible.

The value of MINEMC must be in the range of 0 to 100, inclusive.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the one-byte field that contains the percent value.

**,MINENTRY=25****,MINENTRY=***minentry*

Use this input parameter to specify the minimum level (as a percent) of “in-use” entries that should exist at the end of the structure alter process. For a list structure, this value is a percent of the “currently-in-use” entries. For a cache structure, this value is a percent of the “currently-in-use-and-changed” entries.

When the specified minimum level is attained, IXLALTER processing will end. If the minimum level cannot be attained, IXLALTER processing will provide a percentage of available entries as close to the target level as possible.

The value of MINENTRY must be in the range of from 0 to 100.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the one-byte field that contains the percent value.

**,MONITOR=DEFAULT****,MONITOR=CFREQRATE****,MONITOR=IXLREBLD****,MONITOR=VALUE,MONITORVAL=***monitorval*

Use this input parameter to specify how the system is to monitor progress toward completion of structure repopulation during user-managed rebuild or duplexing rebuild. Repopulation refers to the establishment of exploiter data in the rebuild new structure instance in the interval after IXLCONN REQTYPE=REBUILDCONNECT and before IXLREBLD REQUEST=COMPLETE.

The system requires that connectors complete structure repopulation in a timely manner to avoid degrading overall sysplex performance. If a connector does not make satisfactory progress, as defined by the specified or defaulted MONITOR keyword, the system reports a hang and may take automatic action to resolve the situation, depending on the installation setting of the SFM policy CFSTRHANGTIME keyword. The system monitors each connector individually, and the MONITOR specification applies only to the connector initiating the IXLCONN request, not to the structure as a whole.

**Note:** To ensure that the IXLCONN MONITOR support is installed on the system on which you are running, issue IXCQUERY REQINFO=FEATURES. QuReqRfRepopulateProgress, returned from the IXCQUERY request, indicates whether the support is installed.

**DEFAULT**

Indicates that the system is to use the default monitoring algorithm associated with the current release or service level. The default algorithms are as follows:

- z/OS V1R12 without APAR OA40933: Connectors are allowed 2 minutes to complete repopulation. The system declares a hang after 2 minutes have elapsed.
- z/OS V1R13 and above, or z/OS V1R12 with APAR OA40933: Connectors are allowed up to 6 minutes to complete repopulation, as long as the rate of CF requests to the new structure instance exceeds a system-determined minimum value. The system declares a hang:
  - In the first monitoring interval where the required request rate is not observed, after at least 2 minutes have elapsed.
  - After 6 minutes have elapsed, regardless of request rate.

**CFREQRATE**

Indicates that the system is to use the rate of CF requests to the new structure as the sole indication of progress toward repopulation, without imposing a limit on the start-to-end duration. The system declares a hang in the first monitoring interval where the required request rate is not observed, after at least 2 minutes have elapsed. It will not declare a hang as long as the request rate exceeds a system-determined minimum value.

MONITOR=CFREQRATE is not appropriate for use by passive connectors that are simply waiting for another connector to complete structure repopulation and do not initiate CF requests of their own.



**IXLREBLD**

Indicates that the connector will initiate IXLREBLD REQUEST=POPULATING or WAITING requests to indicate that it is making satisfactory progress toward repopulation. The system will declare a hang:

- If the first 2-minute interval elapses without receipt of an IXLREBLD REQUEST=COMPLETE, POPULATING or WAITING.
- For a connector that has invoked IXLREBLD REQUEST=POPULATING, if a 2-minute interval elapses from the time of the last POPULATING request without receipt of an IXLREBLD REQUEST=COMPLETE, POPULATING, or WAITING.
- For a connector that has invoked IXLREBLD REQUEST=WAITING, if a 2-minute interval elapses after the failure of the master connector designated by the WAITFORCONTOKEN keyword without receipt of an IXLREBLD REQUEST=COMPLETE, POPULATING, or WAITING.

**VALUE**

Indicates that the MONITORVAL keyword specifies the method by which the system is to monitor repopulation progress.

**MONITORVAL=monitorval**

Use this input keyword to specify a value indicating the method by which the system is to monitor repopulation progress. The value provided must be equivalent to one of the IxlMonitorXxx constants defined in the IXLYCON macro.

**,NONVOLREQ=NO****,NONVOLREQ=YES**

Use this input parameter to indicate whether the connector's use of the structure requires that the coupling facility in which the structure is to be allocated must be both nonvolatile and failure-independent.

The nonvolatility attribute refers to the ability of certain coupling facilities to maintain their stored data should a power outage occur.

The failure-independent attribute refers to the configuration of a coupling facility in an independent failure domain from the MVS systems that access it. An independent failure domain ensures that there is not a single point of failure.

If there is not a coupling facility which is both nonvolatile and failure-independent, XES gives higher priority to allocating the structure in a nonvolatile coupling facility.

**NO**

Indicates that the system can allocate the structure in a coupling facility regardless of the volatility and failure-independent attributes of the coupling facility.

**YES**

Indicates that the system must attempt to allocate the structure in a coupling facility that is both nonvolatile and failure-independent. After allocation, the connected user must check the CONASTRUCTUREATTRFLAGS field in the connect answer area (IXLYCONA) to determine the allocated structure's nonvolatility and failure-independent attributes.

**,PLISTVER=IMPLIED\_VERSION****,PLISTVER=MAX****,PLISTVER=plistver**

Use this input parameter to specify the version of the macro. See [“Understanding IXLCONN version support”](#) on page 850 for a description of the options available with PLISTVER.

**,RATIO=YES****,RATIO=NO**

Use this input parameter to indicate whether this connection allows changes to the entry-to-element ratio and the event monitor control (EMC) storage percentage when a structure alter is initiated by an IXLALTER request.

**YES**

Indicates that this connection does allow changes to the entry-to-element ratio and the EMC storage percentage when a structure alter is initiated by an IXLALTER request. When specifying **RATIO=YES**, you must also specify **ALLOWALTER=YES** and **CFLEVEL=1** or higher.

**NO**

Indicates that this connection does not allow changes to the entry-to-element ratio and the EMC storage percentage when a structure alter is initiated by an IXLALTER request.

Specifying **RATIO=NO** does not mean that the entry to element ratio cannot change.

It may change as a result of a structure size change, or if the actual structure size (**CONASTRUCTURESIZE**) is less than the minimum control space (**CONAMINSTRUCTURESIZE**) required to allocate the structure with the attributes specified.

**,REBUILD**

Use this input parameter to specify that the requester is participating in a rebuilding process for the structure. Consider using **REQTYPE=REBUILDCONNECT** instead. See the description of **REQTYPE=REBUILDCONNECT** for more information.

You can specify the following IXLCONN parameters, but the system ignores any changes made to the values in effect from the original invocation of IXLCONN for the structure:

```
ALLOWALTER
ALLOWAUTO
ALLOWDUPREBLD
ALLOWREBUILD
ASYNXCI
CFLEVEL
CONDATA
CONDISP
CONLEVEL
```

```
CRITICAL
FUNCTION
MINELEMENT
MINEMC
MINENTRY
RATIO
STRDISP
SUSPEND
```

```
TERMLEVEL
EVENTEXIT
COMPLETEEXIT
LISTTRANEXIT (if required)
NOTIFYEXIT (if required)
CONEXIT (if required)
```

Once the new structure has been allocated, **REBUILD** requests from subsequent users connect the users to the new structure. As with any request to connect to a structure, the connectors must check the connect answer area to determine the actual attributes of the structure.

**REBUILD** requests must be issued from the same address space and from the same system as that from which the original IXLCONN macro for the structure was issued. **REBUILD** requests can be issued from a task other than the task that issued IXLCONN for the original structure.

**,REQTYPE=CONNECT****,REQTYPE=REBUILDCONNECT**

Use this input parameter to specify the type of connect request.

**CONNECT**

Can be used for normal (original) or rebuild connect requests but should only be used for normal connect requests. This is intended only to be used as the default and not actually specified on the IXLCONN invocation.

**REBUILDCONNECT**

Indicates a rebuild connect is requested. The rebuild process allows you to allocate another structure with the same name and to reconstruct data, if applicable, in the newly allocated

structure. The rebuild process is intended for planned reconfiguration and recovery scenarios and requires established protocols to ensure an orderly transition from the old to the new structure.

The requester must be connected to the original structure that is the target of the rebuilding process. If the rebuilding process for the structure has not been initiated (through the SETXCF START,REBUILD command or the IXLREBLD macro), the system rejects the rebuild connect request.

If rebuild has been initiated but the new structure has not been allocated, the rebuild connect request allocates the new structure. With this parameter, you must specify the same name (STRNAME) as that of the original structure. You can also modify the following IXLCONN parameters according to the structure TYPE:

For a cache structure:

ADJUNCT  
MAXELEMNUM  
VECTORLEN

ELEMENTRATIO  
UDFORDER  
ELEMCHAR

NUMSTGCLASS  
DIRRATIO<sup>1</sup>  
NUMCOCLASS

CONNECTIVITY  
NAMECLASSMASK

**Note:**

1. The parameter is ignored by a system with APAR OA33448 installed when KeepRatios=YES is specified.

For a list structure:

ADJUNCT<sup>1</sup>  
ENTRYRATIO<sup>1</sup>  
REFOPTION<sup>1</sup>

EMCSTGPCT<sup>1</sup>  
MAXELEMNUM<sup>1</sup>  
ELEMENTRATIO<sup>1</sup>

LOCKENTRIES  
ELEMCHAR<sup>1</sup>  
LISTHEADERS

CONNECTIVITY  
LISTCNTLTYPE<sup>1</sup>  
VECTORLEN

**Note:**

1. The parameter is ignored by a system with APAR OA33448 installed when KeepRatios=YES is specified.

For a lock structure:

CONNECTIVITY  
LOCKENTRIES  
NUMUSERS<sup>1</sup>  
PCTENTRYRSV

**Note:**

1. The value must be equal to or greater than the value for the original structure.

The following keywords will be IGNORED for rebuild connect requests:

- ASYNCXI
- CFLEVEL
- CONEXIT
- NOTIFYEXIT
- LISTTRANEXIT.

Note that some of these keywords are required, and you must therefore specify them for rebuild connect requests.

If the new structure for rebuild has not yet been allocated, then IXLCONN will allocate a new structure with the same name and using the attributes specified on connect. This connection will be connected to the new structure allocated for rebuild. As with any request to connect to a structure, the connectors must check the connect answer area to determine the actual attributes of the structure.

The rebuild connect request must be issued from the same address space and same system as the original IXLCONN which connected this user to the structure.

The rebuild connect request can be issued from a task other than the connecting task.

**,RETCODE=retcode**

Use this output parameter to specify the location in which the system is to copy the return code from GPR 15.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword to contain the return code.

**,RSNCODE=rsncode**

Use this output parameter to specify the location in which the system is to copy the reason code from GPR 0.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword to contain the reason code.

**,STRDISP=KEEP**

**,STRDISP=DELETE**

Use this input parameter to specify the persistence of the structure when there are no longer any connectors, either active or failed.

**KEEP**

Indicates that the structure remains allocated after all connectors have disconnected from the structure. You can delete a structure with this disposition with the IXLFORCE macro, the IXLDISC macro with the DELETESTR keyword, or the SETXCF FORCE operator command.

**DELETE**

Indicates that the structure is deleted from the coupling facility after all connectors (including failed-persistent connections) have disconnected from the structure.

**STRNAME=strname**

Use this input parameter to specify the name of the structure that you want to connect to in the coupling facility. The structure must be defined in the active CFRM policy at the time of the connect. You can use the IXCQUERY macro to determine which structures have been defined in the active CFRM policy.

The structure name must be 16 characters long, padded on the right with blanks, if necessary. The name can contain numeric characters, uppercase alphabetic characters, national characters (\$, @, #), or an underscore (\_). Structure names must begin with an uppercase alphabetic character. IBM structure names begin with SYS, an IBM component prefix, and the letter A through I.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the 16-character input field that contains the structure name.

**,STRSIZE=0****,STRSIZE=ysize**

Use this input parameter to specify the size of the structure in units of 4K blocks. The system allocates the structure following these guidelines:

- If the size specified is **smaller** than the minimum structure size that is specified or defaulted to in the active CFRM policy (MINSIZE), the system uses the MINSIZE value as the size of the structure.
- If the size specified is **smaller** than the structure size defined in the active CFRM policy (SIZE), the system uses the *ysize* value. The *ysize* is bounded by MINSIZE and SIZE in the active CFRM policy.
- If the size specified is **larger** than the structure size defined in the active CFRM policy, the system uses the size defined in the active policy (SIZE from policy).
- If you use the default value of zero for *ysize*, the system uses the size defined in the active CFRM policy (INITSIZE, if specified, or SIZE).

Note that in all cases, if there are limited coupling facility resources available, the system might allocate the structure using less space than was requested. MINSIZE specified in the active CFRM policy will serve as a minimum bound for the structure on all structure allocation requests (including alter, connect, and rebuild connect) except for new structure allocation during System-managed rebuild. New structure allocation during System-managed rebuild may cause the structure to be allocated with a size smaller than MINSIZE in the active CFRM policy. The actual structure size allocated is returned in field CONASTRUCTURESIZE in the IXLYCONA answer area. The requested structure size is returned in field CONAALLOCREQUESTEDSTRSIZE in IXLYCONA.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the fullword input field that contains the number of 4K blocks to make up the structure.

**,SUSPEND=YES****,SUSPEND=NO****,SUSPEND=FAIL**

Use this input parameter to indicate whether the connection wants the system to:

- Suspend work units that issue coupling facility requests against a structure while it is undergoing a system-managed process (for example, rebuild), regardless of the MODE specified on the coupling facility request.
- Process such requests as indicated by the MODE specification of the request.
- Fail such requests.

The SUSPEND keyword does not affect coupling facility requests that specify MODE=SYNCFail or MODE=SYNCSUSPEND.

**YES**

Indicates that the system will, when possible, override a request's MODE specification and suspend the requester while the structure is quiesced during a system-managed process. Specifying SUSPEND=YES permits the system to limit the number of incoming requests by quiescing the work units that would otherwise be submitting them. This minimizes the system resources required to quiesce activity against the structure undergoing the system-managed process.

When the requester is suspended as the result of specifying SUSPEND=YES, the duration of the suspend depends on the request:

- For cache (IXLCACHE) requests and unserialized list (IXLLIST, IXLLSTC, IXLLSTE, IXLLSTM) requests, the requester's unit of work will be resumed when the request is complete.
- For IXLLOCK and serialized list (IXLLIST, IXLLSTC, IXLLSTE) requests, the requester's unit of work will be resumed as soon as the quiesce for the system-managed process is complete. If the request subsequently encounters contention, control will be returned to the requester's unit of work with the appropriate return code to indicate that the request will be completed asynchronously.

When the requester is resumed after being suspended as the result of specifying SUSPEND=YES, the system will notify the requester of request completion as follows:

- For cache (IXLCACHE) requests, IXLLOCK requests that do not encounter contention, unserialized list (IXLLIST, IXLLSTC, IXLLSTE, IXLLSTM) requests, and serialized list (IXLLIST, IXLLSTC, IXLLSTE) requests that do not encounter contention:
  - If the MODE requested that the system attempt to complete the request synchronously (for example, SYNCEXIT), the requester will receive a return code indicating synchronous completion once the process completes.
  - If the MODE requested asynchronous processing (for example, ASYNCEXIT), the requester will receive a return code indicating asynchronous completion, and the requester will be notified of the results of the request through the specified mechanism (for example, the Complete exit).
- For IXLLOCK and serialized list (IXLLIST, IXLLSTC, IXLLSTE) requests that encounter contention and can tolerate asynchronous completion (for example, not SYNCSUSPEND):
  - The requester will receive the appropriate return code indicating asynchronous completion, and will be notified of the results of the request through the mechanism corresponding to the MODE specification once contention is resolved.

## NO

Indicates that the connector cannot tolerate suspension of units of work submitting coupling facility requests except as specified on the MODE keyword of list (IXLLIST, IXLLSTC, IXLLSTE, IXLLSTM), cache (IXLCACHE), and IXLLOCK requests, and will therefore use suspend/resume processing only on units of work that specify MODE=SYNCSUSPEND. The system will quiesce all other activity by deferring requests internally until the system-managed process completes.

## FAIL

Indicates that the connector cannot tolerate the potentially long-term suspension or delay of units of work submitting coupling facility requests while the structure is quiesced for system-managed processing. Requests that cannot be immediately processed due to the structure being quiesced for system-managed processing will be failed. Such requests will neither be deferred internally for asynchronous processing, nor will the requesting unit of work be suspended. If the structure is quiesced to fall out of a system-managed duplexing rebuild, which is only a short-term delay, then the request will be processed as if SUSPEND=NO had been specified. SUSPEND=FAIL is not applicable to lock or serialized list structures. Connect attempts to these types of structures will fail if SUSPEND=FAIL is specified.

### **,TERMLEVEL=TASK**

### **,TERMLEVEL=ADDRSPACE**

### **,TERMLEVEL=SYSTEM**

Specifies the first level at which the system is to take action if it becomes necessary to terminate this connector. The system may take such an action for the following purposes:

- To relieve a hang in a subsequent process that is related to the structure to which the invoker is connecting. The system will take automatic action if the hang is caused by failure of a connector to provide an expected response, and if the installation has specified a nonzero CFSTRHANGTIME value in the SFM policy. The TERMLEVEL keyword does not affect applicable hang relief actions that are not specific to a connector, such as stopping a rebuild.
- In response to an unexpected error while processing a coupling facility request initiated by the connector or during other processing associated with such a request.

**Note:** Depending on the nature of the problem, the system might take other actions before attempting to terminate the connector.

### **TERMLEVEL=TASK**

When termination of this connector is required, the system's first level of response will be to terminate the task from which the IXLCONN macro was issued. This is the default level, and should be used unless the application cannot tolerate task termination.

**TERMLEVEL=ADDRSPACE**

When termination of this connector is required, the system's first level of response will be to terminate the home address space from which the IXLCONN macro was issued. This level should be selected only if failure of the connector's task will leave sysplex resources in an unusable condition or for some other reason cannot be tolerated.

Task-level recovery is permitted to run when the connector's address space is terminated as a result of this selection.

**TERMLEVEL=SYSTEM**

When termination of this connector is required, the system's first level of response will be to partition the connector's system. This level should be selected only if failure of the connector's task or address space will leave sysplex resources in an unusable condition or for some other reason cannot be tolerated.

**,TYPE=CACHE****,TYPE=LIST****,TYPE=LOCK**

Use this input parameter to identify the type of structure in the coupling facility to which you want to connect. The parameter descriptions applicable to a cache structure follow. For parameter descriptions applicable to a list structure, see [“Parameters for TYPE=LIST” on page 874](#); for parameter descriptions applicable to a lock structure, see [“Parameters for TYPE=LOCK” on page 879](#).

## Parameters for TYPE=CACHE

**,ADJUNCT=NO****,ADJUNCT=YES**

Use this input parameter to specify whether adjunct data areas are associated with each cache entry in the coupling facility cache structure. Each adjunct data area is 64 bytes.

**NO**

Indicates that adjunct data areas should not be allocated for the cache structure.

**YES**

Indicates that adjunct data areas should be allocated for each entry in the cache structure.

**,ASYNCXI=0****,ASYNCXI=*asynxci***

Use this input parameter to indicate whether this connection prefers cross-invalidation operations against local caches to be processed synchronously or asynchronously to the completion of cache requests initiated by this connection.

The ASYNCXI keyword only applies for a cache structure. The ASYNCXI keyword is not allowed for structure types list or lock.

**0 (IxLConnAsyncXiNo)**

Specifies that cross-invalidation operations be processed synchronously to the completion of cache requests initiated by this connection.

**1 (IxLConnAsyncXiYes)**

Specifies that cross-invalidation operations be processed asynchronously to the completion of cache operation requests initiated by this connection when supported by the coupling facility where the cache structure is allocated. An ASYNCXI value of 1 is meaningful only when a cache structure is allocated in a CFLEVEL=23 or higher coupling facility.

Any value other than IxLConnAsyncXiNo (0) or IxLConnAsyncXiYes (1) will have the same behavior as specifying a value of 0 (IxLConnAsyncXiNo).

A connected user can invoke the IXLAXISN service to determine whether asynchronous cross-invalidations associated with its connection have completed.

**,DIRRATIO=1****,DIRRATIO=*dirratio***

Use this input parameter to specify a value to express the directory portion of the directory-to-element ratio of the coupling facility cache structure. The value of DIRRATIO must be greater than zero.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the 2-byte input field that contains a value to express the directory portion of the directory-to-element ratio of the coupling facility cache structure.

**,ELEMCHAR=0****,ELEMCHAR=*elemchar***

Use this input parameter to specify the value of an element characteristic used to determine the element size. The element size is calculated with the formula  $256 \times (2^{\text{ELEMCHAR}})$ , where ELEMCHAR is used as the power of 2. For example, if ELEMCHAR=0, then the size of each element is 256 bytes.

The valid values for ELEMCHAR range from zero to a maximum determined by the coupling facility limitation on element size.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the 1-byte input field that contains the value of the element characteristic, used to determine the cache structure element size.

**Note:** The element size, in combination with the maximum number of elements (MAXELEMNUM parameter), determines the size of the data entry — the data written to and read from the cache structure. With a coupling facility of CFLEVEL=0, a data entry can be up to 16 times the data element size, as indicated by the element characteristic. With a coupling facility of CFLEVEL=1 or higher, a data entry can be up to 255 times the data element size.

Use either ELEMCHAR or ELEMINCRNUM to define the element size. You can specify either an element characteristic or an element increment number for element size determination.

**,ELEMENTRATIO=1****,ELEMENTRATIO=*elementratio***

Use this input parameter to specify a value to express the element portion of the directory-to-element ratio of the coupling facility cache structure. If ELEMENTRATIO is zero, the structure is allocated without data elements.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the 2-byte input field that contains a value to express the element portion of the directory-to-element ratio of the coupling facility cache structure.

**,ELEMINCRNUM=*elemincrnum***

Use this input parameter to specify the element increment number. The element size is calculated with the formula  $256 \times \text{ELEMINCRNUM}$ . For example, if ELEMINCRNUM=1, then the size of each element is 256 bytes.

The valid values for ELEMINCRNUM range from 1 to a maximum determined by the coupling facility limitation on element size. The value of ELEMINCRNUM must be a power of 2.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the 1-byte input field that contains the value of the element increment number, used to determine the cache structure element size.

**Note:** The element size, in combination with the maximum number of elements (MAXELEMNUM parameter), determines the size of the data entry — the data written to and read from the cache structure. With a coupling facility of CFLEVEL=0, a data entry can be up to 16 times the data element size, as indicated by the element increment number. With a coupling facility of CFLEVEL=1 or higher, a data entry can be up to 255 times the data element size.

Use either ELEMCHAR or ELEMINCRNUM to define the element size. You can specify either an element characteristic or an element increment number for element size determination.



**,MAXELEMNUM=16****,MAXELEMNUM=***maxelemnum*

Use this input parameter to specify a value to determine the maximum number of data elements for each data entry in the coupling facility cache structure.

- With a coupling facility of CFLEVEL=0 or higher, you can specify from 1 to 16 for MAXELEMNUM.
- With a coupling facility of CFLEVEL=1 or higher, you can specify from 1 to 255 for MAXELEMNUM.

The maximum data entry size in bytes equals MAXELEMNUM multiplied by the element size obtained from the value specified for ELEMCHAR or ELEMINCRNUM.

For example, if ELEMCHAR=0 and MAXELEMNUM=1, the maximum data entry size in bytes is 256. If ELEMCHAR=4 and MAXELEMNUM=16, the maximum data entry size in bytes is 4096×16, or 65,536.

To ensure that the system can assign all possible data elements allocated in a structure to a directory entry, MAXELEMNUM must be greater than or equal to ELEMENTRATIO divided by DIRRATIO. MAXELEMNUM is ignored if ELEMENTRATIO is zero.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the 1-byte input field that contains the maximum number of data elements for each data entry in the coupling facility cache structure.

**,NAMECLASSMASK=0****,NAMECLASSMASK=***nameclassmask*

Use this input parameter to specify the name class mask pattern definition to be applied to entry names for the purpose of assigning entries to name classes in the structure. Name classes may be used to improve the efficiency of processing for commands such as IXLCACHE REQUEST=DELETE\_NAME.

The position of each bit in the name class mask corresponds to the same relative character position in an entry name. If a bit in the name class mask is a one, then the corresponding position in the entry name is applied in the masking operation to determine the name class to which the entry will be assigned. If a bit is a zero, then the corresponding position in the entry name is not applied to assigning the entry to a name class.

Specifying or defaulting to a name class mask of 0 (X'0000'), or specifying a name class mask of all binary ones (X'FFFF'), will result in name classes not being used for the structure. A CFLEVEL of 7 or greater must be specified when any other name class mask is specified, implying that name classes are to be used.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the 16-bit input field that contains the name class mask pattern definition.

**,NUMCOCLASS=numcoclclass**

Use this input parameter to specify the number of cast-out classes for named data entries used with the coupling facility cache structure. You must specify at least 1 cast-out class.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the fullword input field that contains the number of cast-out classes.

**,NUMSTGCLASS=numstgclass**

Use this input parameter to specify the number of storage classes for named data entries used with the coupling facility cache structure. You must specify at least 1 storage class.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the fullword input field that contains the number of storage classes.

**,SUPPRESSEVENTS=NO****,SUPPRESSEVENTS=YES**

Use this input parameter to indicate whether the origination of user connection events (New Connection, Existing Connection, Rebuild New Connection, Rebuild Existing Connection), and user disconnection/failure events (Disconnected or Failed Connection), should be suppressed for this connection. Suppression of these events may provide a significant performance benefit at connect or disconnect time to connectors who do not need the information presented in these events.

Note that suppression of these events is on a per-connection basis, where it is the ORIGINATION of the event that is suppressed, not the RECEIPT of the event. Therefore, the following cases are possible:

- A connection that is suppressing the origination of events may nevertheless receive such events for another connection to the structure that is NOT suppressing the origination of events. In the case of response-required events, this connection is still required to provide responses to any such event that is presented to it, regardless of whether it is suppressing events itself.
- When a connection is suppressing the origination of events, other connections will not receive the suppressed events for that connection, even if the other connections themselves are NOT suppressing events.

Also note that, depending on whether the system on which the connection is running is at OS/390 Release 9 or higher or has OW38840 installed, a request to suppress events may or may not be honored for the current connection.

#### **NO**

Indicates that connection and disconnection related events will be originated normally for this connection.

#### **YES**

Indicates that the origination of connection and disconnection related events may be suppressed for this connection. (Dependent on the level of the system on which the connection is running.)

#### **,UDFORDER=NO**

#### **,UDFORDER=YES**

Use this input parameter to indicate whether a user data field (UDF) order queue should be maintained for each cast-out class for the structure. A coupling facility with CFLEVEL=5 or higher is required when UDFORDER=YES.

Depending on whether UDF-order queues are maintained, a READ\_COSTATS request to retrieve cast-out class statistics will include the following information:

- If a UDF-order queue is maintained, the system will return the user data field of the first entry on the UDF-order queue.
- If a UDF-order queue is not maintained, the system will return the user data field of the first entry on the cast-out class queue.

#### **,VECTORLEN=vectorlen**

Use this input parameter to specify the number of cache buffers in the requestor's local storage which require concurrent registration. The requestor uses the vector length to map each local cache buffer to a named data entry in the coupling facility cache structure. The value of VECTORLEN must be a multiple of 32 or the system will round the value up to a multiple of 32.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the fullword input field that contains the number of cache buffers in the requestor's local storage.

## **Parameters for TYPE=LIST**

#### **,ADJUNCT=NO**

#### **,ADJUNCT=YES**

Use this input parameter to specify whether adjunct data areas are associated with each list entry in the coupling facility list structure. Each adjunct data area is 64 bytes.

#### **NO**

Indicates that there are no adjunct data areas specified for the list structure.

#### **YES**

Indicates that each list entry in the list structure has an associated adjunct data area of 64 bytes.

**,ELEMCHAR=0****,ELEMCHAR=*elemchar***

Use this input parameter to specify the value of an element characteristic used to determine the element size. The element size is calculated with the formula  $256 \times (2^{\text{ELEMCHAR}})$ , where ELEMCHAR is used as the power of 2. For example, if ELEMCHAR=0, then the size of each element is 256 bytes.

The valid values for ELEMCHAR range from zero to a maximum determined by the coupling facility model limitation on element size.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the 1-byte field that contains the value of the element characteristic, used to determine the list structure element size.

**Note:** The element size, in combination with the maximum number of elements (MAXELEMNUM parameter), determines the size of the data entry — the data written to and read from the list structure. With a coupling facility of CFLEVEL=0, a data entry can be up to 16 times the data element size, as indicated by the element characteristic. With a coupling facility of CFLEVEL=1 or higher, a data entry can be up to 255 times the data element size.

Use either ELEMCHAR or ELEMINCRNUM to define the element size. You can specify either an element characteristic or an element increment number for element size determination.

**,ELEMENTRATIO=1****,ELEMENTRATIO=*elementratio***

Use this input parameter to specify a value to express the data element portion of the entry-to-element ratio of the coupling facility list structure. If ELEMENTRATIO is zero, the list structure is allocated without data elements. If ELEMENTRATIO is zero, then ADJUNCT must equal YES.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the 2-byte field that contains a value to express the element portion of the entry-to-element ratio of the coupling facility list structure.

**,ELEMINCRNUM=*elemincnum***

Use this input parameter to specify the element increment number. The element size is calculated with the formula  $256 \times \text{ELEMINCRNUM}$ . For example, if ELEMINCRNUM=1, then the size of each element is 256 bytes.

The valid values for ELEMINCRNUM range from 1 to a maximum determined by the coupling facility model limitation on element size. The value of ELEMINCRNUM must be a power of 2.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the 1-byte field that contains the value of the element increment number, used to determine the list structure element size.

**Note:** The element size, in combination with the maximum number of elements (MAXELEMNUM parameter), determines the size of the data entry — the data written to and read from the list structure. With a coupling facility of CFLEVEL=0, a data entry can be up to 16 times the data element size, as indicated by the element increment number. With a coupling facility of CFLEVEL=1 or higher, a data entry can be up to 255 times the data element size.

Use either ELEMCHAR or ELEMINCRNUM to define the element size. You can specify either an element characteristic or an element increment number for element size determination.

**,EMCSTGPCT=0****,EMCSTGPCT=*emcstgpct***

Use this input parameter to specify the percentage of available list structure storage that is to be set aside for event monitor controls (EMCs). The structure must be a keyed list structure (REFOPTION=KEY) and its allocation must be requested in a coupling facility with CFLEVEL=3 or higher.

Specify the percentage value in hundredths of a percent. For example, to request a value of 20%, code EMCSTGPCT=2000.

The value of EMCSTGPCT must be in the range of from 0 to 10000.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the halfword field that contains a value to express the percentage of available structure storage that is to be set aside for event monitor controls (EMCs).

**,ENTRYIDTYPE=SYSTEM**

**,ENTRYIDTYPE=USER**

Use this input parameter to specify whether the system or the user will assign the list entry IDs for list entries created in this structure. A list entry can be located by its assigned entry ID. A connect request will be rejected if the connector's requested ENTRYIDTYPE attribute does not match the attribute in effect for the currently allocated instance of the structure. A rebuild connect request will be rejected if the connector's requested ENTRYIDTYPE attribute does not match the attribute in effect for the original structure.

**ENTRYIDTYPE=SYSTEM**

The system will assign the list entry IDs for list entries created in this structure. An entry ID will be generated by the system for each list entry created, and it will be unique among the list entries previously and currently allocated in the list structure. A system generated entry ID is never reused and is unique for the life of the structure.

**ENTRYIDTYPE=USER**

The user will assign the entry ID for each list entry created in this structure. An entry ID must be provided by the user for each list entry created, and it must be unique among the list entries currently allocated in the list structure. A user-provided entry ID may be reused over time provided it is unique while assigned to a list entry.

A CFLEVEL of 8 or higher must also be specified when ENTRYIDTYPE=USER is specified. Note that an ENTRYIDTYPE=USER request will be rejected if a coupling facility of CFLEVEL=8 or higher is not available in which to allocate the structure.

**,ENTRYRATIO=1**

**,ENTRYRATIO=entryratio**

Use this input parameter to specify a value to express the list entry portion of the entry-to-element ratio of the coupling facility list structure.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the 2-byte field that contains a value to express the list entry portion of the entry-to-element ratio of the coupling facility list structure.

**,KEYTYPE=ENTRY**

**,KEYTYPE=SECONDARY**

Use this input parameter to specify whether only entry keys, or both entry keys and secondary keys may be used when processing or locating list entries, or when comparing the entry keys of list entries.

**KEYTYPE=ENTRY**

Only entry keys may be used when processing or locating list entries, or when comparing the entry keys of list entries.

Each list entry is assigned a 16-byte entry key when it is created. The user may provide the entry key, specify that the list key value be assigned, or allow the entry key value to be assigned a default value. The entry key may be reassigned when the entry is moved. Event monitoring is provided for keyed sublists.

See the IXLLSTC, IXLLSTE, and IXLLSTM macros for additional information.

**KEYTYPE=SECONDARY**

Both entry keys and secondary keys may be used when processing or locating list entries, or when comparing the entry keys of list entries.

Entry keys are stored in the list entry, whereas secondary keys are stored in the first 32 bytes of the 64-byte list entry adjunct area, thus restricting the available user space in the adjunct area to the second 32 bytes. Note also that the list structure must be allocated to use list entries with adjunct data. Thus, in addition to KEYTYPE=SECONDARY, ADJUNCT=YES must also be specified on the IXLCONN invocation when allocating the structure and when establishing a connection with the structure.

Each list entry is assigned a 16-byte entry key and a 32-byte secondary key when it is created. The user may provide the secondary key or allow the secondary key value to be assigned a default value. While the entry key can be reassigned by any of the move requests, the secondary key can be reassigned only by an IXLLSTM REQUEST=MOVE\_ENTRYLIST invocation. Event monitoring is provided for entry keyed and secondary keyed sublists.

See the IXLLSTC, IXLLSTC, and IXLLSTM macros for additional information.

ADJUNCT=YES and a CFLEVEL of 9 or greater must also be specified when KEYTYPE=SECONDARY is specified.

KEYTYPE=SECONDARY is meaningful only when the structure is allocated in a coupling facility of CFLEVEL=9 or higher.

**,LISTCNTLTTYPE=ENTRY**

**,LISTCNTLTTYPE=ELEMENT**

Use this input parameter to specify whether the system maintains and tracks list limits based on number of entries or number of elements.

**ENTRY**

Specifies that the list limits are specified and tracked as limits on the number of entries which may reside on the list.

**ELEMENT**

Specifies that the list limits are specified and tracked as limits on the total number of data elements which may be associated with entries on the list.

**,LISTHEADERS=listheaders**

Use this input parameter to specify the number of lists (list headers) for the coupling facility list structure. The number must be greater than zero.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the fullword input field that contains the number of list headers to be allocated.

**,LISTTRANEXIT=listtranexit**

Use this input parameter to identify the list transition exit for the requestor.

The list transition exit is used to inform users when one or more lists or their event queue they are monitoring changes from the empty state to the non-empty state.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the list transition exit for the requestor.

**,LOCKENTRIES=0**

**,LOCKENTRIES=lockentries**

Use this input parameter to specify the number of lock entries for the coupling facility list structure. If the value for the number of lock entries is not a power of 2, it is rounded upward to the nearest power of 2. If the value is not specified, or if the requestor specifies zero, then the system does not support serialization for the allocated list structure.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the fullword field that contains the number of lock entries for the coupling facility list structure.

**,MAXCONN=32**

**,MAXCONN=maxconn**

Use this input parameter to specify the maximum number of users that can connect and use the list structure. The actual number of connections that the structure supports is returned in the Connect Answer Area field CONACFACILITYUSERLIMIT.

When the requested MAXCONN attribute value for *maxconn* is in the range 0 to 32, the value is 32.

To allocate a list structure that can support more than 32 connectors, use the MAXCONN keyword as follows:

- Ensure that the MAXCONN keyword is used by all connectors to the structure.
- Ensure that the structure is allocated in a coupling facility with CFLEVEL=17 or higher.

Specifying the MAXCONN keyword with any value indicates that the connection can support a user-id limit change that results from a system-managed process (for example, rebuild). XCF communicates the user-id limit change through the structure state change event. Specifying the MAXCONN keyword with a value greater than 32 indicates that the connector can understand events for subject connections with connection identifiers up to the value specified (for example, EEPEXISTINGCONNECTION or EEPLNEWCONNECTION).

XCF rejects a MAXCONN connect request for the following reasons:

- If the requested MAXCONN attribute value of the connector is lower than the highest in-use connection id of an existing connector to the specified structure.
- If the available connection identifier of the connector is greater than the smallest NUMUSERS or MAXCONN value that existing connections have specified

XCF rejects a rebuild MAXCONN connect request if the requested MAXCONN attribute value of the connector is lower than the number of in-use connections to the old structure instance.

When you migrate applications to use greater than 32 connectors support in a rolling fashion, the application or installation can take the following steps to avoid failed connection attempts:

- Ensure that applications have internal structures coded to accept events for connection identifiers up to the planned value that is to be used on the MAXCONN keyword.
- Ensure that all instances of the application are updated to use the MAXCONN keyword. When all instances of the application have been upgraded to a level that specifies MAXCONN, the structure is eligible to accommodate more than 32 instances of the application.
- Ensure that the structure is reallocated in order to take advantage of greater than 32 connectors in a CFLEVEL 17 coupling facility. You can accomplish this through the rebuild process.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the 1-byte field that contains the maximum number of users of the coupling facility lock structure.

#### **,MAXELEMNUM=16**

#### **,MAXELEMNUM=*maxelemnum***

Use this input parameter to specify a value to determine the maximum number of elements for each list entry in the coupling facility list structure.

- With a coupling facility of CFLEVEL=0 or higher, you can specify from 1 to 16 for MAXELEMNUM.
- With a coupling facility of CFLEVEL=1 or higher, you can specify from 1 to 255 for MAXELEMNUM.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the 1-byte field that contains a value to determine the maximum number of elements for each list entry in the coupling facility list structure.

#### **,NOTIFYEXIT=*notifyexit***

Use this input parameter to identify the notify exit for the requestor.

The notify exit is used to inform a list services (IXLLIST, IXLLSTC, IXLLSTE, IXLLSTM) user that contention exists for a lock held by the user.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the notify exit for the requestor.

#### **,REFOPTION=NOKEYNAME**

#### **,REFOPTION=KEY**

#### **,REFOPTION=NAME**

Use this input parameter to specify how to reference list entries in the coupling facility list structure. The requestor can specify one of the following:

#### **NOKEYNAME**

Indicates that the list entry can be located by the list entry identifier (LEID). The system assigns an LEID for each list entry that is in use in the coupling facility list structure. Neither keys nor names are used to reference list entries.

**KEY**

Indicates that the list entry can be located by a key value. When creating the entry, the requestor can assign a key value to one or more list entries on the macro.

**NAME**

Indicates that the list entry can be located by a unique name. When creating the entry, the requestor can assign a unique name to each list entry on the list services (IXLLIST, IXLLSTC, IXLLSTE, IXLLSTM) macro.

**,VECTORLEN=0****,VECTORLEN=*vectorlen*.**

Use this input parameter to specify the number of vector entries in the vector that is to be used to monitor state transitions for list headers or the event queue. A list header or an event queue undergoes a state transition when it changes from an empty to nonempty state, or from a nonempty to an empty state.

The value of VECTORLEN must be a multiple of 32 or the system will round the value up to a multiple of 32.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the fullword field that contains the number of vector entries in the vector that is to be used to monitor state transitions for list headers or the event queue.

**Parameters for TYPE=LOCK****,CONTEXIT=*contextit***

Use this input parameter to identify the contention exit for the requestor.

The contention exit is used to resolve resource contention based on user-defined protocols.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the contention exit for the requestor.

**,LOCKENTRIES=*lockentries***

Use this input parameter to specify the number of lock entries to be allocated for the coupling facility lock structure. If the value for LOCKENTRIES is not a power of 2, it is rounded upward to the nearest power of 2. If a lock structure contains record data (RECORD=YES), you must specify a nonzero value for LOCKENTRIES. If a lock structure does not contain record data (RECORD=NO) and the value for LOCKENTRIES is zero, the system attempts to allocate the structure with as many lock entries as possible within the given structure size.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the fullword field that contains the number of lock entries to be allocated for the lock structure. If the lock structure supports record data, a value of zero for LOCKENTRIES is not valid. If the lock structure does not support record data, a value of zero indicates that the system should obtain the largest possible number of lock entries given the structure's allocated size.

**,MAXCONN=*maxconn*****,NUMUSERS=*numusers***

Use this input parameter to specify the maximum number of users that can connect and use the lock structure. You must use either NUMUSER or MAXCONN. The actual number of connections that the structure supports is returned in the Connect Answer Area field CONALOCKNUMUSERS.

For NUMUSERS=*numusers*, specify a value in the range of 1 to 32. If NUMUSERS specifies a value over 32, the structure will only allow 32 connections, causing the actual number of supported connections to be 32.

To allocate a structure that can support more than 32 connectors, use the MAXCONN keyword.

For NUMUSERS if the user limit for the allocated structure (returned in CONACFACILITYUSERLIMIT) is greater than 32, the system rejects the connect request for the following reasons:

- The requested NUMUSERS attribute value of the connector is lower than the number of in-use connections to the specified structure.

- The available connection identifier is greater than the smallest NUMUSERS or MAXCONN value that existing connections have specified.
- The available connection identifier is greater than 32.

If you have specified NUMUSERS on a connect request, XCF rejects the request for the following reasons:

- If the requested MAXCONN attribute value of the connector is lower than the highest in-use connection identifier to the specified structure.
- If the available connection identifier is greater than the smallest NUMUSERS or MAXCONN value that existing connections have specified.
- The available connection identifier is greater than 32.

XCF rejects a rebuild connect request for the following reasons:

- If the requested NUMUSERS attribute of the connector is lower than the number of in-use connections to the old structure instance.
- If the initial connect request specified the MAXCONN keyword. Use either MAXCONN or NUMUSERS to specify the maximum number of users that can connect to the lock structure.

To allocate a lock structure that can support more than 32 connectors, use the MAXCONN=*maxconn* keyword as follows:

- Ensure that the MAXCONN keyword is used by all connectors to the structure.
- Ensure that the structure is allocated in a coupling facility with CFLEVEL=17 or higher.

Specifying the MAXCONN keyword with any value indicates that the connection can support a user-id limit change that results from a system-managed process (for example, rebuild). XCF communicates the user-id limit change through the structure state change event. Specifying the MAXCONN keyword with a value greater than 32 indicates that the connector can understand events for subject connections with connection identifiers up to the value specified (for example, Eep1ExistingConnection or Eep1NewConnection).

XCF rejects a MAXCONN connect request for the following reasons:

- If the requested MAXCONN attribute value of the connector is lower than the highest in-use connection id of an existing connector to the specified structure.
- If the available connection identifier of the connector is greater than the smallest NUMUSERS or MAXCONN value that existing connections have specified

XCF rejects a rebuild MAXCONN connect request for the following reasons:

- If the requested MAXCONN attribute value of the connector is lower than the number of in-use connections to the old structure instance.
- If the initial connect request specified the NUMUSERS keyword.

When you migrate applications to use greater than 32 connectors support in a rolling fashion, the application or installation can take the following steps to avoid failed connection attempts:

- Ensure that applications have internal structures coded to accept events for connection identifiers up to the planned value that is to be used on the MAXCONN keyword.
- Ensure that all instances of the application are updated to use the MAXCONN keyword. When all instances of the application have been upgraded to a level that specifies MAXCONN, the structure is eligible to accommodate more than 32 instances of the application.
- Ensure that the structure is reallocated in order to take advantage of greater than 32 connectors in a CFLEVEL 17 coupling facility. You can accomplish this through the rebuild process.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the 1-byte field that contains the maximum number of users of the coupling facility lock structure.



**,MONITORSTORAGE=0****,MONITORSTORAGE=monitorstorage**

Use this input parameter to indicate whether monitoring of storage is desired. Valid values are 0 (or IXLCONNMONITORSTORAGENO) and 1 (or IXLCONNMONITORSTORAGEYES).

A value of 0 (IXLCONNMONITORSTORAGENO) specifies not to monitor storage usage. When storage is exceeded, the XES storage manager issues an abend of X'026'.

A value of 1 (IXLCONNMONITORSTORAGEYES) specifies to monitor storage usage. When the amount of in-use storage reaches a preestablished threshold, incoming requests will be rejected with a return code of IXLRETCODEENVERROR and a reason code of ISLRSNCODERESOURCESCONSTRAINED.

Any other specified value will have the same behavior as specifying a value of 0 (IXLCONNMONITORSTORAGENO).

**,NOTIFYEXIT=notifyexit**

Use this input parameter to specify the name of the notify exit for the requestor.

The notify exit is used to notify resource owners that contention for the resource exists.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the name of the notify exit for the requestor.

**,NUMUSERS=numusers or ,MAXCONN=maxconns**

Use this input parameter to specify the maximum number of users that can connect and use the lock structure. You must use either NUMUSER or MAXCONN. The actual number of connections that the structure supports is returned in the Connect Answer Area field ConaLockNumUsers.

For NUMUSERS=numusers, specify a value in the range of 1 to 32. If NUMUSERS specifies a value over 32, the structure will only allow 32 connections, causing the actual number of supported connections to be 32.

To allocate a structure that can support more than 32 connectors, use the MAXCONN keyword.

For NUMUSERS if the user limit for the allocated structure (returned in ConaCFacilityUserLimit) is greater than 32, the system rejects the connect request for the following reasons:

- The requested NUMUSERS attribute value of the connector is lower than the number of in-use connections to the specified structure.
- The available connection identifier is greater than the smallest NUMUSERS or MAXCONN value that existing connections have specified.
- The available connection identifier is greater than 32.

If you have specified NUMUSERS on a connect request, XCF rejects the request for the following reasons:

- If the requested MAXCONN attribute value of the connector is lower than the highest in-use connection identifier to the specified structure.
- If the available connection identifier is greater than the smallest NUMUSERS or MAXCONN value that existing connections have specified.
- The available connection identifier is greater than 32.

XCF rejects a rebuild connect request for the following reasons:

- If the requested NUMUSERS attribute of the connector is lower than the number of in-use connections to the old structure instance.
- If the initial connect request specified the MAXCONN keyword. Use either MAXCONN or NUMUSERS to specify the maximum number of users that can connect to the lock structure.

To allocate a lock structure that can support more than 32 connectors, use the MAXCONN=maxconns keyword as follows:

- Ensure that the MAXCONN keyword is used by all connectors to the structure.
- Ensure that the structure is allocated in a coupling facility with CFLEVEL=17 or higher.

Specifying the MAXCONN keyword with any value indicates that the connection can support a user-id limit change that results from a system-managed process (for example, rebuild). XCF communicates the user-id limit change through the structure state change event. Specifying the MAXCONN keyword with a value greater than 32 indicates that the connector can understand events for subject connections with connection identifiers up to the value specified (for example, Eep1ExistingConnection or Eep1NewConnection).

XCF rejects a MAXCONN connect request for the following reasons:

- If the requested MAXCONN attribute value of the connector is lower than the highest in-use connection id of an existing connector to the specified structure.
- If the available connection identifier of the connector is greater than the smallest NUMUSERS or MAXCONN value that existing connections have specified

XCF rejects a rebuild MAXCONN connect request for the following reasons:

- If the requested MAXCONN attribute value of the connector is lower than the number of in-use connections to the old structure instance.
- If the initial connect request specified the NUMUSERS keyword.

When you migrate applications to use greater-than-32 connectors support in a rolling fashion, the application or installation can take the following steps to avoid failed connection attempts:

- Ensure that applications have internal structures coded to accept events for connection identifiers up to the planned value that is to be used on the MAXCONN keyword.
- Ensure that all instances of the application are updated to use the MAXCONN keyword. When all instances of the application have been upgraded to a level that specifies MAXCONN, the structure is eligible to accommodate more than 32 instances of the application.
- Ensure that the structure is reallocated in order to take advantage of greater-than-32 connectors in a CFLEVEL 17 coupling facility. You can accomplish this through the rebuild process.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the 1-byte field that contains the maximum number of users of the coupling facility lock structure.

**,PCTENTRYRSV=0**

**,PCTENTRYRSV=*pctentryrsv***

Use this input parameter to specify the percentage of the total number of record data entries in the structure that must remain free after completion of a request that attempts to create a record data entry.

When the composite percent entry reserved value for the structure is non-zero, a request that would create an entry will fail with a return code of IxlRetCodeEnvError, reason code of IxlRsnCodeRtFull if the resulting percentage of free entries would be less than the percent entry reserved value in effect for the structure unless the request expressly overrides the PCTENTRYRSV threshold enforcement.

The percentage of free entries is arrived at by dividing the resulting number of free entries by the total number of entries then multiplying by 100.

The composite percent entry reserved value for the structure is the most restrictive (highest) PCTENTRYRSV value specified by all active connectors to the structure.

Use this keyword to ensure that sufficient free entries remain available to support application and connector recovery scenarios. Valid values for PCTENTRYRSV are from 0 to 99. A value of zero implies that no record data entries are required to remain free after completion of a request.

When needed, an application may indicate that a lock request that attempts to create a record data entry may proceed with creating a record data entry even if the resulting percentage of free entries at the completion of the request would be less than the established percent entry reserved threshold. For more information see the IXLLOCK and IXLRT services.

The PCTENTRYRSV parameter is meaningful only when the lock structure provides recording capabilities (RECORD=YES), the structure is allocated in a CFLEVEL=25 or higher coupling facility and the z/OS system from which the IXLCONN is issued supports the PCTENTRYRSV keyword.

To determine whether the support for the PCTENTRYRSV keyword is available on the system from which you are connecting, issue IXCQUERY REQINFO=FEATURES. QuReqRfPctEntryRsv indicates whether the support is available. If the support is not available and you connect specifying PCTENTRYRSV, the parameter will be ignored.

**,RECORD=NO**

**,RECORD=YES**

Use this input parameter to specify whether the lock user wants to do recording of data about the lock:

**NO**

Indicates the system does not record data.

**YES**

Indicates the system records data.

**,RNAMELEN=FIXED64**

**,RNAMELEN=VAR300**

Use this input parameter to indicate the length attribute of the resource names (either fixed length or variable length). All connectors to the structure must specify the same attribute when they connect to the structure. The system rejects a connect request if the connector's requested RNAMELEN value does not match the attribute in effect for a currently-allocated instance of the structure, as determined by the IXLCONN RNAMELEN specification of the first connector to the structure. The system also rejects a rebuild connect request if RNAMELEN is specified and its value is not consistent with the value specified for the original structure.

**RNAMELEN=FIXED64**

Specifies that the resource names for the structure will have a fixed length of 64 bytes.

**RNAMELEN=VAR300**

Specifies that the resource names for the structure will have a variable length between 1 and 300 bytes.

## Return and reason codes for the IXLCONN macro

---

When the IXLCONN macro returns control to your program:

- GPR 15 (and *retcode*, if you coded RETCODE) contains a return code.
- GPR 0 (and *rsncode*, if you coded RSNCODE) contains a reason code, if applicable.

Macro IXLYCON provides equate symbols for the return and reason codes. The equate symbols that are associated with each hexadecimal return code are as follows:

**0**

IXLRETCODEOK

**4**

IXLRETCODEWARNING

**8**

IXLRETCODEPARMERROR

**C**

IXLRETCODEENVERROR

**10**

IXLRETCODECOMPERROR

The following table identifies the hexadecimal return and reason codes and the equate symbol that is associated with each reason code. xxxx indicates internal information.

Table 48. Return and reason codes for the IXLCONN macro

Hexadecimal return code	Hexadecimal reason code	Equate symbol meaning and action
0	None.	<p><b>Meaning:</b> Processing completed successfully. The system returns data to ANSAREA. To map ANSAREA, use IXLYCONA.</p> <p><b>Action:</b> None. It is the responsibility of the connector to verify that the structure attributes, as recorded in the connect answer area, are acceptable.</p>
4	xxxx0407	<p><b>Equate Symbol:</b> IXLRSNCODESPECIALCONN</p> <p><b>Meaning:</b> The connection was completed. The CONA provides additional status information about the structure and the connector. The field CONAFLAGS indicates any special conditions that affect the connection to the structure. If any of the CONAFLAGS bits are ON, the connector receives this reason code. CONAFLAGS contains flags that indicate one or more of the following: connector has been reconnected, rebuild in progress, rebuild stop in progress, alter in progress, or a user sync point event is set.</p> <p><b>Action:</b> If required, interrogate the CONAFLAGS field to determine the status of the structure.</p> <p>It is the responsibility of the connector to verify that the structure attributes, as recorded in the connect answer area, are acceptable.</p>
8	xxxx0801	<p><b>Equate Symbol:</b> IXLRSNCODEBADPARMLIST</p> <p><b>Meaning:</b> Program error. The IXLCONN parameter list is not accessible.</p> <p><b>Action:</b> Verify that</p> <ul style="list-style-type: none"> <li>• The parameter list address is uncorrupted.</li> <li>• The parameter list is addressable in the caller's primary address space.</li> <li>• If you are calling IXLCONN in AR-mode and you specified the parameter list address using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If you are calling IXLCONN in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLCONN.</li> </ul>
8	xxxx0802	<p><b>Equate Symbol:</b> IXLRSNCODEBADPARMLISTALET</p> <p><b>Meaning:</b> Program error. The IXLCONN parameter list ALET is not valid. The request fails.</p> <p><b>Action:</b> Ensure that the ALET is zero or that the ALET represents a valid entry on the DU-AL.</p>
8	xxxx0804	<p><b>Equate Symbol:</b> IXLRSNCODEBADVERSION#</p> <p><b>Meaning:</b> Program error. Invalid version number in the IXLCONN parameter list.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that your program did not overlay the parameter list storage.</li> <li>• Verify that your program was assembled with the correct macro library for the release of MVS your program is running on.</li> </ul>

Table 48. Return and reason codes for the IXLCONN macro (continued)		
Hexadecimal return code	Hexadecimal reason code	Equate symbol meaning and action
8	xxxx0806	<b>Equate Symbol:</b> IXLRSNCODESRBMODE <b>Meaning:</b> Program error. The requester is in SRB mode. <b>Action:</b> Do not issue the IXLCONN macro when running in SRB mode.
8	xxxx0807	<b>Equate Symbol:</b> IXLRSNCODENOTENABLED <b>Meaning:</b> Program error. The requester is not in an enabled state. <b>Action:</b> Issue the IXLCONN macro while running enabled for I/O and external interrupts.
8	xxxx0808	<b>Equate Symbol:</b> IXLRSNCODEMASTERAS <b>Meaning:</b> Program error. The connection is not permitted from the Master address space. <b>Action:</b> Do not issue the IXLCONN macro from the Master address space.
8	xxxx0809	<b>Equate Symbol:</b> IXLRSNCODEPRIMARYNOTHOME <b>Meaning:</b> Program error. The primary address space does not equal the home address space. <b>Action:</b> Make sure that the primary address space and the home address space are the same at the time of the IXLCONN invocation.
8	xxxx080D	<b>Equate Symbol:</b> IXLRSNCODEAREATOOSMALL <b>Meaning:</b> Program error. ANSAREA is too small, as specified by ANSLEN. <b>Action:</b> Ensure that the ANSLEN parameter properly reflects the size of the area that is specified by ANSAREA. Also, make sure that the length of ANSAREA is large enough to contain the returned data.
8	xxxx080E	<b>Equate Symbol:</b> IXLRSNCODEBADAREA <b>Meaning:</b> Program error. Unable to access ANSAREA. <b>Action:</b> Make sure that ANSAREA is properly addressed prior to the IXLCONN invocation.
8	xxxx080F	<b>Equate Symbol:</b> IXLRSNCODEBADAREALET <b>Meaning:</b> Program error. The ANSAREA ALET is not valid. <b>Action:</b> Verify that the ANSAREA ALET is valid.
8	xxxx081B	<b>Equate Symbol:</b> IXLRSNCODENOLENTRIES <b>Meaning:</b> Program error. The number of lock entries that are specified for a lock structure with record data was zero. <b>Action:</b> Ensure that if a lock structure is to support record data, the value of LOCKENTRIES is greater than zero.

Table 48. Return and reason codes for the IXLCONN macro (continued)		
Hexadecimal return code	Hexadecimal reason code	Equate symbol meaning and action
8	xxxx081C	<p><b>Equate Symbol:</b> IXLRSNCODENOLISTHDRS</p> <p><b>Meaning:</b> Program error. The number of list headers that are specified for LISTHEADERS must be greater than zero.</p> <p><b>Action:</b> Ensure that the IXLCONN LISTHEADERS parameter is a value greater than zero.</p>
8	xxxx081D	<p><b>Equate Symbol:</b> IXLRSNCODEZEROLUSERS</p> <p><b>Meaning:</b> Program error. The number of users that are specified for TYPE=LOCK NUMUSERS must be greater than zero.</p> <p><b>Action:</b> Ensure that the IXLCONN NUMUSERS parameter value is greater than zero.</p>
8	xxxx081F	<p><b>Equate Symbol:</b> IXLRSNCODECONNAME</p> <p><b>Meaning:</b> Program error. CONNAME matches the CONNAME of another active connection to the same structure. CONNAME must be unique for each connection to a structure.</p> <p><b>Action:</b> Make sure that the CONNAME specified is not already connected to the structure.</p>
8	xxxx0820	<p><b>Equate Symbol:</b> IXLRSNCODESTRTYPE</p> <p><b>Meaning:</b> Program error. The structure type that is specified does not match the type of the currently-allocated structure or the RNAMELEN attribute specified does not match that of the currently allocated structure. When you connect to an allocated structure or issue a rebuild connect to a structure, you cannot change some of the structure attributes, specifically TYPE and RNAMELEN. The request fails.</p> <p><b>Action:</b> Specify a value for TYPE or for RNAMELEN that matches the attribute specified when the structure was allocated.</p>
8	xxxx0821	<p><b>Equate Symbol:</b> IXLRSNCODESTRSERIAL</p> <p><b>Meaning:</b> Program error. The serialization attribute for a list structure that is specified by the LOCKENTRIES keyword on connect does not match the currently-allocated structure. When you connect to an allocated structure, you cannot change the structure attributes.</p> <p><b>Action:</b> Specify a value for LOCKENTRIES that matches that specified for original structure.</p>
8	xxxx0823	<p><b>Equate Symbol:</b> IXLRSNCODECONNAMEERR</p> <p><b>Meaning:</b> Program error. CONNAME is not valid.</p> <p><b>Action:</b> Verify that the CONNAME meets the requirements of the keyword.</p>

Table 48. Return and reason codes for the IXLCONN macro (continued)		
Hexadecimal return code	Hexadecimal reason code	Equate symbol meaning and action
8	xxxx084C	<p><b>Equate Symbol:</b> IXLRSNCODENOSAFAUTH</p> <p><b>Meaning:</b> Program error. Connector does not have the proper SAF authorization. In IXLYCONA, CONADIAG1 contains the SAF return code and CONADIAG2 contains the SAF reason code.</p> <p><b>Action:</b> Determine why the installation has not allowed the proper SAF authorization for access to the structure.</p>
8	xxxx0857	<p><b>Equate Symbol:</b> IXLRSNCODEBADMAXCONN</p> <p><b>Meaning:</b> Program error.</p> <p>For a lock structure the same keyword (MAXCONN or NUMUSERS) that was used on the initial connect was not used on the rebuild connect request.</p> <p>For a list structure the use of keyword MAXCONN is inconsistent between the initial IXLCONN request and the IXLCONN REBUILD request. If a connector to a list structure explicitly specifies a MAXCONN value on the initial IXLCONN request, the connector must explicitly specify a MAXCONN value on the IXLCONN REBUILD request. The actual specified value can be different from the value that is coded on the initial IXLCONN request. If a connector takes the default for MAXCONN on the initial IXLCONN request, the connector must take the default for MAXCONN on the IXLCONN REBUILD request.</p> <p><b>Action:</b> Specify the same keyword on the initial connect and the rebuild connect.</p>
8	xxxx085B	<p><b>Equate Symbol:</b> IXLRSNCODERECDLISTATTR</p> <p><b>Meaning:</b> Program error. The record data attribute for the structure being rebuilt is not the same as the record data for the original structure.</p> <p><b>Action:</b> Specify a value for RECORD on the IXLCONN REBUILD that matches that specified for the original structure.</p>
8	xxxx085C	<p><b>Equate Symbol:</b> IXLRSNCODEINVALIDLISTATTR</p> <p><b>Meaning:</b> Program error. A list structure must be allocated with one of the following: lock entries, data elements, or adjunct data. None were specified.</p> <p><b>Action:</b> Specify the request with at least one of the following: lock entries, data elements, or adjunct data.</p>
8	xxxx085D	<p><b>Equate Symbol:</b> IXLRSNCODEINVALIDSTGCLASS</p> <p><b>Meaning:</b> Program error. NUMSTGCLASS for a cache structure cannot equal zero.</p> <p><b>Action:</b> Specify a value for NUMSTGCLASS that is greater than zero.</p>

Table 48. Return and reason codes for the IXLCONN macro (continued)		
Hexadecimal return code	Hexadecimal reason code	Equate symbol meaning and action
8	xxxx085E	<p><b>Equate Symbol:</b> IXLRSNCODEINVALIDCOCLASS</p> <p><b>Meaning:</b> Program error. NUMCOCLASS for a cache structure cannot equal zero.</p> <p><b>Action:</b> Specify a value for NUMCOCLASS that is greater than zero.</p>
8	xxxx085F	<p><b>Equate Symbol:</b> IXLRSNCODEINVALIDVECTORLEN</p> <p><b>Meaning:</b> Program error. VECTORLEN for a cache structure cannot equal zero. The request fails.</p> <p><b>Action:</b> Specify a value for VECTORLEN that is nonzero for a cache structure.</p>
8	xxxx0860	<p><b>Equate Symbol:</b> IXLRSNCODEDIRRATIO</p> <p><b>Meaning:</b> Program error. DIRRATIO for a cache structure cannot equal zero. Directory entries are required for a cache structure.</p> <p><b>Action:</b> Specify a value for DIRRATIO that is greater than zero.</p>
8	xxxx0861	<p><b>Equate Symbol:</b> IXLRSNCODEENTRYRATIO</p> <p><b>Meaning:</b> Program error. When ELEMENTRATIO is greater than zero, ENTRYRATIO must be greater than zero. List entry controls are required for a list structure.</p> <p><b>Action:</b> Specify a value for ENTRYRATIO that is greater than zero.</p>
8	xxxx0862	<p><b>Equate Symbol:</b> IXLRSNCODEMAXELEMNUM</p> <p><b>Meaning:</b> Program error.</p> <ul style="list-style-type: none"> <li>• LIST Structure: If ELEMENTRATIO is not zero, then MAXELEMNUM must be greater than or equal to ELEMENTRATIO divided by ENTRYRATIO when allocating a list structure.</li> <li>• CACHE Structure: If ELEMENTRATIO is not zero, the MAXELEMNUM for must be greater than or equal to ELEMENTRATIO divided by DIRRATIO when allocating a cache structure.</li> </ul> <p><b>Action:</b> Modify the value of MAXELEMNUM to meet the required conditions.</p>
8	xxxx0863	<p><b>Equate Symbol:</b> IXLRSNCODETASKTERM</p> <p><b>Meaning:</b> Program error. The IXLCONN request is rejected because the requesting task is going through task termination. IXLCONN cannot be issued from a resource manager.</p> <p><b>Action:</b> Examine your protocol to ensure that the IXLCONN macro is not issued from a resource manager.</p>
8	xxxx086B	<p><b>Equate Symbol:</b> IXLRSNCODEELEMINCRNUM</p> <p><b>Meaning:</b> Program error. The value that is specified for ELEMINCRNUM is not valid. It must be nonzero and be a power of 2.</p> <p><b>Action:</b> Modify the value of ELEMINCRNUM to meet the required conditions.</p>



Table 48. Return and reason codes for the IXLCONN macro (continued)		
Hexadecimal return code	Hexadecimal reason code	Equate symbol meaning and action
8	xxxx086F	<p><b>Equate Symbol:</b> IXLRSNCODEINVALIDCFLEVEL</p> <p><b>Meaning:</b> Program error. The request to alter a structure is rejected because ALLOWALTER=YES was specified and a CFLEVEL of zero was either specified or taken as a default. A CFLEVEL of 1 or higher is required when ALLOWALTER=YES is specified.</p> <p><b>Action:</b> Ensure that CFLEVEL=1 is specified.</p>
8	xxxx0870	<p><b>Equate Symbol:</b> IXLRSNCODEREBUILDVECTORLEN</p> <p><b>Meaning:</b> Program error. The VECTORLEN value that is specified for the rebuild structure is not consistent with the VECTORLEN attribute of the original structure.</p> <p><b>Action:</b> If you specified VECTORLEN on the IXLCONN request for the original structure, then you must also specify it on the IXLCONN request for the rebuild structure. Or, if you did not specify VECTORLEN on the IXLCONN request for the original structure, then you must not specify it on the IXLCONN request for the rebuild structure.</p>
8	xxxx0871	<p><b>Equate Symbol:</b> IXLRSNCODEMAXELEMNUMELEMCHAR</p> <p><b>Meaning:</b> Program error. The values that are specified in MAXELEMNUM and ELEMCHAR would result in the structure having greater than 64K entries.</p> <p><b>Action:</b> Modify the value(s) of MAXELEMNUM and/or ELEMCHAR to meet the 64K limit.</p>
8	xxxx0872	<p><b>Equate Symbol:</b> IXLRSNCODEMINENTRY</p> <p><b>Meaning:</b> Program error. The value that is specified for MINENTRY is not valid.</p> <p><b>Action:</b> Modify the value of MINENTRY to be within the range of from 0 to 100.</p>
8	xxxx0873	<p><b>Equate Symbol:</b> IXLRSNCODEMINELEMENT</p> <p><b>Meaning:</b> Program error. The value that is specified in MINELEMENT is not valid.</p> <p><b>Action:</b> Modify the value of MINELEMENT to be within the range of from 0 to 100.</p>
8	xxxx087E	<p><b>Equate Symbol:</b> IXLRSNCODEBADFUNCTION</p> <p><b>Meaning:</b> Program error. The value that is specified in the FUNCTION keyword is not valid (bad character, leading blanks, all blank).</p> <p><b>Action:</b> Correct the value of FUNCTION.</p>

Table 48. Return and reason codes for the IXLCONN macro (continued)

Hexadecimal return code	Hexadecimal reason code	Equate symbol meaning and action																		
8	xxxx0881	<p><b>Equate Symbol:</b> IXLRSNCODEBADCFLEVEL</p> <p><b>Meaning:</b> Program error. The request is rejected because the specified parameter is not appropriate for the value specified for CFLEVEL. The specified reason is indicated in the following table. The reason code index is returned in the CONADIAG2 field of the connect answer area mapped by IXLYCONA.</p> <table> <tr> <th>Rsn Code Index</th><th>Specified Parameter</th><th>Minimum CFLEVEL</th></tr> <tr> <td>1</td><td>EMCSTGPCT&gt;0</td><td>3</td></tr> <tr> <td>2</td><td>UDFORDER=YES</td><td>5</td></tr> <tr> <td>3</td><td>NAMECLASSMASK</td><td>7</td></tr> <tr> <td>4</td><td>ENTRYIDTYPE=USER</td><td>8</td></tr> <tr> <td>5</td><td>KEYTYPE=SECONDARY</td><td>9</td></tr> </table> <p><b>Action:</b> Correct the parameter value in error.</p>	Rsn Code Index	Specified Parameter	Minimum CFLEVEL	1	EMCSTGPCT>0	3	2	UDFORDER=YES	5	3	NAMECLASSMASK	7	4	ENTRYIDTYPE=USER	8	5	KEYTYPE=SECONDARY	9
Rsn Code Index	Specified Parameter	Minimum CFLEVEL																		
1	EMCSTGPCT>0	3																		
2	UDFORDER=YES	5																		
3	NAMECLASSMASK	7																		
4	ENTRYIDTYPE=USER	8																		
5	KEYTYPE=SECONDARY	9																		
8	xxxx0882	<p><b>Equate Symbol:</b> IXLRSNCODEBADREFOPTION</p> <p><b>Meaning:</b> Program error. The request is rejected because the specified parameter is not appropriate for the value specified for REFOPTION.</p> <p><b>For PARAMETER REFOPTION must be EMCSTGPCT&gt;0 KEY</b></p> <p><b>Action:</b> Correct the parameter value in error.</p>																		
8	xxxx0883	<p><b>Equate Symbol:</b> IXLRSNCODEBADEMSTGPCT</p> <p><b>Meaning:</b> Program error. The value that is specified for EMCSTGPCT is not valid. The value must be between 0 and 10000.</p> <p><b>Action:</b> Correct the value of EMCSTGPCT.</p>																		
8	xxxx0884	<p><b>Equate Symbol:</b> IXLRSNCODEBADMINEMC</p> <p><b>Meaning:</b> Program error. The value that is specified for MINEMC is not valid. The value must be between 0 and 100.</p> <p><b>Action:</b> Correct the value of MINEMC.</p>																		
8	xxxx088D	<p><b>Equate Symbol:</b> IXLRSNCODEBADENTRYIDTYPE</p> <p><b>Meaning:</b> Program error. The requested ENTRYIDTYPE is not consistent with the ENTRYIDTYPE attribute of the allocated structure. Either ENTRYIDTYPE=USER was requested but the structure was previously allocated with ENTRYIDTYPE=SYSTEM, or ENTRYIDTYPE=SYSTEM was requested and the structure was previously allocated with ENTRYIDTYPE=USER.</p> <p><b>Action:</b> Correct the value of ENTRYIDTYPE.</p>																		

Table 48. Return and reason codes for the IXLCONN macro (continued)																				
Hexadecimal return code	Hexadecimal reason code	Equate symbol meaning and action																		
8	xxxx088E	<p><b>Equate Symbol:</b> IXLRSNCODEINCONSISTENTPARM</p> <p><b>Meaning:</b> Program error. A keyword specification was made that also requires one or more other keywords to be specified. The specific reason is indicated in the following table. The reason code index is returned in the CONADIAG2 field of the connect answer area that is mapped by IXLYCONA.</p> <table border="1"> <thead> <tr> <th>Rsn Code Index</th><th>Specified Parameter</th><th>Also Required</th></tr> </thead> <tbody> <tr> <td>1</td><td>KEYTYPE=SECONDARY</td><td>ADJUNCT=YES</td></tr> <tr> <td>2</td><td>LISTCNTLTTYPE=ELEMENT</td><td>ELEMENTRATIO&gt;0</td></tr> <tr> <td>3</td><td>KEEPRATIO</td><td>TYPE&lt;&gt;LOCK</td></tr> <tr> <td>4</td><td>KEEPRATIO</td><td>ALLOC&lt;&gt;NO</td></tr> <tr> <td>8</td><td>ASYNCDUPLEX=YES</td><td>TYPE=LOCK</td></tr> </tbody> </table> <p><b>Action:</b> Correct the parameter value in error.</p>	Rsn Code Index	Specified Parameter	Also Required	1	KEYTYPE=SECONDARY	ADJUNCT=YES	2	LISTCNTLTTYPE=ELEMENT	ELEMENTRATIO>0	3	KEEPRATIO	TYPE<>LOCK	4	KEEPRATIO	ALLOC<>NO	8	ASYNCDUPLEX=YES	TYPE=LOCK
Rsn Code Index	Specified Parameter	Also Required																		
1	KEYTYPE=SECONDARY	ADJUNCT=YES																		
2	LISTCNTLTTYPE=ELEMENT	ELEMENTRATIO>0																		
3	KEEPRATIO	TYPE<>LOCK																		
4	KEEPRATIO	ALLOC<>NO																		
8	ASYNCDUPLEX=YES	TYPE=LOCK																		
8	xxxx089E	<p><b>Equate Symbol:</b> IXLRSNCODEBADMINCFLEVEL</p> <p><b>Meaning:</b> Program error. Request rejected. The specified MINCFLEVEL value is greater than the specified CFLEVEL value.</p> <p><b>Action:</b> Correct the value for MINCFLEVEL or CFLEVEL as appropriate.</p>																		
8	xxxx089F	<p><b>Equate Symbol:</b> IXLRSNCODEPCTENTRYRSV</p> <p><b>Meaning:</b> Program error. The value that is specified in PCTENTRYRSV is not valid.</p> <p><b>Action:</b> Modify the value of PCTENTRYRSV to be within the range of 0 to 99.</p>																		
8	xxxx08A9	<p><b>Equate Symbol:</b> IXLRSNCODEBADSPENDOPTION</p> <p><b>Meaning:</b> Program error. Request rejected. SUSPEND=FAIL is not valid for lock or serialized list structures.</p> <p><b>Action:</b> Correct the value either of the SUSPEND parameter.</p>																		
8	xxxx08B5	<p><b>Equate Symbol:</b> IXLRSNCODEBADMONITORVAL</p> <p><b>Meaning:</b> Program error. The value specified by MONITORVAL is not valid.</p> <p><b>Action:</b> Specify a value equivalent to a valid value of MONITOR.</p>																		
C	xxxx0C02	<p><b>Equate Symbol:</b> IXLRSNCODENOMORECONNS</p> <p><b>Meaning:</b> Environmental error. The structure has the maximum number of permitted connections.</p> <p><b>Action:</b> Retry the request at a later time. The ENF signal 35 will be signaled when coupling facility resources become available.</p> <p>The maximum number of permitted connections may be limited by the value of PLEXCFG.</p>																		

Table 48. Return and reason codes for the IXLCONN macro (continued)		
Hexadecimal return code	Hexadecimal reason code	Equate symbol meaning and action
C	xxxx0C04	<p><b>Equate Symbol:</b> IXLRSNCODEJOINFAILED</p> <p><b>Meaning:</b> Environmental error. IXCJOIN has failed. (XES invokes IXCJOIN on behalf of a user of a serialized structure (TYPE=LOCK or TYPE=LIST with LOCKENTRIES). Additional diagnosis information exists in the connect answer area, IXLYCONA. The field CONADIAG1 contains the IXCJOIN return code and CONADIAG2 contains the IXCJOIN reason code.</p> <p><b>Action:</b> Use the IXCJOIN return and reason codes to understand why IXCJOIN failed and attempt to resolve the problem.</p>
C	xxxx0C05	<p><b>Equate Symbol:</b> IXLRSNCODENOTINPOLICY</p> <p><b>Meaning:</b> Environmental error. The active CFRM policy for the installation does not specify the structure. The request fails.</p> <p><b>Action:</b> Specify a structure that is defined in the CFRM active policy or switch to a new CFRM policy that contains the structure for which this IXLCONN was invoked.</p>
C	xxxx0C06	<p><b>Equate Symbol:</b> IXLRSNCODENOCONN</p> <p><b>Meaning:</b> Environmental error. The system does not have connectivity to the coupling facility that contains the specified structure. This might occur due to operator commands such as VARY PATH,OFFLINE or CONFIG CHP,OFFLINE, or hardware errors such as coupling facility or path failures.</p> <p><b>Action:</b> Determine why connectivity was not established and attempt the request again. In IXLYCONA, the field CONACONNECTORCONNECTIVITY is set if at least one active connector has connectivity to the structure.</p>
C	xxxx0C08	<p><b>Equate Symbol:</b> IXLRSNCODENOFAC</p> <p><b>Meaning:</b> Environmental error. Allocation of the structure failed because there was no suitable coupling facility to allocate the structure based on the preference list in the active CFRM policy.</p> <p><b>Action:</b> In IXLYCONA, the field CONAFACILITYARRAY contains information about each coupling facility in which XES attempted to allocate the structure. Examine each reason code that is associated with each coupling facility to understand the reason why the allocation failed for that coupling facility.</p> <p>For additional information about the connect failure, search the hardcopy log for messages IXL013I and IXL015I. For an XCF signaling list structure, search for message IXC463I. Examine the system LOGREC symptom record for details. The symptom record has the information from the IXLCONN parameter list and the connect answer area (IXLYCONA).</p>

Table 48. Return and reason codes for the IXLCONN macro (continued)

Hexadecimal return code	Hexadecimal reason code	Equate symbol meaning and action
C	xxxx0C09	<p><b>Equate Symbol:</b> IXLRSNCODECONNPREVENTED</p> <p><b>Meaning:</b> Environmental error. New connections to the requested structure are being prevented at this time for one of the following reasons:</p> <ul style="list-style-type: none"> <li>• All active connectors have confirmed either the Rebuild Quiesce event or the Duplex Complete event. New connections will not be permitted until the rebuild stop or rebuild cleanup is completed.</li> <li>• The structure is allocated in a coupling facility that is failed. New connections will not be permitted until the structure is rebuilt, or all connections disconnect causing the structure to be deallocated.</li> <li>• The coupling facility containing the structure is not available for use because policy reconciliation is in progress. New connections will not be permitted until policy reconciliation is complete.</li> <li>• New structure allocations for this structure name are not permitted because there is a pending policy change for this structure. New connections will not be permitted until the structure cleanup is complete.</li> <li>• Structure cleanup is in progress. New connections will not be permitted until the structure cleanup is complete.</li> <li>• A system-managed process (for example, system-managed rebuild) is in progress.</li> </ul> <p>When the structure is in the Duplex Established phase of a system-managed duplexing rebuild, CONASTRUCTURESMDUPESTAB flag will be set. If the user specified or defaulted to ALLOWAUTO=NO on the connect, the connect will be prevented as long as the structure remains duplexed.</p> <p>If the user specified or defaulted to ALLOWALTER=NO on the connect and the two structure instances have a different maximum structure object count, entry-to-element ratio or event monitor control storage percentage, the connect will be prevented as long as the structure remains duplexed with that difference.</p> <ul style="list-style-type: none"> <li>• System-managed asynchronous duplexing is established. When the structure is in the Async Duplex Established phase of a system-managed duplexing rebuild, the CONASTRUCTURESMSYNCDUPESTAB flag is set. If the user specified or defaulted to ASYNCDUPLEX=NO on the connect, the connect is prevented as long as the structure remains async duplex established. The attempt by XES to stop duplexing to allow the connect was not successful.</li> <li>• The connect request originated on a system that is not capable of exploiting some functional support with which the target structure was previously allocated.</li> </ul> <p>For each of the reasons listed, except for the last reason (unable to exploit functional support), an ENF Event Code 35 will signal when the condition preventing new connections completes.</p> <p><b>Action:</b> Determine why connection to the structure was prevented and retry the request at a later time.</p>

Table 48. Return and reason codes for the IXLCONN macro (continued)		
Hexadecimal return code	Hexadecimal reason code	Equate symbol meaning and action
C	xxxx0C0A	<p><b>Equate Symbol:</b> IXLRSNCODESTRNOTALLOCATED</p> <p><b>Meaning:</b> Environmental error. ALLOC=NO was specified and the structure is not allocated.</p> <p><b>Action:</b> Determine why the structure is not allocated. Reissue the IXLCONN request without ALLOC=NO if you wish to allocate the structure.</p>
C	xxxx0C0C	<p><b>Equate Symbol:</b> IXLRSNCODENOCONNDUPLEXNEWSTR</p> <p><b>Meaning:</b> Environmental error. This system does not have connectivity to the coupling facility that contains the specified duplexed new structure. This might occur due to operator command such as VARY PATH,OFFLINE or CONFIG CHP,OFFLINE, or hardware errors such as coupling facility or channel path failures. MVS stops the structure duplexing rebuild for the structure.</p> <p><b>Action:</b> Determine why connectivity was not established and attempt the request again. In IXLYCONA, the field CONACONNECTORCONNECTIVITY is set if at least one active connector has connectivity to the structure.</p>
C	xxxx0C11	<p><b>Equate Symbol:</b> IXLRSNCODEDEFINE</p> <p><b>Meaning:</b> Environmental error. The local vector requested on connect could not be defined on a TYPE=CACHE or TYPE=LIST request.</p> <p><b>Action:</b> Retry the request later.</p>
C	xxxx0C12	<p><b>Equate Symbol:</b> IxlrSnCodeConnStgNotAvail</p> <p><b>Meaning:</b> Environmental error. The system could not create a data space.</p> <p><b>Action:</b> Retry the request later.</p>
C	xxxx0C19	<p><b>Equate Symbol:</b> IXLRSNCODETIMERNOTSET</p> <p><b>Meaning:</b> Environmental error. XES DIE could not be established for this system. Specifically, the XES DIE could not be established for this system.</p> <p><b>Action:</b> Retry the request later.</p>
C	xxxx0C23	<p><b>Equate Symbol:</b> IXLRSNCODECONNOTINPOL</p> <p><b>Meaning:</b> Environmental error. The request failed because there is a failed-persistent connection with the same connection name that has not been reconciled into the CFRM policy.</p> <p><b>Action:</b> Examine your active CFRM policy. The CFRM couple data set requires additional CONNECT records. This might require that the CFRM couple data set be reformatted.</p>

Table 48. Return and reason codes for the IXLCONN macro (continued)		
Hexadecimal return code	Hexadecimal reason code	Equate symbol meaning and action
C	xxxx0C24	<p><b>Equate Symbol:</b>IXLRSNCODEINCOMPATNUMUSER</p> <p><b>Meaning:</b> Environmental error. The composite value of all the NUMUSERS or MAXCONN values specified by connectors to the current structure prevents any additional connections to the structure. This can occur for the following reasons:</p> <ul style="list-style-type: none"> <li>• On initial connect, the available connection identifier is greater than the smallest NUMUSERS or MAXCONN value specified by an existing connection.</li> <li>• On initial connect, the available connection identifier is greater than the NUMUSERS or MAXCONN value specified on the current IXLCONN request.</li> <li>• On Initial connect, the largest connection identifier in use by an existing connection is greater than the NUMUSERS or MAXCONN value specified on the current IXLCONN request.</li> <li>• On a rebuild connect request the largest connection identifier in use by an existing connection to the original structure is greater than the NUMUSERS or MAXCONN value specified on the current IXLCONN REBUILD request.</li> </ul> <p><b>Action:</b> The installation has implemented an unsafe migration path to greater than 32 connectors to a structure. Retry the request at a later time. Examine your protocol to determine why your specification for the maximum userid limit conflicts with the existing connections.</p>
C	xxxx0C25	<p><b>Equate Symbol:</b> IXLRSNCODESTRFAILURE</p> <p><b>Meaning:</b> Environmental error. Request failed because structure failure has occurred. The request fails.</p> <p><b>Action:</b> Retry the request at a later time. The system will issue ENF signal 35 when coupling facility resources become available.</p>

Table 48. Return and reason codes for the IXLCONN macro (continued)

Hexadecimal return code	Hexadecimal reason code	Equate symbol meaning and action
C	xxxx0C27	<p><b>Equate Symbol:</b> IXLRSNCODERSPNOTREC</p> <p><b>Meaning:</b> Environmental error. A connector to a structure has either failed or disconnected. All surviving connectors to the structure have not yet responded to the EEPLDISCFAILCONN event. XES cannot perform cleanup processing for a terminating connector until all surviving connectors have responded.</p> <p>The EEPLDISCFAILCONN event allows users to respond with either a return code in IXLYEEPL or with an IXLYEEPL return code and a later IXLEERSP response. Users are <u>not</u> required to use only IXLEERSP when responding.</p> <p><b>Action:</b> Surviving connectors can specify how XES is to perform cleanup processing by responding to the EEPLDISCFAILCONN event in one of three ways:</p> <ul style="list-style-type: none"> <li>• Set return code X'0' in IXLYEEPL (normal processing)</li> <li>• Set return code X'1' in IXLYEEPL (used for failed-persistent connections)</li> <li>• Set return code X'8' in IXLYEEPL to delay your response and then respond with IXLEERSP.</li> </ul> <p>Examine your protocol to determine why responses have not been issued. The CONADISCFAILEDCONFSTRING field of the IXLYCONA contains a string indicating the connections that must provide a response for the disconnect/failure of the previous instance of the connection. Retry the request at a later time. The ENF signal 35 will be signaled when all responses for the disconnect/failure have been received.</p>
C	xxxx0C29	<p><b>Equate Symbol:</b> IXLRSNCODEXESNOTACTIVE</p> <p><b>Meaning:</b> Environmental error. The CFRM function is not active or not available.</p> <p><b>Action:</b> Bring the CFRM couple data set in use in the sysplex by issuing the SETXCF COUPLE,TYPE=CFRM,PCOUPLE= command.</p>
C	xxxx0C30	<p><b>Equate Symbol:</b> IXLRSNCODEDUMPINPROGRESS</p> <p><b>Meaning:</b> Environmental error. The connection could not be completed because SVC Dump holds serialization on the structure.</p> <p><b>Action:</b> Retry the request at a later time. The ENF signal 35 will be signaled when coupling facility resources become available.</p>
C	xxxx0C3C	<p><b>Equate Symbol:</b> IXLRSNCODEREBUILDCONNECT</p> <p><b>Meaning:</b> Environmental error. The rebuild connection failed because the original connector failed (task was terminated) while processing this IXLCONN REBUILD request. This reason code is only applicable when the IXLCONN REBUILD request is issued from a different task than the task that owns the original connection.</p> <p><b>Action:</b> None. The original connect that failed cannot reconnect.</p>



Table 48. Return and reason codes for the IXLCONN macro (continued)		
Hexadecimal return code	Hexadecimal reason code	Equate symbol meaning and action
C	xxxx0C3D	<p><b>Equate Symbol:</b> IXLRSNCODENOTREBUILDING</p> <p><b>Meaning:</b> Environmental error. The connection failed because the requester issued IXLCONN REBUILD for a structure that is not rebuilding.</p> <p><b>Action:</b> Verify that the structure name has been specified correctly.</p>
C	xxxx0C44	<p><b>Equate Symbol:</b> IXLRSNCODEREBUILDCONNEXISTS</p> <p><b>Meaning:</b> Environmental error. The connection failed because REBUILD connect already exists for this CONNAME.</p> <p><b>Action:</b> Verify that the connection name has been specified correctly.</p>
C	xxxx0C45	<p><b>Equate Symbol:</b> IXLRSNCODEREBUILDBADCONN</p> <p><b>Meaning:</b> Environmental error. Either the issuer of IXLCONN with the REBUILD option is not a connector in the address space or system from which this request was issued or the connector is not active.</p> <p><b>Action:</b> To issue an IXLCONN REBUILD request, the user must be connected to the original structure and the request must be issued from the same address space and system as the original request.</p>
C	xxxx0C47	<p><b>Equate Symbol:</b> IXLRSNCODEREBUILDCONNPHASE</p> <p><b>Meaning:</b> Environmental error. The connection failed because the requester issued IXLCONN REBUILD during the wrong phase of the rebuild process. The cause is one of the following:</p> <ul style="list-style-type: none"> <li>• The IXLCONN REBUILD was issued before all connections have provided a response to the Rebuild Quiesce event.</li> <li>• The IXLCONN REBUILD was issued after all connections have issued IXLREBLD REQUEST=COMPLETE.</li> </ul> <p><b>Action:</b> Examine your protocol to determine why the request was issued during the wrong phase of the rebuild process.</p>
C	xxxx0C4E	<p><b>Equate Symbol:</b> IXLRSNCODEREBUILDCONNECTSTOP</p> <p><b>Meaning:</b> Environmental error. The IXLCONN REBUILD request was not successful because a rebuild stop occurred.</p> <p><b>Action:</b> Determine why the rebuild stop occurred.</p>
C	xxxx0C50	<p><b>Equate Symbol:</b> IXLRSNCODEREBUILDCONNECTNOPREF</p> <p><b>Meaning:</b> Environmental error. The IXLCONN REBUILD request was not successful because there is no policy information for this structure in the active CFRM policy. This structure was reconciled into the policy from the coupling facility when the sysplex was IPLed with a downlevel CFRM couple data set.</p> <p><b>Action:</b> Switch to a new CFRM active policy that includes the definition for this structure.</p>

Table 48. Return and reason codes for the IXLCONN macro (continued)

Hexadecimal return code	Hexadecimal reason code	Equate symbol meaning and action
C	xxxx0C52	<p><b>Equate Symbol:</b> IXLRSNCODEALLOWREBLD</p> <p><b>Meaning:</b> Environmental error. The connect request was rejected because a connector specified either ALLOWREBLD=NO and structure rebuild is in progress or ALLOWDUPREBLD=NO and structure duplexing is in progress.</p> <p><b>Action:</b> Examine your protocol to determine why a user specified that rebuild or duplexing was not to be allowed.</p>
C	xxxx0C53	<p><b>Equate Symbol:</b> IXLRSNCODECFLEVEL</p> <p><b>Meaning:</b> Environmental error. The connect request was not successful because the requester specified a CFLEVEL value that is greater than the maximum CFLEVEL supported by the MVS release on which the IXLCONN was requested.</p> <p><b>Action:</b> The maximum CFLEVEL supported by the MVS release is returned in the CONAMVSRELEASEMAXCFLEVEL field of the IXLYCONA. Interrogate this field to determine the supported level.</p>
C	xxxx0C64	<p><b>Equate Symbol:</b> IXLRSNCODESTRALTERNOTALLOW</p> <p><b>Meaning:</b> Environmental error. The connect request was rejected because alter is in progress and the connection specified ALLOWALTER=NO on IXLCONN.</p> <p><b>Action:</b> Wait until the structure alter is complete and then retry the connect request. The system issues an ENF event code 35 specifying the name of the structure when the structure alter completes.</p>
C	xxxx0C65	<p><b>Equate Symbol:</b> IXLRSNCODESTRALTERRESTRICT</p> <p><b>Meaning:</b> The connect request was rejected because alter is in progress and the RATIO, MINENTRY, and/or MINELEMENT specification is more restrictive than the current composite for existing connections.</p> <p><b>Action:</b> Wait until the structure alter is complete and then retry the connect request. The system issues an ENF event code 35 specifying the name of the structure when the structure alter completes.</p>
C	xxxx0C9A	<p><b>Equate Symbol:</b> IX.RSNCODEINSUFFCFLEVELUSER</p> <p><b>Meaning:</b> The connect request was rejected because the user specified a MINCFLEVEL value that is greater than the level of the coupling facility in which the target structure is allocated. The level of the coupling facility in which the target structure is allocated is returned in the CONACFACILITYCFLEVEL field of IXLYCONA.</p> <p><b>Action:</b> Determine the level of the coupling facility in which the target structure is allocated. If the IXLCONN request failed because the structure is allocated in a coupling facility that is below the program's minimum requirements for CFLEVEL as specified by MINCFLEVEL, have the structure moved to a coupling facility that is at or above that MINCFLEVEL. Attempt the IXLCONN request with the same MINCFLEVEL.</p>

Table 48. Return and reason codes for the IXLCONN macro (continued)		
Hexadecimal return code	Hexadecimal reason code	Equate symbol meaning and action
C	xxxx0CA3	<p><b>Equate Symbol:</b> IXLRSNCODESTRALTERSCM</p> <p><b>Meaning:</b> Environmental error. Request rejected because the connector specified ALLOWALTER=NO but the target structure is allocated to support storage-class memory.</p> <p><b>Action:</b> If storage-class memory is intended for use with the structure, the application must receive an upgrade so that all connectors support alter.</p>
C	FFFFFFFF	<p><b>Equate Symbol:</b> IXLRSNCODENOTAVAILABLE</p> <p><b>Meaning:</b> Environmental error. XES functions are not available. This can occur because the coupling facility hardware necessary to provide XES functions is not present.</p> <p><b>Action:</b> Re-IPL the system, or follow your particular failure management protocol.</p>
10	xxxx10xx	<p><b>Equate Symbol:</b> IXLRETCODECOMPERROR</p> <p><b>Meaning:</b> Failure in XES processing. Additional diagnostic information might exist in the following fields in IXLYCONA: CONADIAG0, CONADIAG1, CONADIAG2, CONADIAG6, CONADIAG7, CONADIAG8, and CONADIAG9.</p> <p>When the reason code is 02011007 with diagnostic information:</p> <ul style="list-style-type: none"> <li>• CONADIAG0: 00000002</li> <li>• CONADIAG1: 0000000C</li> <li>• CONADIAG2: 00021007</li> </ul> <p>The system might not have connectivity to the coupling facility that contains the specified structure. This might occur due to operator commands such as VARY PATH,OFFLINE or CONFIG CHP,OFFLINE, or hardware errors such as coupling facility or path failures. Determine why connectivity was not established and attempt the request again.</p> <p><b>Action:</b> Provide the IXLYCONA information listed earlier to the IBM support center.</p>



## Chapter 51. IXLCSP – XES Structure Computation Service

### Description

The IXLCSP service assists in coupling facility capacity planning and exploitation by providing a means to evaluate the following coupling facility structure values:

- Required structure size, given structure attributes and object counts
- Resulting structure object counts, given structure attributes and size.
- Required storage-class memory (SCM) size, given structure attributes and object counts in SCM.

Calculations do not result in the actual allocation of structures, nor do they affect the current contents of the target coupling facility.

Calculations are appropriate to the CFLEVEL of the target coupling facility, but do not take into account current coupling facility conditions such as storage constraints. These conditions might prevent the actual allocation of a structure in the target coupling facility, or might cause the structure to be allocated with significantly different counts than returned by IXLCSP.

You can use the IXLMG service to obtain additional information about the target coupling facility, such as CFLEVEL, model-dependent limits, and storage.

### Environment

The following are the environment requirements for the caller.

Environment	Environment requirement
Minimum authorization:	Supervisor state or PKM allowing key 0 - 7
Dispatchable unit mode:	Task or SRB
Cross memory mode:	PASN=HASN, any SASN.
AMODE:	31-bit
ASC mode:	Primary or access register (AR)
Interrupt status:	Enabled for I/O and external interrupts
Locks:	No locks held, with no enabled, unlocked task (EUT) FRRs established.
Control parameters:	Must be in the primary address space or be in an address/data space that is addressable through a public entry on the caller's dispatchable unit access list (DU-AL)

### Programming Requirements

The caller's parameter list must be addressable from the unit of work issuing the request.

The IXLYCSPA mapping macro provides the format of the area that the ANSAREA points to. Include that macro in your program.

## Input Register Information

---

Before issuing the IXL CSP macro, the caller does not have to place any information into any register unless using it in register notation for a particular parameter, or using it as a base register.

## Output Register Information

---

When control returns to the caller of the IXL CSP macro, the general purpose registers (GPRs) contain:

### Register

#### Contents

**0**

Reason code, if applicable, if GPR 15 return code is non-zero

**1**

Used as a work register by the system

**2-13**

Unchanged

**14**

Used as a work register by the system

**15**

Return code

When control returns to the caller of the IXL CSP macro, the access registers (ARs) contain:

### Register

#### Contents

**0-1**

Used as work registers by the system

**2-13**

Unchanged

**14-15**

Used as work registers by the system

**For registers that the system changes**, a caller who depends on these registers containing the same value before and after issuing the macro must save these registers before issuing the macro and restore them after the system returns control.

## Performance Implications

---

None.

## Understanding IXL CSP version support

---

The IXL CSP macro supports versions 0 through 4.

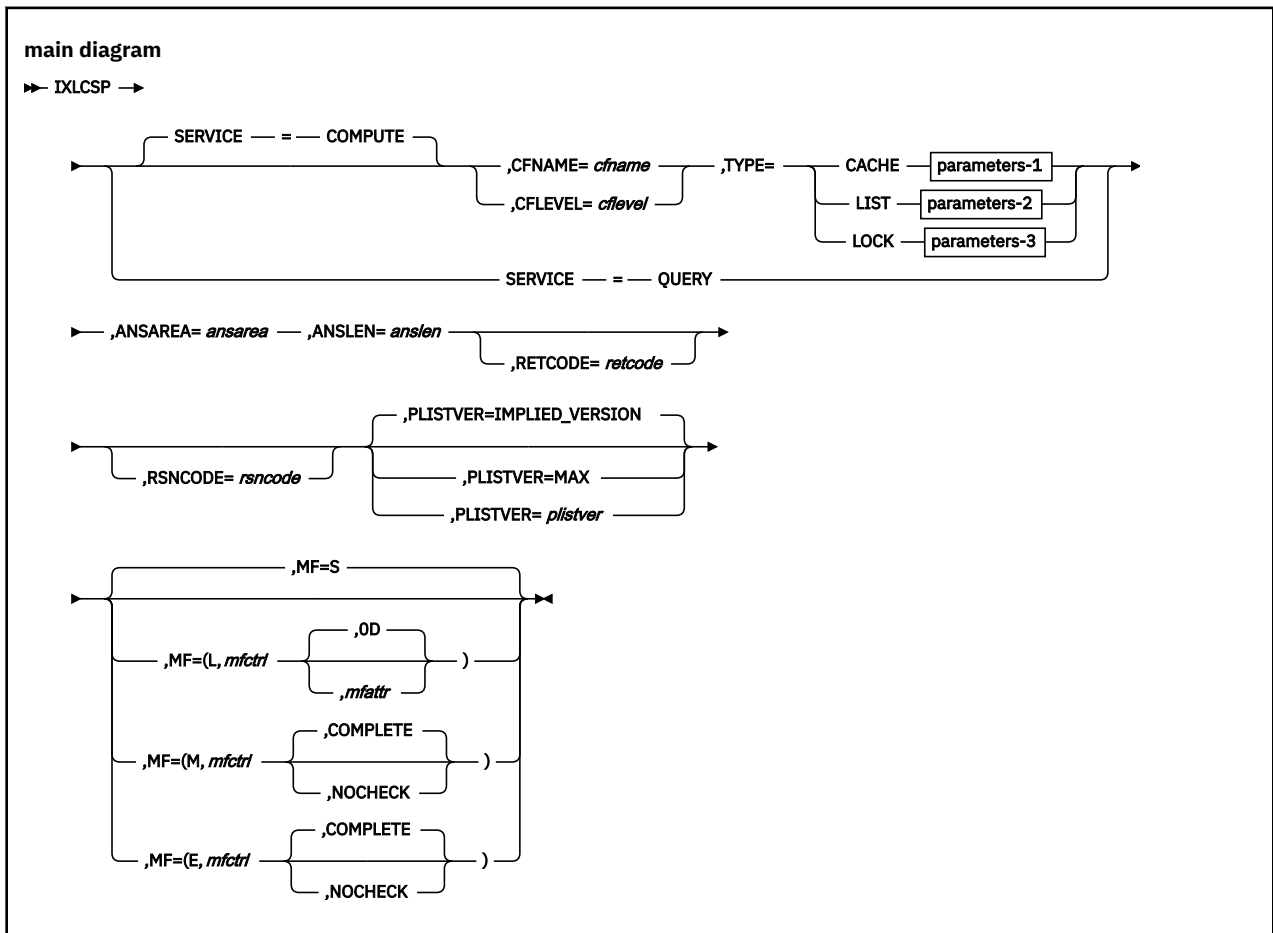
- Keywords not specifically noted here are supported by version 0 and subsequent versions of the IXL CSP macro.
- The following keyword is supported by version 1 and subsequent versions of the IXL CSP macro.
  - KEYTYPE
- The following keywords are supported by version 2 and subsequent versions of the IXL CSP macro.
  - SCMALGORITHM
  - SCMELEMENTCOUNT
  - SCMENTRYCOUNT
  - SCMMAXSIZE

- The following keywords are supported by version 4 and subsequent versions of the IXLCSP macro.
  - CFLEVEL
  - SERVICE

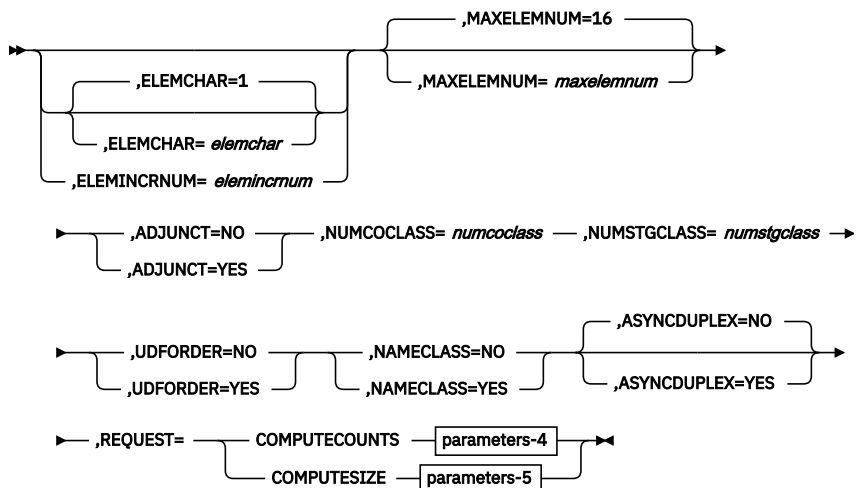
Specify the version of the parameter list that you want generated with the PLISTVER keyword. See Chapter 2, “Specifying a Macro Version Number,” on page 5 for considerations when specifying the version of the parameter list with PLISTVER.

## Syntax diagram

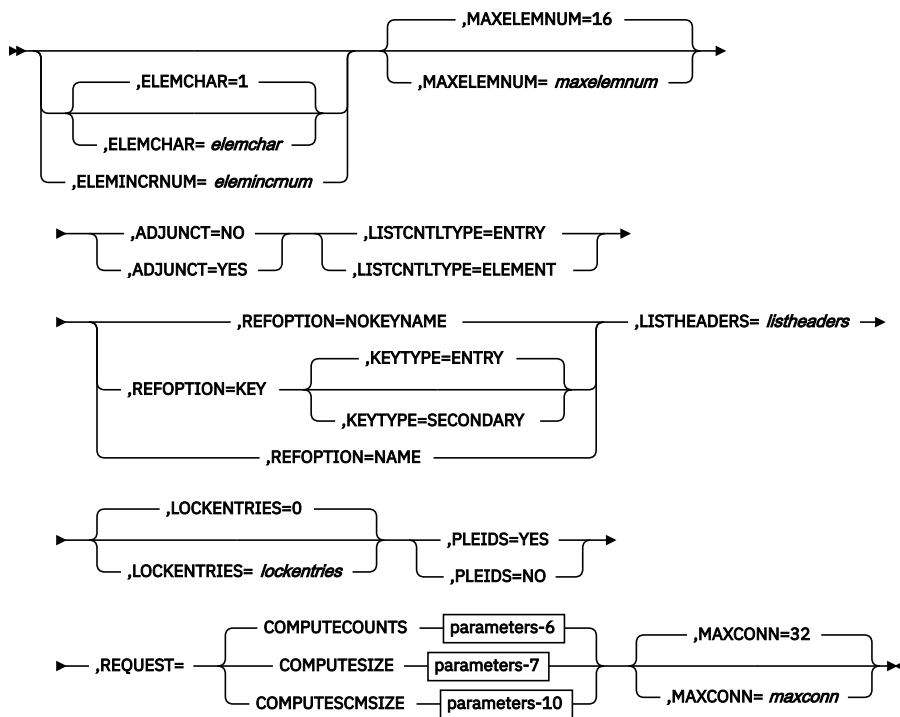
The syntax of the IXLCSP macro is as follows:



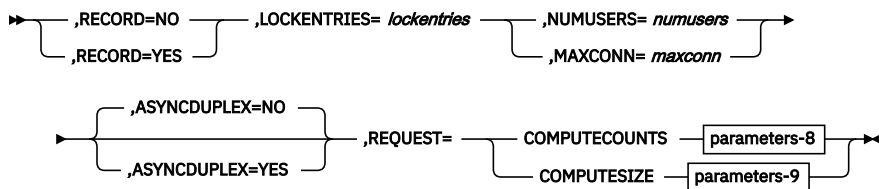
## parameters-1



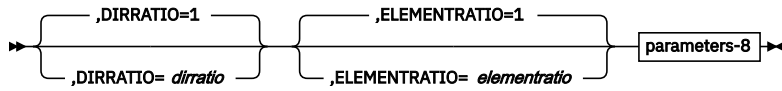
## parameters-2



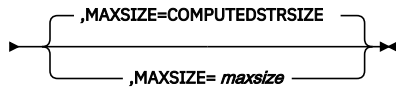
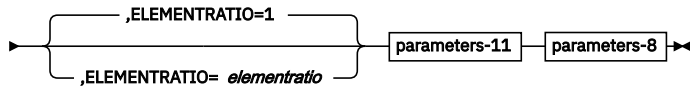
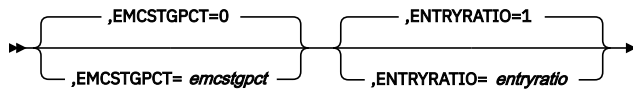
## parameters-3





**parameters-4****parameters-5**

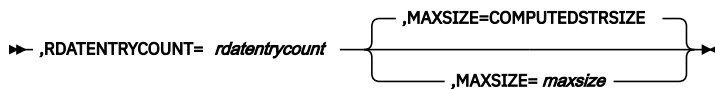
>> ,DIRENTRYCOUNT= *direntrycount* — ,ELEMENTCOUNT= *elementcount* →

**parameters-6****parameters-7**

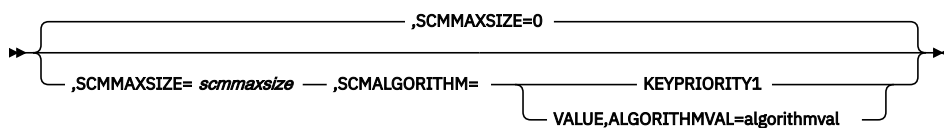
>> ,EMCCOUNT= *emccount* — ,ENTRYCOUNT= *entrycount* — ,ELEMENTCOUNT= *elementcount* →

**parameters-8**

>> ,STRSIZE= *strsize* — ,MAXSIZE= *maxsize* →

**parameters-9****parameters-10**

>> ,SCMENTRYCOUNT= *scmentrycount* — ,SCMELEMENTCOUNT= *scmelementcount* →

**parameters-11**

## Parameter descriptions

---

The parameter descriptions are listed in alphabetical order. Default values are underlined:

**,ADJUNCT=NO**

**,ADJUNCT=YES**

Use this input parameter to indicate whether adjunct data is associated with each structure entry. Adjunct data is 64 bytes per entry.

**NO**

The structure entries do not have associated adjunct data.

**YES**

The structure entries have associated adjunct data.

**,ALGORITHMVAL=algorithmval**

Use this input value to specify the value, as defined in IXLYCON, of the SCMALGORITHM value that the CF will use to manage storage-class memory. The following IXLYCON constants are valid values:

- IXLALGORITHMKEYPRIORITY1

If you specify a value other than one of the valid IXLYCON constants, the IXLCSPP request fails with reason code IXLRSNCOEADSCMALGORITHM. See the appropriate SCMALGORITHM keyword values for specific algorithm information.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a one-byte input field that contains a representation of the desired storage-class memory management algorithm.

**,ANSAREA=ansarea**

Use this output area to contain information returned about the request. The storage area is mapped by the IXLYCSPA macro.

Not all fields in the answer area are applicable to all request types.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an output area to contain information about the request.

**,ANSLEN=anslen**

Use this input parameter to specify the length of the answer area. The length should be long enough to accommodate the IXLYCSPA mapping of the answer area.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword input field that contains the length of the answer area.

**,ASYNCDUPLEX=NO**

**,ASYNCDUPLEX=YES**

Use this optional input parameter to indicate whether the structure for which the computation is being performed is expected to be duplexed using the system-managed asynchronous duplexing protocol.

**NO**

The structure will not be asynchronously duplexed. NO is the default.

**YES**

The structure will be asynchronously duplexed and thus requires the additional controls necessary to support that protocol.

ASYNCDUPLEX=YES is valid when the target coupling facility is at a CFLEVEL  $\geq$  21. If it is specified for a CF at a lower CFLEVEL, the request will complete with return code 4 reason code IxIRsnCodeWarningCFLevel and will not consider asynchronous duplexing in performing the computations.

**CFLEVEL=cflevel**

Use this input parameter to specify the CFLEVEL for which computations are to be performed. The system can support calculations for any CFLEVEL that is installed in a CF accessible to the invoking system and for CFLEVEL(s) that are emulated in software. The set of emulated CFLEVELs varies

by z/OS release and maintenance level. Use IXLCSP SERVICE=QUERY to obtain a list of CFLEVELs supported by the invoking system.

An input value of 0 requests calculation at the highest CFLEVEL available to the invoking system. On successful completion, CSPA\_ActualCfLevel will contain the CFLEVEL for which the calculation was performed.

Test the QuReqRfIxlcpCFLevel bit in the QuReqFeatures string returned by an IXCQUERY REQINFO=FEATURES invocation to determine whether the current system supports the CFLEVEL keyword. Macro IXCYQUAA defines the QuReqRfIxlcpCFLevel flag and QuReqFeatures string.

#### **CFNAME=*cfname***

Use this input parameter to specify the name of the coupling facility for which computations are to be performed. The specified coupling facility must be:

- Defined in the active CFRM policy
- Connected to the system from which the request is issued
- CFLEVEL=8 or higher.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-character input field that contains the name of the coupling facility.

#### **,DIRENTRYCOUNT=*direntrycount***

Use this input parameter to specify the desired maximum number of directory entries in the structure.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword input area to contain the maximum number of directory entries.

#### **,DIRRATIO=*1***

#### **,DIRRATIO=*dirratio***

Use this input parameter to specify a value to express the directory portion of the directory-to-element ratio of the coupling facility structure. The value of DIRRATIO must be greater than zero.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the 2-byte input field that contains a value to express the directory portion of the directory-to-element ratio of the coupling facility structure.

#### **,ELEMCHAR=*0***

#### **,ELEMCHAR=*elemchar***

Use this input parameter to specify the value of an element characteristic used to determine the element size. The element size is calculated with the formula  $256 \times (2^{**ELEMCHAR})$ , where ELEMCHAR is used as the power of 2. For example, if ELEMCHAR=0, then the size of each element is 256 bytes.

The valid values for ELEMCHAR range from zero to a maximum determined by the coupling facility limitation on element size.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the 1-byte input field that contains the value of the element characteristic, used to determine the structure element size.

**Note:** The element size, in combination with the maximum number of elements (MAXELEMNUM parameter), determines the size of the data entry — the data written to and read from the structure. With a coupling facility of CFLEVEL=0, a data entry can be up to 16 times the data element size, as indicated by the element characteristic. With a coupling facility of CFLEVEL=1 or higher, a data entry can be up to 255 times the data element size.

Use either ELEMCHAR or ELEMINCRNUM to define the element size. You can specify either an element characteristic or an element increment number for element size determination.

#### **,ELEMENTCOUNT=*elementcount***

Use this input parameter to specify the desired maximum number of data elements in the structure.

For a directory-only cache structure, or a list structure without data elements, specify ELEMENTCOUNT=0. Zero cannot be specified when ELEMENT is specified for LISTCNTLTYPE.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword input area to contain the maximum number of data elements in the structure.

**,ELEMENTRATIO=1**

**,ELEMENTRATIO=elementratio**

Use this input parameter to specify a value to express the element portion of the directory-to-element ratio for a cache structure or the entry-to-element ratio for a list structure. If ELEMENTRATIO is zero, the calculation reflects a structure allocated without data elements. Zero cannot be specified when ELEMENT is specified for LISTCNTLTYPE.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the 2-byte input field that contains a value to express the element portion of the directory-to-element ratio of the coupling facility structure.

**,ELEMINCRNUM=elemincrnum**

Use this input parameter to specify the element increment number. The element size is calculated with the formula  $256 * \text{ELEMINCRNUM}$ . For example, if ELEMINCRNUM=1, then the size of each element is 256 bytes.

The valid values for ELEMINCRNUM range from 1 to a maximum determined by the coupling facility limitation on element size. The value of ELEMINCRNUM must be a power of 2.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the 1-byte input field that contains the value of the element increment number, used to determine the cache structure element size.

**Note:** The element size, in combination with the maximum number of elements (MAXELEMNUM parameter), determines the size of the data entry — the data written to and read from the structure. With a coupling facility of CFLEVEL=0, a data entry can be up to 16 times the data element size, as indicated by the element increment number. With a coupling facility of CFLEVEL=1 or higher, a data entry can be up to 255 times the data element size.

Use either ELEMCHAR or ELEMINCRNUM to define the element size. You can specify either an element characteristic or an element increment number for element size determination.

**,EMCCOUNT=emccount**

Use this input parameter to specify the desired maximum number of event monitor controls in the structure.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword input area to contain the maximum number of event monitor controls in the structure.

**,EMCSTGPCT=0**

**,EMCSTGPCT=emcstgpct**

Use this input parameter to specify the percentage of available list structure storage that is to be set aside for event monitor controls (EMCs). The amount of storage requested is that over and above the storage amount included in the marginal structure size.

Specify the percentage value in hundredths of a percent. For example, to request a value of 20%, code EMCSTGPCT=2000.

The value of EMCSTGPCT must be in the range of 0 to 10000. A nonzero value indicates that the connector intends to perform sublist monitoring. Note that for sublist monitoring, the structure must be a keyed list structure (REFOPTION=KEY).

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the halfword field that contains a value to express the percentage of available structure storage, over and above the storage amount included in the marginal structure size, that is to be set aside for event monitor controls (EMCs).

**,ENTRYCOUNT=entrycount**

Use this input parameter to specify the desired maximum number of list entries in the structure.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword input area to contain the maximum number of directory or list entries in the structure.

**,ENTRYRATIO=1****,ENTRYRATIO=entryratio**

Use this input parameter to specify a value to express the list entry portion of the entry-to-element ratio of the coupling facility list structure.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the 2-byte field that contains a value to express the list entry portion of the entry-to-element ratio of the coupling facility list structure.

**,KEYTYPE=ENTRY****,KEYTYPE=SECONDARY**

Use this input parameter to indicate whether keyed list entries will use entry keys and secondary keys or only entry keys.

**ENTRY**

Specifies that only list entry keys are used.

**SECONDARY**

Specifies that list entry keys and secondary list entry keys are used.

Secondary keys are stored in the first 32 bytes of the 64-byte list entry adjunct area. Therefore, when KEYTYPE=SECONDARY is specified, adjunct data must also be requested by specifying ADJUNCT=YES. KEYTYPE=SECONDARY requires a coupling facility of CFLEVEL=9 or higher.

**,LISTCNTLTYPE=ENTRY****,LISTCNTLTYPE=ELEMENT**

Use this input parameter to specify whether the system maintains and tracks list limits based on number of entries or number of elements.

**ENTRY**

Specifies that the list limits are specified and tracked as limits on the number of entries that may reside on the list.

**ELEMENT**

Specifies that the list limits are specified and tracked as limits on the total number of data elements that may be associated with entries on the list.

**Note:** Neither ELEMENTRATIO nor ELEMENTCOUNT can be specified as zero with this specification.

**,LISTHEADERS=listheaders**

Use this input parameter to specify the number of lists (list headers) for the coupling facility list structure. The number must be greater than zero.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the fullword input field that contains the number of list headers to be allocated.

**,LOCKENTRIES=0****,LOCKENTRIES=lockentries**

Use this input parameter to specify the number of lock entries that are associated with the structure. If the value for the number of lock entries is not a power of 2, it is rounded upward to the nearest power of 2.

- When specified for a list structure, a LOCKENTRIES value of 0 indicates an unserialized list structure.
- When specified for a lock structure, a LOCKENTRIES value of 0 indicates that IXLCSP is to determine the maximum number of locks that can be accommodated by a structure of the specified size. Specifying LOCKENTRIES=0 is valid only when RECORD=NO is specified.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the fullword field that contains the number of lock entries for the coupling facility structure.

**,MAXCONN=32****,MAXCONN=***maxconn*

Use this input parameter to specify the maximum number of users that can connect and use the list or lock structure. When the requested MAXCONN keyword value is in the range 0 to 32, the value is 32.

To perform computations for a structure that is to support more than 32 connectors, you must use the keyword MAXCONN, and ensure that the target coupling facility is CFLEVEL=17 or higher. If you use MAXCONN on a request on a request that is targeted for a coupling facility with a CFLEVEL less than 17, the value for the computation is limited by the user-id limit of the coupling facility.

For lock structures use either NUMUSERS or MAXCONN to specify the maximum number of users that can connect to the lock structure.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the 1-byte field that contains the maximum number of connectors to the coupling facility structure.

**,MAXELEMNUM=16****,MAXELEMNUM=***maxelemnum*

Use this input parameter to specify a value to determine the maximum number of data elements for each data entry in the coupling facility structure. MAXELEMNUM must be in the decimal range of 1 to 255.

The maximum data entry size in bytes equals MAXELEMNUM multiplied by the element size that is obtained from the value specified for ELEMCHAR or ELEMINCRNUM.

For example, if ELEMCHAR=0 and MAXELEMNUM=1, the maximum data entry size in bytes is 256. If ELEMCHAR=4 and MAXELEMNUM=16, the maximum data entry size in bytes is 4096(16), or 65,536.

To ensure that the system can assign all possible data elements that are allocated in a structure to a directory or list entry, MAXELEMNUM must be greater than or equal to:

- ELEMENTRATIO divided by DIRRATIO (for cache structures)
- ELEMENTRATIO divided by ENTRYRATIO (for list structures)

when computing structure counts. MAXELEMNUM is ignored if ELEMENTRATIO is zero.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the 1-byte input field that contains the maximum number of data elements for each data entry in the coupling facility structure.

**,MAXSIZE=COMPUTEDSTRSIZE****,MAXSIZE=***maxsize*

Use this input parameter to specify the maximum size of the structure in units of 4K blocks. The MAXSIZE parameter corresponds to the SIZE parameter that would be used in defining the structure in the CFRM policy.

If MAXSIZE is specified, and its value is greater than the size computed from the other input parameters, then it is also used as an input to the computation. If MAXSIZE is not specified, the computation uses the computed structure size as the maximum size.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword input area to contain the maximum size of the structure in units of 4K blocks.

**,MF=S**  
**,MF=(L,mfctrl)**  
**,MF=(L,mfctrl,mfattr)**  
**,MF=(L,mfctrl,0D)**  
**,MF=(M,mfctrl)**  
**,MF=(M,mfctrl,COMPLETE)**  
**,MF=(M,mfctrl,NOCHECK)**  
**,MF=(E,mfctrl)**  
**,MF=(E,mfctrl,COMPLETE)**  
**,MF=(E,mfctrl,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an in-line parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

#### **,mfctrl**

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the parameter list.

#### **,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembler DS pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

#### **,COMPLETE**

#### **,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

#### **COMPLETE**

Use this parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

#### **NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if SMILE=*var* were an optional parameter and the default is SMILE=NO\_SMILE then it would not be documented. However, if the default was SMILE=:), then it would be documented because a value would be the default.

#### **,NAMECLASS=NO**

#### **,NAMECLASS=YES**

Use this input parameter to indicate whether the structure supports name classes. Name classes are supported when the IXLCONN request that results in allocation of the structure specifies a value of NAMECLASSMASK other than X'0000' or X'FFFF'.

#### **NO**

The structure does not support name classes.

**YES**

The structure supports name classes.

**,NUMCOCLASS=*numcoclass***

Use this input parameter to specify the desired number of cast-out classes to be used with the coupling facility cache structure.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the fullword input field that contains the number of cast-out classes.

**,NUMSTGCLASS=*numstgclass***

Use this input parameter to specify the desired number of storage classes used with the coupling facility cache structure.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the fullword input field that contains the number of storage classes.

**,NUMUSERS=*numusers***

Use this input parameter to specify the maximum number of users that can connect and use the lock structure.

When you use the NUMUSERS keyword, the actual number of connections that are used for computations does not exceed 32. If NUMUSERS specifies a value over 32, the value for the computation is set to 32.

For XC to perform computations for a structure that supports more than 32 connectors, you must use the keyword MAXCONN and ensure that the target coupling facility is CFLEVEL=17 or higher. If you use MAXCONN is on a request targeted for a coupling facility with a CFLEVEL less than 17, the value of the computation is limited by the user-id limit of the coupling facility.

For lock structures, use either NUMUSERS or MAXCONN to specify the maximum number of users that can connect to the lock structure.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the 1-byte field that contains the maximum number of connectors to the coupling facility lock structure.

**,PLEIDS=YES****,PLEIDS=NO**

Use this input parameter to specify whether the list structure is to be allocated with programmable list entry identifiers (PLEIDs). For list structures that are to be used for XCF signaling, specify PLEIDS=NO. For all other list structures, specify PLEIDS=YES.

**YES**

The calculation will reflect a list structure allocated with PLEIDs.

**NO**

The calculation will reflect a list structure that is not allocated with PLEIDs.

**,PLISTVER=IMPLIED\_VERSION****,PLISTVER=MAX****,PLISTVER=*plistver***

Use this input parameter to specify the version of the macro. See [“Understanding IXLCSP version support”](#) on page 902 for a description of the options available with PLISTVER.

**,RDATENTRYCOUNT=*rdatentrycount***

Use this input parameter to specify the desired number of record data entries in a lock structure.

RDATENTRYCOUNT is ignored when RECORD=NO is specified.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword input field that contains the maximum number of record data entries.

**,RECORD=NO****,RECORD=YES**

Use this input parameter to specify whether the lock structure has associated record data.



**NO**

The lock structure does not have record data.

**YES**

The lock structure does have record data.

**,REFOPTION=NOKEYNAME****,REFOPTION=KEY****,REFOPTION=NAME**

Use this input parameter to specify how to reference list entries in the coupling facility list structure. The list entry can always be referenced by list entry identifier (LEID) or by unkeyed position. Choose one of the following options to specify how the list entry is to be referenced.

**NOKEYNAME**

The list entries have neither keys nor names.

**KEY**

The list entries have keys.

**NAME**

The list entries have names.

**,REQUEST=COMPUTECOUNTS****,REQUEST=COMPUTESIZE****,REQUEST=COMPUTESCMSIZE**

Use this input parameter to specify the function requested.

**COMPUTECOUNTS**

Compute the number of structure objects that can be contained within a structure of the specified size and attributes.

**COMPUTESIZE**

Compute the size of a structure having the specified attributes and containing the specified number of objects.

**COMPUTESCMSIZE**

Compute the amount of storage-class memory that is required to accommodate the specified number of objects.

REQUEST=COMPUTESCMSIZE is valid when the target coupling facility is at a CFLEVEL greater than or equal to 19.

**,RETCODE=*retcode***

Use this output parameter to specify the location in which the system is to copy the return code from GPR 15.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword to contain the return code.

**,RSNCODE=*rsncode***

Use this output parameter to specify the location in which the system is to copy the reason code from GPR 0.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword to contain the reason code.

**,SCMALGORITHM=KEYPRIORITY1****,SCMALGORITHM=VALUE**

Use this input parameter to identify the algorithm that the coupling facility uses to manage the storage-class memory that is associated with this structure.

This parameter is ignored if SCMMAXSIZE is specified as 0.

**KEYPRIORITY1**

The first byte of the list entry key identifies the migration priority. Data elements, REFOPTION=KEY, and LISTHEADERS=1023 are required.

**VALUE**

Indicates the use of the contents of ALGORITHMVAL to identify the algorithm that is used to manage storage-class memory.

**,SCMELEMENTCOUNT=*scmelementcount***

Use this input parameter to specify the desired maximum number of data elements that can overflow to storage-class memory.

The minimum value of SCMELEMENTCOUNT with SCMALGORITHM=KEYPRIORITY1 is 1. The maximum value of SCMELEMENTCOUNT is 4294967295 ( $2^{32}-1$ ).

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-byte input area to contain the maximum number of data elements in storage-class memory for the structure.

**,SCMENTRYCOUNT=*scmentrycount***

Use this input parameter to specify the desired maximum number of list entries that can overflow to storage-class memory.

The minimum value of SCMENTRYCOUNT with SCMALGORITHM=KEYPRIORITY1 is 1. The maximum value of SCMENTRYCOUNT is 4294967295 ( $2^{32}-1$ ).

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-byte input area to contain the maximum number of list entries in storage-class memory for the structure.

**,SCMMAXSIZE=*scmmaxsize***

Use this input parameter to specify, in units of 4K blocks, the maximum amount of storage-class (flash) memory that the structure is allowed to use. The SCMMAXSIZE keyword corresponds to the parameter of the same name that is used in defining the structure in the CFRM policy.

The maximum value of SCMMAXSIZE is 4294967295 ( $2^{32}-1$ ).

SCMMAXSIZE is valid when the target coupling facility is at a CFLEVEL greater than or equal to 19.

**To Code:** Specify, in units of 4K blocks, the RS-type name or address (using a register from 2 to 12) of an 8-byte input area to contain the maximum amount of storage-class memory for the structure.

**SERVICE=COMPUTE****SERVICE=QUERY**

Use this input parameter to specify the service requested.

**COMPUTE**

Perform a structure sizing calculation.

**QUERY**

Report the set of CFLEVELs for which the invoking system can perform structure sizing calculations. Answer area mapping CSPA\_CFLevelInfo describes the set of supported CFLEVELs.

Test the QuReqRfIxlcsCFLevel bit in the QuReqFeatures string returned by an IXCQUERY REQINFO=FEATURES invocation to determine whether the current system supports SERVICE=QUERY. Macro IXCYQUAA defines the QuReqRfIxlcsCFLevel flag and QuReqFeatures string.

**,STRSIZE=*strsize***

Use this input parameter to specify the structure size in units of 4K blocks. The STRSIZE parameter corresponds to the INITSIZE parameter that would be used in defining the structure in the CFRM policy.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword input field to contain the structure size in units of 4K blocks.

**,TYPE=CACHE****,TYPE=LIST****,TYPE=LOCK**

Use this input parameter to specify the type of structure for which computations are to be made.

**,UDFORDER=NO**

**,UDFORDER=YES**

Use this input parameter to indicate whether a user data field (UDF) order queue should be maintained for each cast-out class for the structure.

**NO**

The calculation is to reflect a cache structure that does not maintain UDF order queues.

**YES**

The calculation is to reflect a cache structure that maintains UDF order queues.

## ABEND Codes

Abend X'026' (See *z/OS MVS System Codes* for more information on this abend.)

## Return and Reason Codes

When the IXL CSP macro returns control to your program:

- GPR 15 (and *retcode*, if you coded RETCODE) contains a return code.
- GPR 0 (and *rsncode*, if you coded RSNCODE) contains a reason code if applicable.

Macro IXLYCON provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

**0**

IXLRETCODEOK

**4**

IXLRETCODEWARNING

**8**

IXLRETCODEPARMERROR

**C**

IXLRETCODEENVERROR

**10**

IXLRETCODECOMPERROR

The following table identifies the hexadecimal return and reason codes and the equate symbol associated with each reason code.

Table 49. Return and Reason Codes for the IXL CSP Macro		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0	None.	<b>Meaning:</b> IXL CSP request successful. <b>Action:</b> None.
4	xxxx0431	<b>Equate Symbol:</b> IXLRSNCODEWARNINGCFLEVEL <b>Meaning:</b> Warning. Request completed, but the target coupling facility was at a CFLEVEL too low to process some of the specified parameters. The computation did not take those parameters into account. The CSPA_DIAGNOSTICCODE field of the IXLYCSPA answer area identifies the parameters that required a higher CFLEVEL. <b>Action:</b> Repeat the IXL CSP request and specify a coupling facility at or above the CFLEVEL that is identified by the CSPA_NEEDED CFLEVEL field of the IXLYCSPA answer area.

Table 49. Return and Reason Codes for the IXLCSPP Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0801	<b>Equate Symbol:</b> IXLRSNCODEBADPARMLIST <b>Meaning:</b> Program error. The parameter list is either not addressable or not accessible. <b>Action:</b> Verify that the parameter list address is valid.
8	xxxx0802	<b>Equate Symbol:</b> IXLRSNCODEBADPARMLISTALET <b>Meaning:</b> Program error. The parameter list ALET does not represent either the caller's primary address space or an address or data space on the caller's DU-AL. <b>Action:</b> Verify that the parameter list ALET is valid.
8	xxxx0804	<b>Equate Symbol:</b> IXLRSNCODEBADVERSIONNUM <b>Meaning:</b> The version number in the parameter list is not valid. <b>Action:</b> <ul style="list-style-type: none"> <li>Verify that your program did not overlay the parameter list storage.</li> <li>Verify that your program was assembled with the correct macro library for the release of z/OS on which your program is running.</li> </ul>
8	xxxx0807	<b>Equate Symbol:</b> IXLRSNCODENOTENABLED <b>Meaning:</b> Program error. Caller is not enabled. <b>Action:</b> Verify that the program is enabled for I/O and external interrupts.
8	xxxx0809	<b>Equate Symbol:</b> IXLRSNCODEPRIMARYNOTHOME <b>Meaning:</b> Program error. The requestor's primary address space is not equal to the requestor's home address space. <b>Action:</b> Ensure that the IXLCSPP service is invoked in the correct cross-memory environment.
8	xxxx080D	<b>Equate Symbol:</b> IXLRSNCODEAREATOOSMALL <b>Meaning:</b> Program error. ANSAREA is too small, as specified by ANSLEN. <b>Action:</b> Ensure that the ANSLEN parameter properly reflects the size of the area specified by ANSAREA. Also, make sure that the length of ANSAREA is large enough to contain the data returned.
8	xxxx080E	<b>Equate Symbol:</b> IXLRSNCODEBADAREA <b>Meaning:</b> Program error. The answer area specified by ANSAREA is either not addressable or not accessible. <b>Action:</b> Ensure that the address specified for ANSAREA is valid.

Table 49. Return and Reason Codes for the IXLCSP Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx080F	<b>Equate Symbol:</b> IXLRSNCODEBADAREALET <b>Meaning:</b> Program error. The ANSAREA ALET is not valid. <b>Action:</b> Verify that the ANSAREA ALET is valid.
8	xxxx081B	<b>Equate Symbol:</b> IXLRSNCODENOLENTRIES <b>Meaning:</b> Program error. The number of lock table entries specified was 0 for a size computation for a lock structure with record data, or was negative. <b>Action:</b> Verify that the number of lock table entries specified is correctly defined.
8	xxxx081C	<b>Equate Symbol:</b> IXLRSNCODENOLISTHDRS <b>Meaning:</b> Program error. The number of list headers specified on list structure computations must be greater than 0. <b>Action:</b> Verify that the number of list headers specified is correctly defined.
8	xxxx081D	<b>Equate Symbol:</b> IXLRSNCODEZEROLUSERS <b>Meaning:</b> Program error. The number of users specified on lock structure computations must be greater than 0. <b>Action:</b> Verify that the number of users specified is correctly defined.
8	xxxx0824	<b>Equate Symbol:</b> IXLRSNCODEWRONGSTRTYPE <b>Meaning and Action:</b> Incorrect structure type specified.
8	xxxx085C	<b>Equate Symbol:</b> IXLRSNCODEINVALIDLISTATTR <b>Meaning:</b> Program error. List structure computations must specify one of the following: lock entries, data elements, or adjunct. None was specified. <b>Action:</b> Verify that a required list structure attribute is specified.
8	xxxx085D	<b>Equate Symbol:</b> IXLRSNCODEINVALIDSTGCLASS <b>Meaning:</b> Program error. The value of NUMSTGCLASS cannot be 0. <b>Action:</b> Verify that you have specified a correct value for NUMSTGCLASS.
8	xxxx085E	<b>Equate Symbol:</b> IXLRSNCODEINVALIDCOCLASS <b>Meaning:</b> Program error. The value of NUMCOCLASS cannot be 0. <b>Action:</b> Verify that you have specified a correct value for NUMCOCLASS.

Table 49. Return and Reason Codes for the IXLCS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0860	<p><b>Equate Symbol:</b> IXLRSNCODEDIRRATIO</p> <p><b>Meaning:</b> Program error. The value of DIRRATIO or DIRENTRYCOUNT cannot be 0. Directory entries are required for a cache structure.</p> <p><b>Action:</b> Verify that you have specified a correct value for DIRRATIO or DIRENTRYCOUNT.</p>
8	xxxx0861	<p><b>Equate Symbol:</b> IXLRSNCODEENTRYRATIO</p> <p><b>Meaning:</b> Program error. The value of ENTRYRATIO cannot be 0 when ELEMENTRATIO is greater than 0. ENTRYCOUNT cannot be 0 when either ELEMENTCOUNT is greater than 0 or ADJUNCT=YES. SCMENTRYCOUNT cannot be 0 when either SCMELEMENTCOUNT is greater than 0 or ADJUNCT=YES. Entries are required for a list structure with data.</p> <p><b>Action:</b> Verify that you have specified a correct value for ENTRYRATIO, ENTRYCOUNT, or SCMENTRYCOUNT.</p>
8	xxxx0862	<p><b>Equate Symbol:</b> IXLRSNCODEMAXELEMNUM</p> <p><b>Meaning:</b> Program error.</p> <ul style="list-style-type: none"> <li>For a list structure, MAXELEMNUM must be greater than or equal to ELEMENTRATIO divided by ENTRYRATIO if ELEMENTRATIO is not 0.</li> <li>For a cache structure, MAXELEMNUM must be greater than or equal to ELEMENTRATIO divided by DIRRATIO if ELEMENTRATIO is not 0.</li> </ul> <p>The value of MAXELEMNUM cannot be 0.</p> <p><b>Action:</b> Verify that you have specified a correct value for MAXELEMNUM.</p>
8	xxxx086B	<p><b>Equate Symbol:</b> IXLRSNCODEELEMINCRNUM</p> <p><b>Meaning:</b> Program error. The value specified for ELEMINCRNUM is not valid. It must be non-zero and be a power of 2.</p> <p><b>Action:</b> Verify that you have specified a correct value for ELEMINCRNUM.</p>
8	xxxx0871	<p><b>Equate Symbol:</b> IXLRSNCODEMAXELEMNUMELEMCHAR</p> <p><b>Meaning:</b> Program error. The values specified in MAXELEMNUM and either ELEMCHAR or ELEMINCRNUM would result in entries of size greater than 64K.</p> <p><b>Action:</b> Verify that you have specified correct values for MAXELEMNUM and ELEMCHAR or ELEMINCRNUM.</p>

Table 49. Return and Reason Codes for the IXLCSP Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action						
8	xxxx0881	<p><b>Equate Symbol:</b> IXLRNCODEBADCFLEVEL</p> <p><b>Meaning:</b> Program error. Parameters are not appropriate for the CFLEVEL supported by the designated coupling facility. The field at offset SCPA_DiagnosticCode in the answer area designated by ANSAREA further describes the problem.</p> <table border="1"> <thead> <tr> <th>Diag Code</th><th>Specified Parameter</th><th>Minimum CFLEVEL</th></tr> </thead> <tbody> <tr> <td>5</td><td>KEYTYPE=SECONDARY</td><td>9</td></tr> </tbody> </table> <p><b>Action:</b> Correct the parameter value in error.</p>	Diag Code	Specified Parameter	Minimum CFLEVEL	5	KEYTYPE=SECONDARY	9
Diag Code	Specified Parameter	Minimum CFLEVEL						
5	KEYTYPE=SECONDARY	9						
8	xxxx0882	<p><b>Equate Symbol:</b> IXLRNCODEBADREFOPTION</p> <p><b>Meaning:</b> Program error. The request is rejected because the specified parameter is not appropriate for the value specified for REFOPTION.</p> <p><b>For PARAMETER REFOPTION must be</b>  <b>EMCSTGPCT&gt;0</b>  KEY</p> <p><b>Action:</b> Correct the parameter value in error.</p>						
8	xxxx0883	<p><b>Equate Symbol:</b> IXLRNCODEBADEMCSTGPCT</p> <p><b>Meaning:</b> Program error. The value specified for EMCSTGPCT is not valid. The value must be between 0 and 10000.</p> <p><b>Action:</b> Correct the value of EMCSTGPCT.</p>						
8	xxxx0886	<p><b>Equate Symbol:</b> IXLRNCODEBADREQUEST</p> <p><b>Meaning and Action:</b> Incorrect SERVICE or REQUEST value specified.</p>						
8	xxxx0888	<p><b>Equate Symbol:</b> IXLRNCODEBADSTRUCTURESIZE</p> <p><b>Meaning:</b> Program error. The request is rejected because the specified structure size is greater than the specified maximum structure size, or smaller than the calculated marginal structure size.</p> <p><b>Action:</b> Correct the value of STRSIZE.</p>						
8	xxxx0889	<p><b>Equate Symbol:</b> IXLRNCODECALCULATIONOVERFLOW</p> <p><b>Meaning:</b> Program error. The request is rejected because the calculation of the structure size or counts encountered an arithmetic overflow condition.</p> <p><b>Action:</b> Verify that all values specified are correct.</p>						

Table 49. Return and Reason Codes for the IXLCSP Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action																		
8	xxxx088A	<p><b>Equate Symbol:</b> IXLRSNCODEBADASCMODE</p> <p><b>Meaning:</b> Program error. The request is rejected because the caller is not in primary or AR mode.</p> <p><b>Action:</b> Ensure that the caller is running in primary or AR ASC mode.</p>																		
8	xxxx088B	<p><b>Equate Symbol:</b> IXLRSNCODEBADELEMCHARORINCRNUM</p> <p><b>Meaning:</b> Program error. The request is rejected because the caller's ELEMCHAR or ELEMINCRNUM specification exceeds the maximum data element size of the input coupling facility.</p> <p><b>Action:</b> Correct the value of either ELEMCHAR or ELEMINCRNUM and resubmit the job.</p>																		
8	xxxx088C	<p><b>Equate Symbol:</b> IXLRSNCODECOMPUTEREJECTED</p> <p><b>Meaning:</b> Program error. The request is rejected. The request could not be processed due to input that is not valid. The field at offset CSPA_DiagnosticCode in the answer area designated by ANSAREA further describes the problem. Applicable diagnostic codes are defined in IXLYCSPA.</p> <p><b>Action:</b> Examine the CSPA_DiagnosticCode field to identify the input that is not valid.</p>																		
8	xxxx088E	<p><b>Equate Symbol:</b> IXLRSNCODEINCONSISTENTPARM</p> <p><b>Meaning:</b> Program error. Request rejected. A keyword specification was made that also requires one or more other keywords to be specified. The field at offset CSPA_DiagnosticCode in the answer area designated by ANSAREA further describes the problem.</p> <table border="1"> <thead> <tr> <th>Diag Code</th><th>Specified Parameter</th><th>Also Required</th></tr> </thead> <tbody> <tr> <td>1</td><td>KEYTYPE=SECONDARY</td><td>ADJUNCT=YES</td></tr> <tr> <td>2</td><td>LISTCNTLTTYPE=ELEMENT</td><td>data elements</td></tr> <tr> <td>5</td><td>SCMALGORITHM=KEYPRIORITY1</td><td>Data elements</td></tr> <tr> <td>6</td><td>SCMALGORITHM=KEYPRIORITY1</td><td>REFOPTION=KEY</td></tr> <tr> <td>7</td><td>SCMALGORITHM=KEYPRIORITY1</td><td>Lists</td></tr> </tbody> </table> <p><b>Action:</b> Correct the parameter value in error.</p>	Diag Code	Specified Parameter	Also Required	1	KEYTYPE=SECONDARY	ADJUNCT=YES	2	LISTCNTLTTYPE=ELEMENT	data elements	5	SCMALGORITHM=KEYPRIORITY1	Data elements	6	SCMALGORITHM=KEYPRIORITY1	REFOPTION=KEY	7	SCMALGORITHM=KEYPRIORITY1	Lists
Diag Code	Specified Parameter	Also Required																		
1	KEYTYPE=SECONDARY	ADJUNCT=YES																		
2	LISTCNTLTTYPE=ELEMENT	data elements																		
5	SCMALGORITHM=KEYPRIORITY1	Data elements																		
6	SCMALGORITHM=KEYPRIORITY1	REFOPTION=KEY																		
7	SCMALGORITHM=KEYPRIORITY1	Lists																		
8	xxxx08B0	<p><b>Equate Symbol:</b> IXLRSNCODEBADSCMALGORITHM</p> <p><b>Meaning:</b> Program error. The value specified by the SCMALGORITHM keyword is not valid.</p> <p><b>Action:</b> Verify that you have specified a correct value for SCMALGORITHM.</p>																		



Table 49. Return and Reason Codes for the IXLCSP Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action						
8	xxxx08B1	<p><b>Equate Symbol:</b> IXLRSNCODEVALUEOUTOFRANGE</p> <p><b>Meaning:</b> Program error. The value that is specified by an input parameter is out of range; either too low or too high. The CSPA_DIAGNOSTICCODE field of the IXLYCSPA answer area identifies the parameter that was out of bounds.</p> <p><b>Action:</b> Ensure that you specify a value within the acceptable range for the affected parameter.</p>						
8	xxxx08BB	<p><b>Equate Symbol:</b> IXLRSNCODEMINIMUMCOUNT</p> <p><b>Meaning:</b> Program error. Computation would result in counts smaller than the minimum entry count or the minimum element count. Applicable only when the computation request specifies SCMMAXSIZE &gt; 0.</p> <p><b>Action:</b> When computing by counts, specify entry and element counts greater than or equal to the minimum amount that is required for use by SCM (20000x). When computing by size, specify a size that is sufficient to produce at least the minimum number of entries and elements.</p>						
8	xxxx08BE	<p><b>Equate Symbol:</b> IXLRSNCODEUNSUPPORTEDCFLEVEL</p> <p><b>Meaning and Action:</b> The request specified a CFLEVEL value not available to this system, because there is no connected CF that provides the requested level and the z/OS system from which the request was submitted does not support emulation of the requested level in software. Answer area mapping CSPA_CFLevelInfo describes the set of supported CFLEVELs.</p>						
C	xxxx0C06	<p><b>Equate Symbol:</b> IXLRSNCODENOCONECT</p> <p><b>Meaning:</b> Environmental error. The system from which the request is issued lost connectivity to the coupling facility named by the CFNAME after initial validation of the coupling facility.</p> <p><b>Action:</b> Try the request again after connectivity has been restored, or specify a different coupling facility.</p>						
C	xxxx0C68	<p><b>Equate Symbol:</b> IXLRSNCODEBADREQCFLEVEL</p> <p><b>Meaning:</b> Environmental error. The CFLEVEL of the coupling facility that is specified by the CFNAME keyword is insufficient to process the request.</p> <table><tr><td>Specified Request</td><td>Minimum CFLevel</td></tr><tr><td>Any</td><td>8</td></tr><tr><td>ComputeSCMSize</td><td>19</td></tr></table> <p><b>Action:</b> Try the request again and specify a coupling facility with a CFLEVEL that is sufficient for the requested calculation.</p>	Specified Request	Minimum CFLevel	Any	8	ComputeSCMSize	19
Specified Request	Minimum CFLevel							
Any	8							
ComputeSCMSize	19							

Table 49. Return and Reason Codes for the IXL CSP Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C98	<p><b>Equate Symbol:</b> ISLRSNCODECFNOTACCESSIBLE</p> <p><b>Meaning:</b> Environmental error. The coupling facility specified by the CFNAME keyword cannot be accessed from this system. Possible causes include:</p> <ul style="list-style-type: none"> <li>• The coupling facility is not described by the active CFRM policy.</li> <li>• There is no CFRM couple data set.</li> <li>• The system from which the request is issued does not have connectivity to the coupling facility.</li> <li>• The coupling facility has failed.</li> </ul> <p><b>Action:</b> Investigate the possible causes why the coupling facility cannot be accessed, correct the problem, and resubmit the request.</p>
C	FFFFFFFF	<p><b>Meaning:</b> XES functions are not available. This can occur because the coupling facility hardware necessary to provide XES function is not present.</p> <p><b>Action:</b> Re-IPL the system, or follow your particular management protocol.</p>
10	xxxx10xx	<p><b>Meaning:</b> Failure in XES processing. The state of the resource request is unpredictable.</p> <p><b>Action:</b> Save the reason code information and contact the IBM support center.</p>

## Chapter 52. IXLDISC – Disconnecting from a Coupling Facility Structure

### Description

Use the IXLDISC macro to disconnect from a coupling facility structure. You should specify a reason for disconnecting. If the disconnection is normal, specify REASON=NORMAL. If the disconnection is an attempt to delete the structure, specify REASON=DELETESTR. If the disconnection is part of an error routine, issue REASON=FAILURE. For disconnections that specify REASON=FAILURE, if the connection disposition is KEEP, the connection is placed in a failed-persistent state after all surviving peer connections acknowledge the disconnect. The connection remains defined to the coupling facility.

In order to disconnect from a structure, the requestor must specify the CONTOKEN from the IXLCONN request issued for the structure. The IXLDISC macro must also be issued from the same task that issued the IXLCONN macro for the structure. For more information on IXLDISC, see *z/OS MVS Programming: Sysplex Services Guide*.

### Environment

The following are the environment requirements for the caller.

Environment	Environment requirement
Authorization:	Supervisor state or PKM keys 0 - 7
Dispatchable unit mode:	Task
Cross memory mode:	PASN=HASN, any SASN. The primary address space must be equal to the requestor's primary address space at the time of the connection.
AMODE:	31-bit
ASC mode:	Primary or access register (AR)
Interrupt status:	Enabled for I/O and external interrupts
Locks:	No locks held, with no enabled, unlocked task (EUT) FRRs established
Control parameters:	Must be in the primary address space or be in an address/data space that is addressable through a public entry on the caller's dispatchable unit access list (DU-AL)

### Restrictions

The IXLDISC macro must be issued from the same task that issued the IXLCONN macro for the structure.

If disconnecting during the rebuild process, issue IXLDISC under the same task that issued the original IXLCONN request and specify the original contoken.

Note that if you disconnect from a lock structure with locks held, the system treats it as disconnecting with REASON=FAILURE.

## Input Register Information

---

Before issuing the IXLDISC macro, the caller does not have to place any information into any register unless using it in register notation for a particular parameter, or using it as a base register.

## Output Register Information

---

When control returns to the caller of the IXLDISC macro, the general purpose registers (GPRs) contain:

### Register Contents

**0**

Reason code, if applicable, if GPR 15 return code is nonzero

**1**

Used as work register by the system

**2-13**

Unchanged

**14**

Used as work register by the system

**15**

Return code

When control returns to the caller of the IXLDISC macro, the access registers (ARs) contain:

### Register Contents

**0-1**

Used as work registers by the system

**2-13**

Unchanged

**14-15**

Used as work registers by the system

**For registers that the system changes,** a caller who depends on these registers containing the same value before and after issuing the macro must save these registers before issuing the macro and restore them after the system returns control.

## Programming Requirements

---

If the program is in AR mode, issue the SYSSTATE ASCENV=AR macro before IXLDISC. SYSSTATE ASCENV=AR tells the system to generate code appropriate for AR mode.

## Performance Implications

---

After IXLDISC is issued, all peer connections are notified of the disconnect/failure event. The performance of IXLDISC depends on recovery and cleanup protocols of the surviving connections.

IXLDISC processing might involve referencing or updating the CFRM active policy to reflect the operation that has been requested. When invoking this macro, be aware of the I/O to the CFRM couple data set that might be generated.

## Understanding IXLDISC Version Support

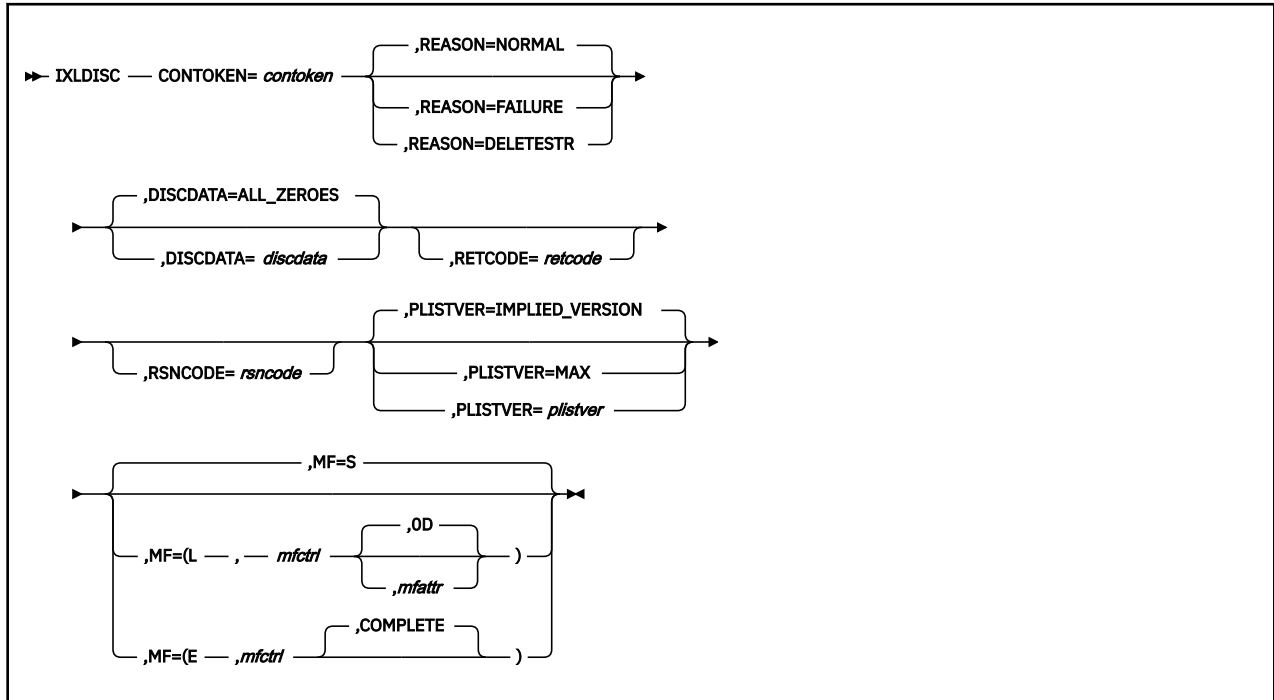
---

The IXLDISC macro supports version 0 keywords and functions.

Specify the version of the parameter list that you want generated with the PLISTVER keyword. See Chapter 2, “Specifying a Macro Version Number,” on page 5 for considerations when specifying the version of the parameter list with PLISTVER.

## Syntax Diagram

The syntax of the IXLDISC macro is as follows:



## Parameter Descriptions

The parameter descriptions are listed in alphabetical order. Default values are underlined:

### **CONTOKEN=**contoken

Use this input parameter to specify the connection identifier that was returned by the IXLCONN service. The connection identifier uniquely identifies the user's connection to the structure.

During rebuild processing, use the original, not the temporary, contoken.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the 16-character field that contains the connect token.

### **,DISCDATA=**ALL\_ZEROES

### **,DISCDATA=**discdata

Use this input parameter to specify eight bytes of connector data that the system will pass to the event exit of other connectors when you disconnect from the structure.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the eight-character field.

**,MF=S**  
**,MF=(L,mfctrl)**  
**,MF=(L,mfctrl,mfattr)**  
**,MF=(L,mfctrl,0D)**  
**,MF=(M,mfctrl)**  
**,MF=(M,mfctrl,COMPLETE)**  
**,MF=(M,mfctrl,NOCHECK)**  
**,MF=(E,mfctrl)**  
**,MF=(E,mfctrl,COMPLETE)**  
**,MF=(E,mfctrl,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

**,mfctrl**

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

**,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

**,COMPLETE**

**,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

**COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if SMILE=*var* were an optional parameter and the default is SMILE=NO\_SMILE then it would not be documented. However, if the default was SMILE=:), then it would be documented because a value would be the default.

**NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

**,PLISTVER=IMPLIED\_VERSION**

**,PLISTVER=MAX**

**,PLISTVER=plistver**

Use this input parameter to specify the version of the macro. See [“Understanding IXLDISC Version Support”](#) on page 924 for a description of the options available with PLISTVER.

**,REASON=**NORMAL

**,REASON=**FAILURE

**,REASON=**DELETESTR

Use this input parameter to specify the reason for the disconnection: The disconnect reason is presented to all surviving connectors' event exits.

#### **NORMAL**

Indicates normal termination. The connection disposition specified on IXLCONN does not apply and the connection is made not-defined. If the disconnection results in there being no connections (active or failed-persistent) to the structure, the structure disposition applies. The structure disposition determines if the structure is deleted (STRDISP=DELETE) or retained (STRDISP=KEEP).

If a connection to a lock structure with RECORD=YES owns lock resources and issues IXLDISC REASON=NORMAL, the disconnect is treated as if the user specified REASON=FAILURE.

#### **FAILURE**

Indicates termination due to a failure. The connection disposition specified on IXLCONN applies. After all surviving connections are notified and confirm the disconnect, the connection is made not-defined (CONDISP=DELETE) or failed-persistent (CONDISP=KEEP). If the disconnection results in there being no connections (active or failed-persistent) to the structure, the structure disposition applies. The structure disposition determines if the structure is deleted (STRDISP=DELETE) or retained (STRDISP=KEEP).

#### **DELETESTR**

Indicates normal termination. The connection disposition specified on IXLCONN does not apply and the connection is made not-defined. If the disconnection results in there being no connections (active or failed-persistent) to the structure, the structure is deleted even if the structure has a disposition of KEEP. This keyword allows a structure with a disposition of KEEP to be deleted by the last connector when it is known that the structure is no longer needed. Without this keyword, the program would need to issue IXLFORCE to delete such a structure, or the installation would need to manually delete the structure (there are also situations that might cause XES to implicitly delete a structure with no connectors).

If a connection to a lock structure with RECORD=YES owns lock resources, and issues IXLDISC REASON=DELETESTR, the disconnect is treated as if the user specified REASON=FAILURE. In particular, note that if the structure has a disposition of KEEP, the structure will persist even if it has no connectors.

**,RETCODE=***retcode*

Use this output parameter to specify the location in which the system is to copy the return code from GPR 15.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword to contain the return code.

**,RSNCODE=***rsncode*

Use this output parameter to specify the location in which the system is to copy the reason code from GPR 0.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword to contain the reason code.

## Return and Reason Codes

---

When the IXLDISC macro returns control to your program:

- GPR 15 (and *retcode*, if you coded RETCODE) contains a return code.
- When the value in GPR 15 is not zero, if applicable, GPR 0 (and *rsncode*, if you coded RSNCODE) contains a reason code.

The IXLYCON macro provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

**0**

IXLRETCODEOK

**4**

IXLRETCODEWARNING

**8**

IXLRETCODEPARMERROR

**C**

IXLRETCODEENVERROR

**10**

IXLRETCODECOMPERROR

The following table identifies the hexadecimal return and reason codes and the equate symbol associated with each reason code.

Table 50. Return and Reason Codes for the IXLDISC Macro		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0	None	<b>Meaning:</b> Disconnection is successful. The connect token is now invalid and will be rejected on any subsequent requests. <b>Action:</b> None.
4	xxxx0401	<b>Equate Symbol:</b> IXLRSNCODEOWNINGRESOURCES <b>Meaning:</b> Disconnection is successful. Requestor owned the resources in the lock structure from which the requestor has disconnected. The disconnect will be treated as abnormal regardless of whether the connector specified REASON=NORMAL. If the connection is CONDISP=KEEP, the connector is made failed-persistent after all confirmations are received. <b>Action:</b> The system invalidates the connect token of the requestor.
8	xxxx0801	<b>Equate Symbol:</b> IXLRSNCODEBADPARMLIST <b>Meaning:</b> Program error. The IXLDISC parameter list is not accessible. <b>Action:</b> Verify that: <ul style="list-style-type: none"> <li>• The parameter list address is uncorrupted</li> <li>• The parameter list is addressable in the caller's primary address space.</li> <li>• If you are calling IXLDISC in AR-mode and you specified the parameter list address using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If you are calling IXLDISC in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLDISC.</li> </ul>



Table 50. Return and Reason Codes for the IXLDISC Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0802	<p><b>Equate Symbol:</b> IXLSRNCODEBADPARMLISTALET</p> <p><b>Meaning:</b> Program error. The IXLDISC parameter list ALET is not valid.</p> <p><b>Action:</b> Verify that:</p> <ul style="list-style-type: none"> <li>• If you are calling IXLDISC in AR-mode and you specified the parameter list address using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If you are calling IXLDISC in AR-mode, SYSSTATE ASCENV=AR must be issued before issuing the IXLDISC.</li> <li>• If you are not running in AR-mode, and the parameter list does not need to be ALET-qualified, the corresponding access register should contain zeros.</li> </ul>
8	xxxx0804	<p><b>Equate Symbol:</b> IXLSRNCODEBADVERSION#</p> <p><b>Meaning:</b> Program error. The IXLDISC parameter list contains an invalid version number</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that your program did not overlay the parameter list storage.</li> <li>• Verify that your program was assembled with the correct macro library for the release of MVS your program is running on.</li> </ul>
8	xxxx0805	<p><b>Equate Symbol:</b> IXLSRNCODEBADTCB</p> <p><b>Meaning:</b> Program error. The task that issued the IXLDISC macro is different from the task that issued the corresponding IXLCONN macro.</p> <p><b>Action:</b> Set up your program so that it does the disconnect under the same TCB that the connect was done under.</p>
8	xxxx0806	<p><b>Equate Symbol:</b> IXLSRNCODESRBMODE</p> <p><b>Meaning:</b> Program error. The requestor is in SRB mode.</p> <p><b>Action:</b> You must be running in task mode under the task that did the connect to issue IXLDISC.</p>
8	xxxx0807	<p><b>Equate Symbol:</b> IXLSRNCODENOTENABLED</p> <p><b>Meaning:</b> Program error. The requestor is not in an enabled state.</p> <p><b>Action:</b> The requestor must be enabled for I/O and external interrupts.</p>
8	xxxx0809	<p><b>Equate Symbol:</b> IXLSRNCODEPRIMARYNOTHOME</p> <p><b>Meaning:</b> Program error. The requestor's primary address space is not the same as the home address space.</p> <p><b>Action:</b> The primary address space must equal the home address space to issue the IXLDISC. The primary address space must be the same address space the IXLCONN was issued from.</p>

Table 50. Return and Reason Codes for the IXLDISC Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx080A	<p><b>Equate Symbol:</b> IXLRSNCODEBADCONTOKEN</p> <p><b>Meaning:</b> Program error. The requestor specified a CONTOKEN that is not valid, or the request was issued from an address space other than the connector's address space.</p> <p><b>Action:</b> Verify the CONTOKEN specified was the original one received in the IXLCONN answer area after the connect was issued. The disconnect must be issued from the same address space as the connect.</p>
C	FFFFFFFF	<p><b>Equate Symbol:</b> IXLRSNCODENOTAVAILABLE</p> <p><b>Meaning:</b> Environmental error. There are no coupling facility services available. The hardware support necessary to provide coupling facility services might not be present.</p> <p><b>Action:</b> Re-IPL the system, or follow your particular failure management protocol.</p>
10	—	<p><b>Meaning:</b> XES processing has failed.</p> <p><b>Action:</b> Save the reason code information and contact the IBM support center.</p>

## Chapter 53. IXLEERSP – Responding to an Event

### Description

Use the IXLEERSP macro if you are an active connection to a structure in the coupling facility and must provide an asynchronous response to an event exit event (at a time other than when the event exit gets control). Use IXLEERSP to respond to the following specific events reported to your event exit:

- Disconnected or Failed Connection event
- Existing Connection event for a failed-persistent connection
- Rebuild Quiesce event to start rebuilding or duplexing a structure
- Rebuild Cleanup event to indicate that resource clean-up for rebuilding or duplexing a structure is complete
- Rebuild Stop event to confirm that the system stop the rebuilding or duplexing process
- Rebuild Connection Failure event to confirm the failure of a rebuild connection
- Structure Temporarily Unavailable event to acknowledge that a system-managed process has been initiated.

For EVENT=DISCFAILCONN, the requestor is able to indicate, using RELEASECONN=YES, that all needed recovery for the failed connector has been performed, and that, as a result, the failed connector need not be made failed-persistent when all confirmations are received. Using PROXYRESPONSE=YES, the requestor is also able to provide a response on behalf of a peer connector that is the subject of a Disconnected or Failed Connection event. The events for which the response can be provided are the Rebuild Cleanup and Rebuild Stop events.

Before using IXLEERSP, the requestor must have set a return code X'08' in the event exit parameter list (IXLYEEPL). See field IXLRCVENTEXITLATERESPONSE in IXLYCON. Setting the return code indicates, at the time the event is presented to the event exit, that the requestor intends to issue IXLEERSP to respond to the event. If the connector intends to respond to the event asynchronously, the connector must use the IXLEERSP macro.

### Environment

The following are the environment requirements for the caller.

Environment	Environment requirement
Authorization:	Supervisor state or PKM keys 0 - 7
Dispatchable unit mode:	Task
Cross memory mode:	PASN=HASN, any SASN, The primary address space must be equal to the requestor's primary address space at the time of the connection.
AMODE:	31-bit
ASC mode:	Primary or access register (AR)
Interrupt status:	Enabled for I/O and external interrupts
Locks:	No locks held, with no enabled, unlocked task (EUT) FRRs established
Control parameters:	Must be in the primary address space or be in an address/data space that is addressable through a public entry on the caller's dispatchable unit access list (DU-AL)

## Restrictions

---

The connector must set return code X'08' in IXLVEEPL within the event exit to indicate that IXLEERSP is to provide the response to the event.

## Input Register Information

---

Before issuing the IXLEERSP macro, the caller does not have to place any information into any register unless using it in register notation for a particular parameter, or using it as a base register.

## Output Register Information

---

When control returns to the caller of the IXLEERSP macro, the general purpose registers (GPRs) contain:

### Register

#### Contents

**0**

Reason code, if applicable, if GPR 15 return code is non-zero.

**1**

Used as work register by the system

**2-13**

Unchanged

**14**

Used as a work register by the system

**15**

Return code

When control returns to the caller of the IXLEERSP macro, the access registers (ARs) contain:

### Register

#### Contents

**0-1**

Used as work registers by the system

**2-13**

Unchanged

**14-15**

Used as work registers by the system

**For registers that the system changes,** a caller who depends on these registers containing the same value before and after issuing the macro must save these registers before issuing the macro and restore them after the system returns control.

## Programming Requirements

---

If the program is in AR mode, issue the SYSSTATE ASCENV=AR macro before IXLEERSP. SYSSTATE ASCENV=AR tells the system to generate code appropriate for AR mode.

## Performance Implications

---

IXLEERSP processing might involve referencing or updating the CFRM active policy to reflect the operation that has been requested. When invoking this macro, be aware of the I/O to the CFRM couple data set that might be generated.

## Understanding IXLEERSP Version Support

---

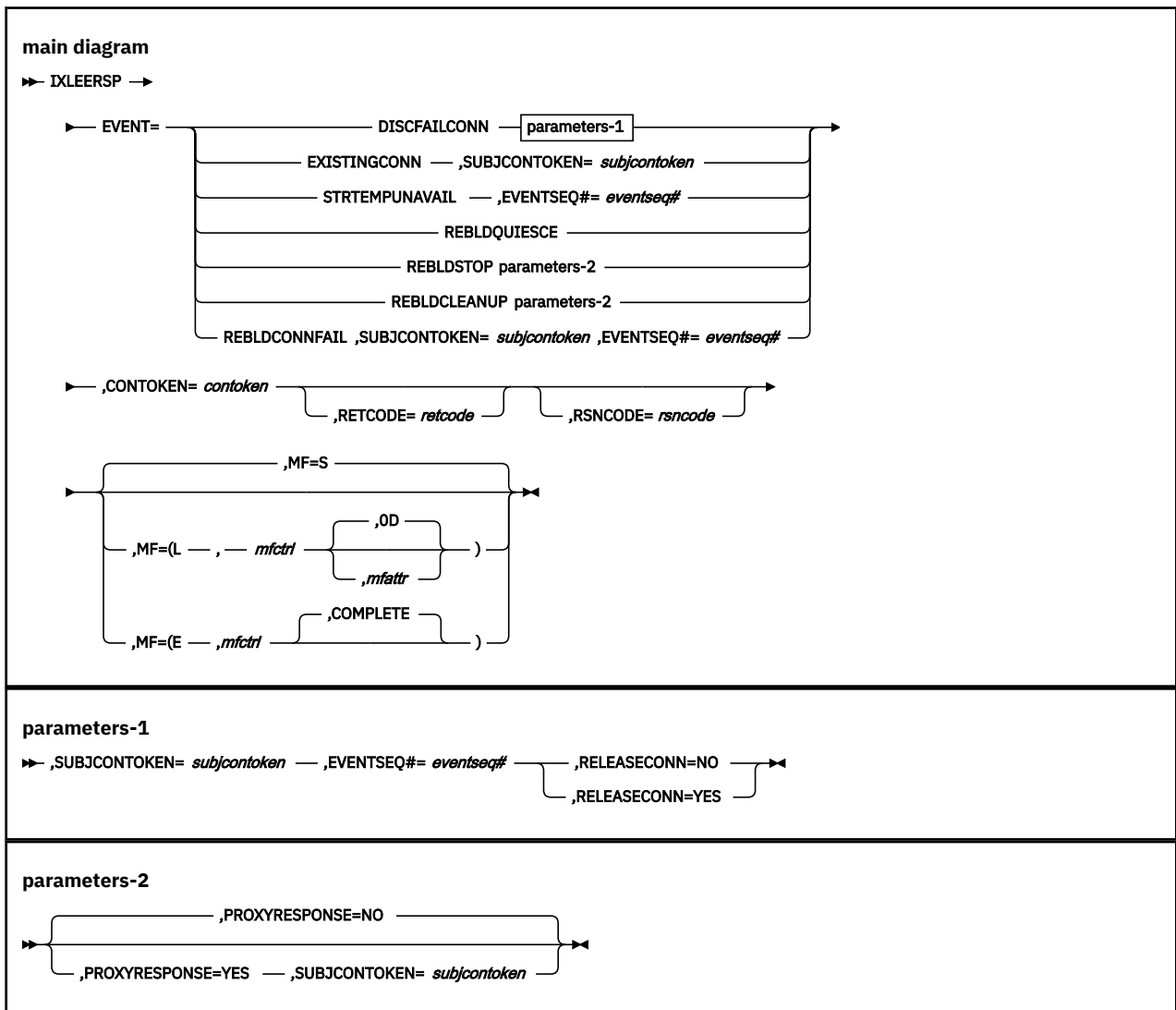
The IXLEERSP macro supports versions in the range of 0 - 2.

- Keywords not specifically noted here are supported by version 0 and subsequent versions of the IXLEERSP macro.
- The following keyword is supported by version 1 and subsequent versions of the IXLEERSP macro.
  - PROXYRESPONSE
- The following event type is supported by version 2 and subsequent versions of the IXLEERSP macro.
  - STRTEMPUNAVAIL

Specify the version of the parameter list that you want generated with the PLISTVER keyword. See Chapter 2, “Specifying a Macro Version Number,” on page 5 for considerations when specifying the version of the parameter list with PLISTVER.

## Syntax Diagram

The syntax of the IXLEERSP macro is written as follows:



## Parameter Descriptions

The parameter descriptions are listed in alphabetical order. Default values are underlined>.

### **,CONTOKEN=contoken**

Use this input parameter to specify the responding connection's connect token. CONTOKEN is returned to the connect answer area when the connector issues the IXLCONN macro for the structure.

Note that during rebuild, you must specify the original contoken, not the temporary one.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the 16-character field that contains the requestor's connect token.

**EVENT=DISCFAILCONN**  
**EVENT=EXISTINGCONN**  
**EVENT=REBLDQUIESCE**  
**EVENT=REBLDSTOP**  
**EVENT=REBLDCLEANUP**  
**EVENT=REBLDCONNFAIL**  
**EVENT=STRTEMPUNAVAIL**

Use this input parameter to specify the connection event to which the requestor is responding:

#### **DISCFAILCONN**

Indicates the Disconnected or Failed Connection event (equal to field EEPLDISCFAILCONNECTION in the IXLYEEPL parameter list). The SUBJCONTOKEN keyword is required to identify the connection that is the subject of the event. The RELEASECONN keyword is required. It indicates whether the connection should remain persistent or be released.

#### **EXISTINGCONN**

Indicates an Existing Connection event for a connection that is failed-persistent (equal to field EEPEXISTINGCONNECTION in the IXLYEEPL parameter list). This event is reported to new connections only. Specifying EVENT=EXISTINGCONN deletes the failed-persistent connection from the structure, and the existing connection is no longer defined to the coupling facility. The SUBJCONTOKEN keyword is required to identify the connection that is the subject of the event.

#### **REBLDQUIESCE**

Indicates the Rebuild Quiesce event to rebuild or duplex a structure (equal to field EEPLREBUILDQUIESCE in the IXLYEEPL parameter list). Rebuild is initiated through the SETXCF START,REBUILD command or the IXLREBLD REQUEST=START request. Duplexing is initiated through the SETXCF START,REBUILD,DUPLEX command or the IXLREBLD REQUEST=STARTDUPLEX request. (See Chapter 81, “IXLREBLD — Rebuilding or duplexing a structure,” on page 1539.) Users must respond to the Rebuild Quiesce event. Specifying EVENT=REBLDQUIESCE indicates that the requestor participating in the user-managed process to rebuild or duplex the structure has quiesced all activity to the structure including the completion of processing based on the use of restart tokens.

Note that if you do not provide a REBLDQUIESCE response, you must either stop the rebuild or duplex process or disconnect from the structure.

#### **REBLDSTOP**

Indicates the Rebuild Stop event to stop rebuilding or duplexing a structure (equal to field EEPLREBLDSTOP in the IXLYEEPL parameter list). Stop rebuilding is initiated through the SETXCF STOP,REBUILD command or the IXLREBLD REQUEST=STOP request. Stop duplexing is initiated through the SETXCF STOP,REBUILD,DUPLEX command or the IXLREBLD REQUEST=STOPDUPLEX request. (See Chapter 81, “IXLREBLD — Rebuilding or duplexing a structure,” on page 1539.) Users must respond to the Rebuild Stop event. Specifying EVENT=REBLDSTOP indicates that the requestor is participating in stopping the rebuild or duplexing process.

#### **REBLDCLEANUP**

Indicates the Rebuild Cleanup event (equal to field EEPLREBLDCLEANUP in the IXLYEEPL parameter list). When all connectors to a rebuilt structure issue IXLREBLD REQUEST=COMPLETE to indicate that they have completed processing, the system reports the Rebuild Cleanup event to the event exit. When all connectors to a duplexed structure issue IXLREBLD REQUEST=DUPLEXCOMPLETE to indicate that they have completed processing, the system also reports the Rebuild Cleanup event to the event exit. Users must respond to this event to confirm that resource cleanup for rebuilding is complete. Specifying EVENT=REBLDCLEANUP indicates that the requestor has completed the rebuilding process and has no knowledge of the old structure or has completed the duplexing process and has no knowledge of the old structure.

**REBLDCONNFAIL**

Indicates that the connection identified by SUBJCONTOKEN failed in its connection to the new structure during rebuild processing. The SUBJCONTOKEN keyword is required to identify the connection that is the subject of the event.

**STRTEMPUNAVAIL**

Indicates that the structure is temporarily unavailable for coupling facility requests against it. The structure is undergoing a system-managed process, such as system-managed rebuild. Specifying EVENT=STRTEMPUNAVAIL indicates that the responder has completed preparations for the structure being unavailable for the duration of the system-managed process. The EVENTSEQ# is required to identify the specific Structure Temporarily Unavailable event for which the response applies.

**,EVENTSEQ#=eventseq#**

Use this input parameter to specify the event sequence number. The event sequence number for the event must be the same value that was presented to this connection in the event exit parameter list (IXLYEEPL).

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the fullword field that contains the event sequence number.

**,MF=S****,MF=(L,mfctrl)****,MF=(L,mfctrl,mfattr)****,MF=(L,mfctrl,0D)****,MF=(M,mfctrl)****,MF=(M,mfctrl,COMPLETE)****,MF=(M,mfctrl,NOCHECK)****,MF=(E,mfctrl)****,MF=(E,mfctrl,COMPLETE)****,MF=(E,mfctrl,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

**,mfctrl**

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

**,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

**,COMPLETE****,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

**COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if `SMILE=var` were an optional parameter and the default is `SMILE=NO_SMILE` then it would not be documented. However, if the default was `SMILE=-`, then it would be documented because a value would be the default.

**NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

**,PROXYRESPONSE=NO****,PROXYRESPONSE=YES**

Use this input parameter to indicate whether this response is being provided on behalf of a failing connector.

**NO**

Indicates that this is not a proxy response. The response is on behalf of the connector described by the `CONTOKEN` keyword.

**YES**

Indicates that this is a proxy response. The response is being provided on behalf of the connector described by the `SUBJCONTOKEN` keyword.

**Note:** To ensure that the `PROXYRESPONSE` feature is installed on the system on which you are running, issue `IXCQUERY REQINFO=FEATURES`. `QUREQRFPROXYRESPONSE`, returned from the `IXCQUERY` request, indicates whether the `PROXYRESPONSE` feature is installed.

**,RELEASECONN=NO****,RELEASECONN=YES**

Use this input parameter to specify whether the connection should remain persistent or be released.

**NO**

Indicates that the requestor has completed processing for the failed connection event, and specifies that XES should continue processing for the failed connection. The persistence attribute of the connection is not affected by this response.

**YES**

Indicates that the requestor has completed processing for the failed connection event, specifies that XES should continue processing for the failed connection, and specifies that this connection is no longer required to be persistent.

**,RETCODE=retcode**

Use this output parameter to specify the location in which the system is to copy the return code from GPR 15.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword to contain the return code.

**,RSNCODE=rsncode**

Use this output parameter to specify the location in which the system is to copy the reason code from GPR 0.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword to contain the reason code.

**,SUBJCONTOKEN=subjcontoken**

Use this input parameter to specify the `CONTOKEN` for the connection described by the event. `SUBJCONTOKEN` must be the same value that was presented to the event exit in field `EEPLSUBJCONTOKEN` of the `IXLYEEPL`.

Do not provide a proxy response on behalf of a failing connector until AFTER the `EEPLDISCFAILCONNECTION` event for the subject user has been presented to your event exit. You



can use the value of the EEPLSUBJCONTOKEN field from that invocation of the event exit as input for this parameter.

Note that if the subject connector is not in the failing state, the IXLEERSP request will fail with the IXLRSCODESUBJCONNNOTFAILING reason code.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the 16-character field that contains the CONTOKEN for the connection described by the event.

## Return and Reason Codes

When the IXLEERSP macro returns control to your program:

- GPR 15 (and *retcode*, if you coded RETCODE) contains a return code.
- GPR 0 (and *rsncode*, if you coded RSNCODE) contains a reason code.

Macro IXLYCON provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

**0**

IXLRETCODEOK

**4**

IXLRETCODEWARNING

**8**

IXLRETCODEPARMERROR

**C**

IXLRETCODEENVERROR

**10**

IXLRETCODECOMPERROR

The following table identifies the hexadecimal return and reason codes and the equate symbol associated with each reason code.

Table 51. Return and Reason Codes for the IXLEERSP Macro		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0	None	<b>Meaning:</b> Processing for IXLEERSP has completed successfully. <b>Action:</b> None.
4	xxxx041D	<b>Equate Symbol:</b> IXLRSCODEIGNOREFORREBUILDSTOP <b>Meaning:</b> The response for the Rebuild Quiesce event was ignored because a Rebuild Stop is in progress. <b>Action:</b> You should have been notified of the Rebuild Stop event. You may have issued IXLEERSP for the wrong event. You may have issued IXLEERSP for the Rebuild Quiesce event more than once. You may have connected during the Rebuild Quiesce window and a Rebuild Stop has been processed already. If so, respond to the Rebuild Stop event. Check your protocol for the rebuild process.

Table 51. Return and Reason Codes for the IXLEERSP Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	xxxx0428	<p><b>Equate Symbol:</b> IXLRSNCODEIGNOREFORSYSGDSTOP</p> <p><b>Meaning:</b> The response for the Structure Temporarily Unavailable event was ignored because the system-managed rebuild process has been stopped.</p> <p><b>Action:</b> None.</p>
8	xxxx0801	<p><b>Equate Symbol:</b> IXLRSNCODEBADPARMLIST</p> <p><b>Meaning:</b> Program error. The parameter list for this request is not addressable. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Verify that:</p> <ul style="list-style-type: none"> <li>• The parameter list address is uncorrupted.</li> <li>• The parameter list is addressable in the caller's primary address space.</li> <li>• If you are issuing this macro while disabled, the parameter list resides in either page-fixed or DREF storage.</li> <li>• If you are issuing this macro in AR-mode and you specified the parameter list address using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If you are issuing this macro in AR-mode, SYSSTATE ASCENV=AR must be issued before issuing this macro.</li> </ul>
8	xxxx0802	<p><b>Equate Symbol:</b> IXLRSNCODEBADPARMLISTALET</p> <p><b>Meaning:</b> Program error. The IXLEERSP parameter list ALET is not valid.</p> <p><b>Action:</b> All parameters must be in the primary address space so all ALETs should be set up accordingly. Fix the ALET of the parameter list and re-issue the macro.</p>
8	xxxx0804	<p><b>Equate Symbol:</b> IXLRSNCODEBADVERSIONNUM (or IXLRSNCODEBADVERSION#)</p> <p><b>Meaning:</b> The version number in the macro parameter list is not compatible with the level of XES currently being used. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that your program did not overlay the parameter list storage.</li> <li>• Verify that your program was assembled with the correct macro library for the release of MVS on which your program is running.</li> </ul>
8	xxxx0806	<p><b>Equate Symbol:</b> IXLRSNCODESRBMODE</p> <p><b>Meaning:</b> Program error. The requestor is in SRB mode.</p> <p><b>Action:</b> You must be running in task mode to issue this macro.</p>

Table 51. Return and Reason Codes for the IXLEERSP Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0807	<p><b>Equate Symbol:</b> IXLRSNCODENOTENABLED</p> <p><b>Meaning:</b> Program error. The requestor is not in an enabled state.</p> <p><b>Action:</b> The user must be enabled for I/O and external interrupts to issue this macro.</p>
8	xxxx0809	<p><b>Equate Symbol:</b> IXLRSNCODEPRIMARYNOTHOME</p> <p><b>Meaning:</b> Program error. The primary address space does not equal the home address space.</p> <p><b>Action:</b> The primary address space must be equal to the primary address space when the IXLCONN was issued. The home address space must be equal to the primary address space.</p>
8	xxxx080A	<p><b>Equate Symbol:</b> IXLRSNCODEBADCONTOKEN</p> <p><b>Meaning:</b> Program error. The specified connect token (CONTOKEN) was not valid for one of the following reasons:</p> <ol style="list-style-type: none"> <li>1. The user with the connection identifier represented by the token has disconnected from the structure.</li> <li>2. The connector's task (the task that issued IXLCONN) ended.</li> <li>3. The specified token is not the token that was returned from IXLCONN.</li> <li>4. The request was issued from an address space other than the address space in which IXLCONN was issued.</li> <li>5. The connect token was invalidated during rebuild.</li> <li>6. The connect token was invalidated by XES.</li> </ol> <p><b>Note:</b> The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Take the action with the corresponding meaning.</p> <ol style="list-style-type: none"> <li>1. Discontinue use of the structure. Perform recovery and cleanup for the structure and check the protocol for the use of the structure to determine why you did not know of the disconnection.</li> <li>2. Discontinue use of the structure. Perform recovery and cleanup for the structure and check the protocol for the use of the structure to determine why you did not know that the task ended.</li> <li>3. The contoken is returned in the CONACONTOKEN field of the answer area specified on the IXLCONN request.</li> <li>4. Issue your request from the same address space the IXLCONN was issued in.</li> <li>5. Wait for the rebuild to complete, and try again.</li> <li>6. Discontinue use of the structure. Perform recovery and cleanup for the structure.</li> </ol>

Table 51. Return and Reason Codes for the IXLEERSP Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C13	<p><b>Equate Symbol:</b> IXLRSNCODEREQPURGED</p> <p><b>Meaning:</b> Environmental error. The request was purged prior to completion of the request. Possible reasons include:</p> <ul style="list-style-type: none"> <li>• The connector failed.</li> <li>• The connector disconnected.</li> <li>• The requestor failed.</li> <li>• The request was purged by IXLPURGE.</li> <li>• Requests were purged when the connector provided an IXLEERSP response for the Rebuild Stop or Rebuild Cleanup event.</li> <li>• The secondary address space was no longer valid.</li> </ul> <p><b>Action:</b> None if this is expected. Otherwise, determine why the connector failed.</p>
C	xxxx0C3F	<p><b>Equate Symbol:</b> IXLRSNCODECONNNOTDEFINED</p> <p><b>Meaning:</b> Environmental error. The CONTOKEN is for a connection that is in the not defined state.</p> <p><b>Action:</b> The connection issuing the IXLEERSP macro must be active. Verify the CONTOKEN passed. Check your protocol to determine why you are responding to this event.</p>
C	xxxx0C40	<p><b>Equate Symbol:</b> IXLRSNCODECONNNOTACTIVE</p> <p><b>Meaning:</b> Environmental error. The CONTOKEN is for a connection that is not in the active state.</p> <p><b>Action:</b> The connection issuing the IXLEERSP macro must be in the active state. Verify the CONTOKEN passed. Check your protocol to determine why you are responding to this event.</p>
C	xxxx0C41	<p><b>Equate Symbol:</b> IXLRSNCODEUNEXPECTEDRESPONSE</p> <p><b>Meaning:</b> Environmental error. This connection is responding to an event via IXLEERSP to which the system is not currently expecting a response.</p> <p><b>Action:</b> Prior to issuing the IXLEERSP macro, an event exit must have passed a return code of X'8' in the IXLYEEPL to indicate to XES that a response will be made to the event through the IXLEERSP macro. Otherwise, the response will not be expected. Check your event exit, and your protocol for handling events.</p>
C	xxxx0C42	<p><b>Equate Symbol:</b> IXLRSNCODEINVALIDEVENT</p> <p><b>Meaning:</b> Environmental error. The response provided via IXLEERSP is not one that is expected for the event presented.</p> <p><b>Action:</b> Verify the event specified and the parameters passed on the IXLEERSP.</p>

Table 51. Return and Reason Codes for the IXLEERSP Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C48	<p><b>Equate Symbol:</b> IXLRNICODESUBJCONNNOTDEFINED</p> <p><b>Meaning:</b> Environmental error. The connection identified by SUBJCONTOKEN is in the not defined state.</p> <p><b>Action:</b> Verify that your parameter list has not been overlaid. Verify that SUBJCONTOKEN is from EEPLSUBJCONTOKEN from the EEPL passed to the event exit.</p>
C	xxxx0C49	<p><b>Equate Symbol:</b> IXLRNICODEREBUILDEERSPIGNORED</p> <p><b>Meaning:</b> Environmental error. EVENT=REBLDCONNFAIL is not valid because the connection identified by SUBJCONTOKEN is not in the active state. The original connection has terminated.</p> <p><b>Action:</b> None necessary, but check your protocol to determine why you thought you needed to respond to this event. Verify that SUBJCONTOKEN is from EEPLSUBJCONTOKEN from the EEPL passed to the event exit. You may have responded more than once to this event.</p>
C	xxxx0C6D	<p><b>Equate Symbol:</b> IXLRNICODESUBJCONNNOTFAILING</p> <p><b>Meaning:</b> Environmental error. An attempt to respond by proxy on behalf of the connector identified by the value provided for the SUBJTOKEN keyword failed because that connector is not in the failing state.</p> <p><b>Action:</b> Ensure that you do not respond by proxy on behalf of a failing connector until after you have been presented with the EEPLDISCFAILCONNECTION event exit.</p>
C	xxxx0C91	<p><b>Equate Symbol:</b> IXLRNICODESYSGMDRESPONSENOTPERMITTED</p> <p><b>Meaning:</b> The structure is in system-managed processing. A response is not permitted from the connection. The request is not processed.</p> <p>Applies to the following events: REBLDQUIESCE, REBLDCONNFAIL, REBLDCLEANUP, and REBLDSTOP.</p> <p><b>Action:</b> Check your protocol for system-managed processes.</p>
C	FFFFFFFF	<p><b>Equate Symbol:</b> IXLRNICODENOTAVAILABLE</p> <p><b>Meaning:</b> There are no coupling facility services available. The hardware support necessary to support coupling facility services might not be present.</p> <p><b>Action:</b> Re-IPL the system, or follow your particular failure management protocol.</p>
10	None	<p><b>Meaning:</b> XES processing has failed.</p> <p><b>Action:</b> Save the reason code information and contact the IBM support center.</p>



## Chapter 54. IXLFCOMP — Wait for Completion or Obtain Status of an IXLLIST or IXLCACHE Request

### Description

If you are an IXLLIST or IXLCACHE macro user who specified MODE=ASYNCTOKEN or specified MODE=SYNCTOKEN but had the request processed asynchronously, you can use the IXLFCOMP macro to do either of the following:

- Test whether your list or cache request has completed (OPTYPE=TEST). Choose this option if your task cannot be suspended or your program can perform other work while the list or cache request is being processed. IXLFCOMP's return code indicates whether the request has completed.
- Have your task suspended until the list or cache request completes (OPTYPE=COMPLETE). Choose this option if your task can be suspended and you have no other work to perform while the list or cache request is being processed.

When the request has completed (if it has not completed when you issue IXLFCOMP), control returns to you so you can check the results of the request in the output areas you specified on the list or cache request.

**Note:** The following information assumes that you are familiar with either the IXLLIST macro or the IXLCACHE macro, and that you are using either a list or cache structure.

The IXLFCOMP guidance information in *z/OS MVS Programming: Sysplex Services Guide* contains additional information about using the IXLFCOMP macro, including the use of IXLFCOMP in recovery scenarios.

For more information about the IXLLIST macro, see [Chapter 56, “IXLLIST — List Services,”](#) on page 961.

For more information about the IXLCACHE macro, see [Chapter 30, “IXLCACHE — Cache Services,”](#) on page 457.

### Environment

The following are the environment requirements for the caller.

Environment	Environment requirement
Minimum authorization:	Supervisor state and PSW key 0-7
Dispatchable unit mode:	Task or SRB
Cross memory mode:	HASN=PASN, any SASN. The primary address space must equal the primary address space of the caller of the IXLCONN macro.
AMODE:	31-bit
ASC mode:	Primary or AR
Interrupt status:	If you specify OPTYPE=TEST, your program might be enabled or disabled for I/O and external interrupts.  If you specify OPTYPE=COMPLETE, your program must be enabled (so that it can be suspended if necessary).
Locks:	If your program is disabled, it must hold the CPU lock. Otherwise, no locks may be held.
Control parameters:	See <a href="#">“Restrictions”</a> on page 944

## Programming Requirements

---

- If your program is in AR mode, issue SYSSTATE ASCENV=AR before you issue the IXLFCOMP macro. ASCENV=AR causes the system to generate code appropriate for AR mode. For more information about the SYSSTATE macro, see *z/OS MVS Programming: Assembler Services Reference ABE-HSP*.
- Include the IXLYCON mapping macro to generate the equate symbols for the return codes.

## Restrictions

---

- You can issue IXLFCOMP requests only for IXLLIST or IXLCACHE operations initiated by your own connection.
- Only one IXLFCOMP request can run at a time with a particular request token. If you attempt to issue another IXLFCOMP request with the same token while the first is running, your second request fails.
- All virtual storage areas passed to the IXLFCOMP macro must be addressable in at least one of the following ways:
  - From the primary, home, or secondary address space
  - From the PASN access list
  - From the DU access list
- Any virtual storage area specified on the corresponding prior IXLLIST or IXLCACHE request must still be addressable as it was at the time of the prior request. See the IXLLIST or IXLCACHE macros for the addressability requirements of the virtual storage areas associated with those requests.
- If you are running disabled, the parameter list and all storage areas addressed by macro parameters must reside in either fixed or disabled reference (DREF) storage. Furthermore, any virtual storage areas that were specified on the corresponding prior IXLLIST or IXLCACHE invocation must also reside in fixed or DREF storage if the IXLFCOMP caller is disabled.
- This service cannot be invoked by callers running as a disabled interrupt exit (DIE).

## Input Register Information

---

Before issuing the IXLFCOMP macro, the caller does not have to place any information into any register unless using it in register notation for a particular parameter, or using it as a base register.

## Output Register Information

---

When control returns to the caller, the GPRs contain:

### Register

#### Contents

**0**

Reason code, if applicable

**1**

Used as work register by the system

**2-13**

Unchanged

**14**

Used as work registers by the system

**15**

Return code

When control returns to the caller, the ARs contain:

### Register

#### Contents



**0-1**

Used as work registers by the system

**2-13**

Unchanged

**14-15**

Used as work registers by the system

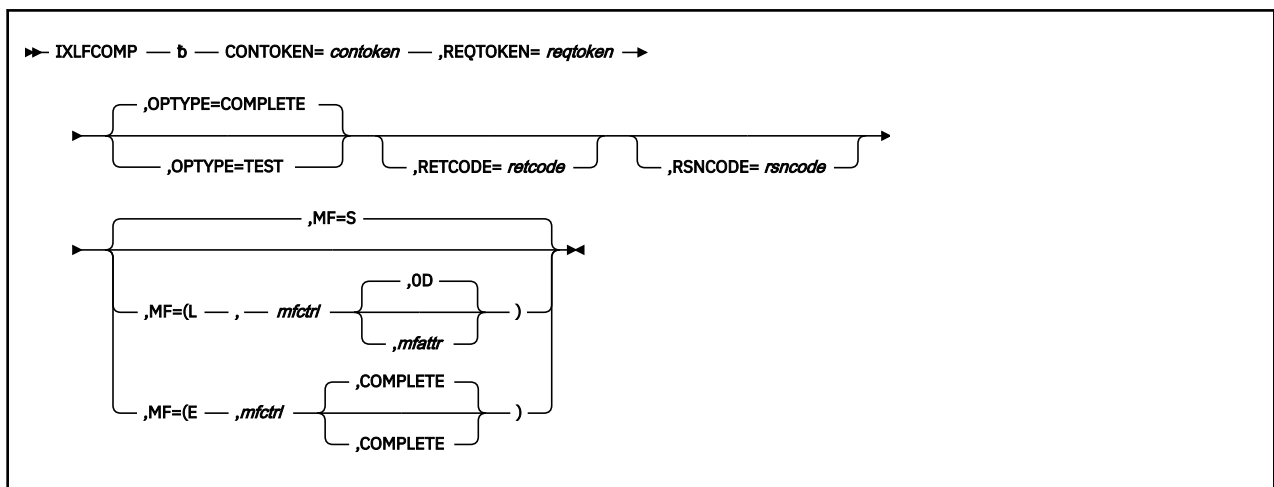
Some callers depend on register contents remaining the same before and after issuing a service. If the system changes the contents of registers on which the caller depends, the caller must save them before issuing the service, and restore them after the system returns control.

## Performance Implications

If OPTYPE=COMPLETE is specified, your task is suspended until the IXLLIST or IXLCACHE request completes.

## Syntax Diagram

The syntax of the IXLFCOMP macro is as follows:



## Parameter Descriptions

The parameter descriptions are listed in alphabetical order. Default values are underlined:

**CONTOKEN=contoken**

Use this input parameter to specify the connect token you received when you issued the IXLCONN macro to connect to the cache or list structure.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-character field that contains the connect token.

**,MF=S****,MF=(L,mfctrl)****,MF=(L,mfctrl,mfattr)****,MF=(L,mfctrl,OD)****,MF=(M,mfctrl)****,MF=(M,mfctrl,COMPLETE)****,MF=(M,mfctrl,NOCHECK)****,MF=(E,mfctrl)****,MF=(E,mfctrl,COMPLETE)****,MF=(E,mfctrl,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

**,mfctrl**

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

**,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

**,COMPLETE**

**,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

**COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if SMILE=*var* were an optional parameter and the default is SMILE=NO\_SMILE then it would not be documented. However, if the default was SMILE=:), then it would be documented because a value would be the default.

**NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

**,OPTYPE=COMPLETE**

**,OPTYPE=TEST**

Use OPTYPE=COMPLETE to request that the IXLFCOMP request return information only when the list or cache request identified by REQTOKEN has completed. If the request identified by REQTOKEN has not completed, your task will be suspended until it has completed.

Use OPTYPE=TEST to test the current processing status of the list or cache request. If your task cannot be suspended, use this option to poll for request completion.

**,REQTOKEN=reqtoken**

Use this input parameter to specify the asynchronous request token returned by the IXLLIST or IXLCACHE request. This token identifies the asynchronous request.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-character field that contains the asynchronous request token.

**,RETCODE=retcode**

Use this output parameter to specify a field to contain the return code. (The return code is also returned in register 15.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the return code when the request has completed.

**,RSNCODE=rsncode**

Use this output parameter to specify a field to contain the reason code returned, if applicable. (The reason code is also returned in register 0.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the reason code (if any) when the request has completed.

## ABEND Codes

---

Abend X'026' (See [z/OS MVS System Codes](#) for more information on this abend.)

## Return and Reason Codes

---

When the IXLFCOMP macro returns control to your program:

- GPR 15 (and *retcode* if you coded RETCODE) contains a return code.
- If the return code is not zero, GPR 0 (and *rsncode* if you coded RSNCODE) contains a reason code, if applicable.

You could receive any of the return codes and reason codes issued by IXLFCOMP, in addition to those described for IXLLIST and IXLCACHE. The return and reason codes that pertain to the IXLFCOMP request are listed in Table 52 on page 947. Otherwise, the return and reason codes pertain to the prior IXLLIST and IXLCACHE request that has completed. Refer to those services for the return and reason code information.

The answer area is not valid if you issue an IXLFCOMP OPTYPE=COMPLETE request and get back RC=IXLRETCODEPARMERROR with RSN=IXLRSNCODENOSUSPENDISABLE.

The equate symbols associated with each IXLFCOMP return code are as follows:

**4**

IXLRETCODEWARNING

**8**

IXLRETCODEPARMERROR

For the reason codes shown below, xxxx denotes information that is not part of the reason code.

Table 52. Return and Reason Codes for the IXLFCOMP Macro		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	xxxx0413	<p><b>Equate symbol:</b> IXLRSNCODEREQNOTCOMP</p> <p><b>Meaning:</b> The request identified by REQTOKEN has not completed. The IXLLIST or IXLCACHE answer area is not valid.</p> <p><b>Action:</b> Issue the IXLFCOMP macro again, with OPTYPE=TEST to determine when the list or cache request will complete (with the answer area filled in) or with OPTYPE=COMPLETE to wait for the list or cache request to complete.</p>

Table 52. Return and Reason Codes for the IXLFCOMP Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0801	<p><b>Equate Symbol:</b> IXLRNCODEBADPARMLIST</p> <p><b>Meaning:</b> Program error. The parameter list for this request is not addressable. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Verify that:</p> <ul style="list-style-type: none"> <li>• The parameter list address is uncorrupted.</li> <li>• The parameter list is addressable in the caller's primary address space.</li> <li>• If you are issuing this macro while disabled, the parameter list resides in either page-fixed or DREF storage.</li> <li>• If you are issuing this macro in AR-mode and you specified the parameter list address using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If you are issuing this macro in AR-mode, SYSSTATE ASCENV=AR must be issued before issuing this macro.</li> </ul>
8	xxxx0804	<p><b>Equate Symbol:</b> IXLRNCODEBADVERSIONNUM (or IXLRNCODEBADVERSION#)</p> <p><b>Meaning:</b> The version number in the macro parameter list is not compatible with the level of XES currently being used. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that your program did not overlay the parameter list storage.</li> <li>• Verify that your program was assembled with the correct macro library for the release of MVS on which your program is running.</li> </ul>
8	xxxx0831	<p><b>Equate symbol:</b> IXLRNCODEBADREQTOKEN</p> <p><b>Meaning:</b> Program error. You specified an asynchronous request token that was not valid. Your IXLFCOMP request failed. The IXLLIST or IXLCACHE answer area is not valid.</p> <p><b>Action:</b> Ensure that the request token passed on the IXLFCOMP request is the same one that you got back from IXLLIST or IXLCACHE on a SYNCTOKEN or ASYNCTOKEN request. Verify that if the IXLLIST or IXLCACHE request was a SYNCTOKEN request, that it was in fact processed asynchronously. If the request was processed synchronously, the request token is not meaningful and there is no reason to use the IXLFCOMP macro. Verify that you are not re-issuing the IXLFCOMP macro using the same request token after observing the completion of the request. Once you observe completion of a request, the request token is no longer valid, and any subsequent IXLFCOMP requests will complete with this reason code.</p>

Table 52. Return and Reason Codes for the IXLFCOMP Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0851	<p><b>Equate symbol:</b> IXLRNOCODENOSUSPENDISABLE</p> <p><b>Meaning:</b> Program error. A request specifying OPTYPE=COMPLETE failed because the prior IXLLIST or IXLCACHE request has not completed, and the caller is disabled and cannot be suspended.</p> <p><b>Action:</b> Release the CPU lock to become enabled and reissue the request.</p>
10	xxxx10xx	<p><b>Equate symbol:</b> IXLRETCODECOMPERROR</p> <p><b>Meaning:</b> XES processing failure. The state of the involved structure and the disposition of the request are unpredictable.</p> <p><b>Action:</b> Contact the IBM Support Center.</p>



## Chapter 55. IXLFORCE – Deleting a Persistent Structure or Failed Connection

### Description

The IXLFORCE service is intended only for use as a cleanup utility; therefore, you do not need to be connected to a structure to invoke the macro. IXLFORCE allows the requestor to delete:

- A persistent structure
- A failed-persistent connection to a structure
- All failed-persistent connections to a structure
- A structure dump
- The serialization held on a structure for a structure dump

A persistent structure can be deleted if :

- On systems at z/OS V1R6, or with APAR OA02620 installed, structures with only failed-persistent connections can be deleted. Otherwise, structures with active or failed-persistent connections cannot be deleted.
- There are no failed-persistent connections pending reconciliation into the CFRM active policy.
- There is no structure dump associated with the structure.

An active connection cannot be deleted. A failed-persistent connection can be deleted only if:

- All connectors active at the time the failed-persistent connection terminated have either provided an event exit response acknowledging the termination of the failed-persistent connector, or have themselves been terminated.

While structure rebuild is in progress:

- A structure cannot be deleted because a structure that has a status of rebuild in progress must have at least one active connector.
- A failed-persistent connection cannot be deleted.

While structure duplexing is in progress, a persistent structure cannot be deleted nor can a failed-persistent connection be deleted except as noted below.

- When a persistent structure is in the Duplex Established phase of a duplexing rebuild and a request to stop the duplexing and switch to the new structure is not in progress, the structure can be deleted. Both the old and the new structure are deleted.
- When a failed-persistent connection to a structure that is in the Duplex Established phase of duplexing rebuild and a request to stop the duplexing and switch to the new structure is not in progress, the failed-persistent connection can be deleted. Failed-persistent connectors to both instances of the structure will be deleted.

Note that forcing the deletion of a structure or a connection without understanding how the structure is being used may cause loss of data.

The security administrator may have controlled the use of installation structures through the use of RACF or another security product. If so, ensure that you are authorized to issue the IXLFORCE macro for the structure.

For more information about structure and connection persistence, see [z/OS MVS Programming: Sysplex Services Guide](#).

## Environment

The following are the environment requirements for the caller.

Environment	Environment requirement
Authorization	Supervisor state or PKM keys 0 - 7
Dispatchable unit mode:	Task
Authorization:	PASN=HASN, any SASN
AMODE:	31-bit
ASC mode:	Primary or access register (AR)
Interrupt status:	Enabled for I/O and external interrupts
Locks:	No locks held, with no enabled, unlocked task (EUT) FRRs established
Control parameters:	Must be in the primary address space or be in an address/data space that is addressable through a public entry on the caller's dispatchable unit access list (DU-AL)

## Restrictions

- The requestor can delete a persistent structure only if there are no active connections to the structure, there are no failed-persistent connections pending reconciliation into the CFRM active policy, and there is no structure dump associated with the structure.

A structure cannot be deleted while structure rebuild is in progress for the structure, because a structure for which rebuild is in progress must have at least one active connector.

A structure in the duplexing rebuild process can be deleted only if there are no active or failed-persistent connections, the structure is in the Duplex Established phase, and a switch to the new structure is not in progress.

- The requestor can delete a failed-persistent connection. The failed-persistent connection will be deleted from the old structure, the new structure, or both instances of the structure. However, an active connection cannot be deleted, nor can a failed-persistent connection be deleted while structure rebuild is in progress for the structure.

A connection to a structure in the duplexing rebuild process can be deleted only when the connection is failed-persistent and the structure is in the Duplex Established phase with a switch to the new structure not in progress.

- The requestor can delete only a single structure, structure dump, or structure dump serialization with one invocation of the IXLFORCE macro. However, the requestor can delete multiple failed-persistent connections to a structure with one invocation of the IXLFORCE macro.

A program can use the IXCQUERY macro to determine the set of objects that are candidates for deletion. In the case of forcing all failed-persistent connections to a structure, IXCQUERY can be used to determine the set of connections that either will be or were forced.

## Input Register Information

Before issuing the IXLFORCE macro, the caller does not have to place any information into any register unless using it in register notation for a particular parameter, or using it as a base register.

## Output Register Information

When control returns to the caller of the IXLFORCE macro, the general purpose registers (GPRs) contain:



### Register Contents

- 0** Reason code if applicable
- 1** Used as work register by the system
- 2-13** Unchanged
- 14** Used as work register by the system
- 15** Return code

When control returns to the caller of the IXLFORCE macro, the access registers (ARs) contain:

### Register Contents

- 0-1** Used as work registers by the system
- 2-13** Unchanged
- 14-15** Used as work registers by the system

**For registers that the system changes,** a caller who depends on these registers containing the same value before and after issuing the macro must save these registers before issuing the macro and restore them after the system returns control.

## Programming Requirements

---

If the program is in AR mode, issue the SYSSTATE ASCENV=AR macro before IXLFORCE. SYSSTATE ASCENV=AR tells the system to generate code appropriate for AR mode.

## Performance Implications

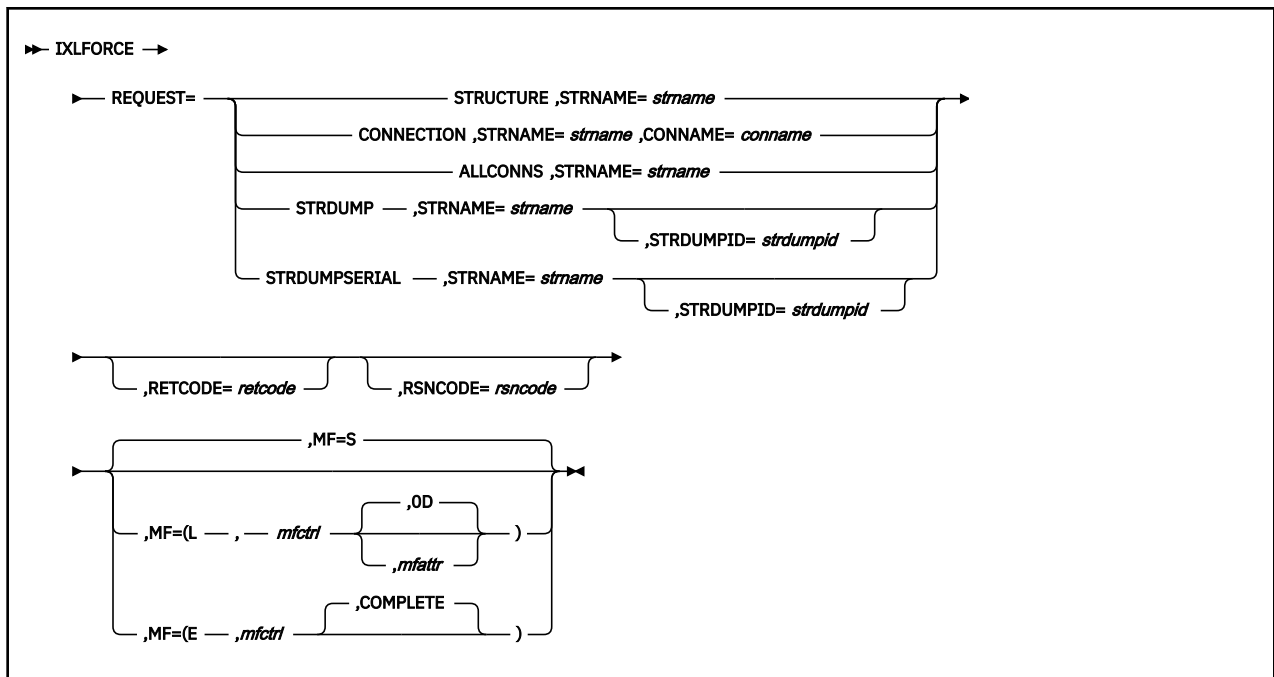
---

IXLFORCE processing might involve referencing or updating the CFRM active policy to reflect the operation that has been requested. When invoking this macro, be aware of the I/O to the CFRM couple data set that might be generated.

## Syntax Diagram

---

The syntax of the IXLFORCE macro is as follows:



## Parameter Descriptions

The parameter descriptions are listed in alphabetical order. Default values are underlined:

### **CONNNAME=conname**

Use this input parameter to specify the connect name of the connection to be deleted. The connection identified is connected to the structure specified by STRNAME.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the 16-character input field that contains the connect name of the connection to be deleted.

### **MF=S**

#### **MF=(L, mfctrl)**

#### **MF=(L, mfctrl, mfattr)**

#### **MF=(L, mfctrl, OD)**

#### **MF=(M, mfctrl)**

#### **MF=(M, mfctrl, COMPLETE)**

#### **MF=(M, mfctrl, NOCHECK)**

#### **MF=(E, mfctrl)**

#### **MF=(E, mfctrl, COMPLETE)**

#### **MF=(E, mfctrl, NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

### **mfctrl**

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

**,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

**,COMPLETE**

**,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

**COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if *SMILE=var* were an optional parameter and the default is *SMILE=NO\_SMILE* then it would not be documented. However, if the default was *SMILE=-*), then it would be documented because a value would be the default.

**NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

**REQUEST=STRUCTURE**

**REQUEST=CONNECTION**

**REQUEST=ALLCONNS**

**REQUEST=STRDUMP**

**REQUEST=STRDUMPSERIAL**

Use this input parameter to specify what the requestor wants to delete.

**STRUCTURE**

Indicates that the structure is to be deleted. A structure can be deleted only when there are no active or failed-persistent connections to the structure. You can use *IXCQUERY* to determine the state of connections to a structure. If there is a structure dump associated with the structure, deallocation of the structure remains pending until the structure dump is released.

**CONNECTION**

Indicates that the specified failed-persistent connection to a structure is to be deleted. If the last connection to a non-persistent structure is deleted, the structure also is deleted.

**ALLCONNS**

Indicates that all failed-persistent connections to a structure are to be deleted. If the last connection to a non-persistent structure is deleted, the structure also is deleted.

**STRDUMP**

Indicates that a structure dump associated with either an active structure or a structure pending deallocation is to be deleted. Structures pending deallocation will be processed only if a nonzero *STRDUMPID* is specified.

**STRDUMPSERIAL**

Indicates that the serialization held on the structure for the structure dump is to be released. For structures pending deallocation, however, release of dumping serialization is not supported because such structures have no active connectors who might be impacted by the dump serialization.

**,RETcode=retcode**

Use this output parameter to specify a field to contain the return code. (The return code is also returned in register 15.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the return code when the request has completed.

**,RSNCODE=*rsncode***

Use this output parameter to specify a field to contain the reason code returned, if applicable. (The reason code is also returned in register 0.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the reason code (if any) when the request has completed.

**,STRDUMPID=0****,STRDUMPID=*strdumpid***

Use this input parameter to specify the structure dump identifier of the structure dump to be deleted. or of the dump for which structure dump serialization is to be released. This is used to uniquely identify the specific instance of a structure dump, since there might be multiple instances of a structure, each with an associated structure dump.

If you do not specify STRDUMPID or specify a value of zero on STRDUMPID, the system deletes, or releases dump serialization for, all structure dumps associated with the allocated structure. To delete a structure dump for a structure pending deallocation, you must specify a non-zero STRDUMPID. Release of structure dump serialization is not supported for a structure pending deallocation because such structures have no active connectors who might be impacted by the dump serialization. If a structure rebuild is in process, the system might delete, or release dump serialization for, the dump associated with the rebuild old structure, rebuild new structure, or both.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a halfword input field that contains the structure dump identifier of the structure dump to be deleted.

**,STRNAME=*strname***

Use this input parameter to specify the name of the structure that is identified on the IXLFORCE request. The structure might be active or pending deallocation. You can use the IXCQUERY macro to determine which structures have been defined in the CFRM active policy.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the 16-character input field that contains the structure name.

## Return and Reason Codes

---

When the IXLFORCE macro returns control to your program:

- GPR 15 (and *retcode*, if you coded RETCODE) contains a return code.
- GPR 0 (and *rsncode*, if you coded RSNCODE) contains a reason code, if applicable.

Macro IXLYCON provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

**0**

IXLRETCODEOK

**4**

IXLRETCODEWARNING

**8**

IXLRETCODEPARMERROR

**C**

IXLRETCODEENVERROR

**10**

IXLRETCODECOMPERROR

The following table identifies the hexadecimal return and reason codes and the equate symbol associated with each reason code.

Table 53. Return and Reason Codes for the IXLFORCE Macro

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0	None	<p><b>Meaning:</b> IXLFORCE processing completed successfully. The structure, failed-persistent connection, all failed-persistent connections, structure dump, or structure dump serialization has been deleted as requested.</p> <p><b>Action:</b> None.</p>
4	xxxx041A	<p><b>Equate Symbol:</b> IXLRSNCODEFORCECONNDELSTR</p> <p><b>Meaning:</b> The IXLFORCE request to delete a connection or all connections succeeded, but the structure disposition was DELETE, and no active connections to the structure remained. The structure was deallocated.</p> <p><b>Action:</b> None necessary. However, if this result is unexpected, determine if the structure disposition was specified properly in IXLCONN.</p>
4	xxxx041BA	<p><b>Equate Symbol:</b> IXLRSNCODEFORCESTRDELCONNS</p> <p><b>Meaning:</b> Force of a structure was successful, but also resulted in the deletion of failed-persistent connection(s).</p> <p><b>Action:</b> None necessary.</p>
4	xxxx041C	<p><b>Equate Symbol:</b> IXLRSNCODEPENDING</p> <p><b>Meaning:</b> The IXLFORCE request could not be processed immediately. The request is pending, and the system will process it when the condition causing the delay is resolved.</p> <p><b>Action:</b> None necessary. If the delay is due to a dump associated with the structure, you might supply a dump data set for SDUMPX to write the dump or you might delete the dump.</p>
8	xxxx0801	<p><b>Equate Symbol:</b> IXLRSNCODEBADPARMLIST</p> <p><b>Meaning:</b> The IXLFORCE parameter list is not accessible.</p> <p><b>Action:</b> Verify that:</p> <ul style="list-style-type: none"> <li>• The parameter list address is uncorrupted.</li> <li>• The parameter list is addressable in the caller's primary address space.</li> <li>• If you are invoking IXLFORCE in AR-mode and you specified the parameter list address using implicit register notation, the corresponding access register was updated appropriately.</li> <li>• If you are invoking IXLFORCE in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLFORCE.</li> </ul>
8	xxxx0802	<p><b>Equate Symbol:</b> IXLRSNCODEBADPARMLISTALET</p> <p><b>Meaning:</b> The IXLFORCE parameter list ALET is not valid.</p> <p><b>Action:</b> Ensure that the ALET is zero or that the ALET represents a valid entry on the DU-AL.</p>

Table 53. Return and Reason Codes for the IXLFORCE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0804	<b>Equate Symbol:</b> IXLRNCODEBADVERSION# <b>Meaning:</b> Version number in the IXLFORCE parameter list is not valid. <b>Action:</b> <ul style="list-style-type: none"> <li>• Verify that your program did not overlay the parameter list storage.</li> <li>• Verify that your program was assembled with the correct macro library for the release of MVS that your program is running on.</li> </ul>
8	xxxx0806	<b>Equate Symbol:</b> IXLRNCODESRBMODE <b>Meaning:</b> The requestor is in SRB mode. The request fails. <b>Action:</b> Do not issue the IXLFORCE macro when running in SRB mode.
8	xxxx0807	<b>Equate Symbol:</b> IXLRNCODENOTENABLED <b>Meaning:</b> The requestor is not in an enabled state. The request fails. <b>Action:</b> Ensure that the user is enabled for I/O and external interrupts.
8	xxxx0809	<b>Equate Symbol:</b> IXLRNCODEPRIMARYNOTHOME <b>Meaning:</b> Issuer's primary address space is not equal to the home address space. <b>Action:</b> Make sure that the primary address space and the home address space are the same at the time of the IXLFORCE invocation.
8	xxxx084C	<b>Equate Symbol:</b> IXLRNCODEBADNOSAFAUTH <b>Meaning:</b> Environmental error. The user does not have the proper SAF authorization. The request fails. <b>Action:</b> Determine why the installation has not allowed the proper SAF authorization for access to the structure.
C	xxxx0C05	<b>Equate Symbol:</b> IXLRNCODESTRNOTINPOLICY <b>Meaning:</b> Environmental error. The requested structure is not in the CFRM active policy. <b>Action:</b> Specify a structure that is currently defined in the CFRM active policy or switch to a new CFRM policy that contains the structure for which this IXLFORCE was invoked.
C	xxxx0C0A	<b>Equate Symbol:</b> IXLRNCODESTRNOTALLOCATED <b>Meaning:</b> Environmental error. The requested structure is not currently allocated. <b>Action:</b> Ensure that the structure name has been specified correctly.

Table 53. Return and Reason Codes for the IXLFORCE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C26	<b>Equate Symbol:</b> IXLSRNCODECONACTIVE <b>Meaning:</b> Environmental error. The requested connection is active. IXLFORCE can only delete connections in a failed-persistent state. <b>Action:</b> Ensure that the connection to be deleted is in a failed-persistent state.
C	xxxx0C27	<b>Equate Symbol:</b> IXLSRNCODERSPNOTREC <b>Meaning:</b> Environmental error. One or more surviving connections have not provided an event exit response for the requested connection. <b>Action:</b> Examine your protocol to determine why responses have not been provided.
C	xxxx0C28	<b>Equate Symbol:</b> IXLSRNCODESTILLACTIVECONN <b>Meaning:</b> Environmental error. The requested structure cannot be deleted because it still has active connections. <b>Action:</b> Do not delete a structure with active connections. Use IXCQUERY to determine the state of the structure connections.
C	xxxx0C29	<b>Equate Symbol:</b> IXLSRNCODEXESNOTACTIVE <b>Meaning:</b> Environmental error. The CFRM function is not active or is not available. <b>Action:</b> Bring the CFRM couple data set in use in the sysplex by issuing the SETXCF COUPLE,TYPE=CFRM,PCOUPLE= command.
C	xxxx0C2A	<b>Equate Symbol:</b> IXLSRNCODENOSUCHCONNECTION <b>Meaning:</b> Environmental error. The requested connection does not exist, or in the case of an ALLCONNS request, there are no failed-persistent connections to be deleted. <b>Action:</b> Ensure that the connection name was specified correctly.
C	xxxx0C33	<b>Equate Symbol:</b> IXLSRNCODECONNPENDINGRECONCIL <b>Meaning:</b> Environmental error. The specified structure still has connections in the coupling facility that are pending reconciliation into the CFRM active policy. <b>Action:</b> Retry the request at a later time.
C	xxxx0C4D	<b>Equate Symbol:</b> IXLSRNCODENOSTRDUMP <b>Meaning:</b> Environmental error. Either no structure dump was associated with the requested structure, or no structure dump with a matching STRDUMPID was associated with the requested structure. <b>Action:</b> Verify that the structure name and/or the structure dump identifier were specified correctly.

Table 53. Return and Reason Codes for the IXLFORCE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C51	<b>Equate Symbol:</b> IXLRSNCODEREBUILDINPROGRESS <b>Meaning:</b> Environmental error. The requested structure is in a structure rebuild or a structure duplexing rebuild process. Structures and connections cannot be deleted while a structure rebuild is in progress. Structures and connections can be deleted while a structure duplexing rebuild is in the Duplex Established phase and no stop to switch to the new structure has been requested. <b>Action:</b> Retry the request at a later time.
C	FFFFFFFF	<b>Equate Symbol:</b> IXLRSNCODENOTAVAILABLE <b>Meaning:</b> Environmental error. XES functions are not available. This can occur because the coupling facility hardware necessary to provide XES functions is not present. <b>Action:</b> Re-IPL the system, or follow your particular management protocol.
10	xxxx10xx	<b>Meaning:</b> Failure in XES processing. <b>Action:</b> Save the reason code information and contact the IBM support center.



## Chapter 56. IXLLIST – List Services

### Description

A list structure is a coupling facility structure that enables multisystem application to share information organized as a set of lists and an optional set of locks. Each list consists of a queue of list entries.

The IXLLIST macro provides the means for applications in a sysplex to access and manipulate the data in a coupling facility list structure.

The IXLLIST macro provides many services. The service you require is designated by the REQUEST parameter. These services are detailed in the following chapters .

Starting with OS/390 Release 9, functions provided by the IXLLIST macro are supplemented with three additional macros — IXLLSTC, IXLLSTE, and IXLLSTM, that applications can use to access and manipulate the data in a coupling facility list structure. IXLLIST invocations will continue to be supported after OS/390 Release 8, but any new functionality will be added to the three new macros.

Be sure to read the IXLLIST guidance information in *z/OS MVS Programming: Sysplex Services Guide*. The information presented here assumes you have done so. Table 54 on page 961 tells you where to find the reference information for each IXLLIST request type and indicates the OS/390 Release 9 new list services macro that provides the same, and possibly additional, functionality:

Table 54. Request Types for IXLLIST	
Request Type	Description
DELETE	Delete a list entry from a coupling facility list structure. You can also read an entry into your buffer and delete it from the coupling facility list structure. See also IXLLSTE REQUEST=DELETE.  See <a href="#">Chapter 57, “IXLLIST REQUEST=DELETE,” on page 967</a> .
DELETE_ENTRYLIST	Delete multiple list entries that are designated in an entry list contained in the storage area specified by BUFLIST or BUFFER. See also IXLLSTM REQUEST=DELETE_ENTRYLIST.  See <a href="#">Chapter 58, “IXLLIST REQUEST=DELETE_ENTRYLIST,” on page 995</a> .
DELETE_MULT	Delete multiple list entries from a coupling facility list structure. See also IXLLSTM REQUEST=DELETE_MULT.  See <a href="#">Chapter 59, “IXLLIST REQUEST=DELETE_MULT,” on page 1017</a> .
DEQ_EVENTQ	Read and dequeue event monitor controls from your event queue. See also IXLLSTC REQUEST=DEQ_EVENTQ.  See <a href="#">Chapter 60, “IXLLIST REQUEST=DEQ_EVENTQ,” on page 1035</a> .
LOCK	Perform locking functions such as SET, RESET, TEST, READNEXT. See also IXLLSTC REQUEST=LOCK.  See <a href="#">Chapter 61, “IXLLIST REQUEST=LOCK,” on page 1053</a> .
MONITOR_EVENTQ	Start or stop monitoring your event queue for the presence of event monitor controls. See also IXLLSTC REQUEST=MONITOR_EVENTQ.  See <a href="#">Chapter 62, “IXLLIST REQUEST=MONITOR_EVENTQ,” on page 1067</a> .
MONITOR_LIST	Start or stop monitoring a specified list for the presence of entries. See also IXLLSTC REQUEST=MONITOR_LIST.  See <a href="#">Chapter 63, “IXLLIST REQUEST=MONITOR_LIST,” on page 1081</a> .

Table 54. Request Types for IXLLIST (continued)

Request Type	Description
MONITOR_SUBLIST	Start or stop monitoring a specified sublist for the presence of entries. See also IXLLSTC REQUEST=MONITOR_SUBLIST. See <a href="#">Chapter 64, “IXLLIST REQUEST=MONITOR_SUBLIST,” on page 1095.</a>
MONITOR_SUBLISTS	Start monitoring a set of sublists for the presence of entries. See also IXLLSTC REQUEST=MONITOR_SUBLISTS. See <a href="#">Chapter 65, “IXLLIST REQUEST=MONITOR_SUBLISTS,” on page 1109.</a>
MOVE	Allows you to move an existing list entry from one list to another list or within the same list. You can also create a new list entry and place it on a list and write data to it. It is also possible to move an entry and read it into your buffer. See also IXLLSTE REQUEST=MOVE. See <a href="#">Chapter 66, “IXLLIST REQUEST=MOVE,” on page 1129.</a>
READ	Read a list entry from the coupling facility list structure into the storage areas specified by BUFFER or BUFLIST and/or ADJAREA. See also IXLLSTE REQUEST=READ. See <a href="#">Chapter 67, “IXLLIST REQUEST=READ,” on page 1169.</a>
READ_EMCONTROLS	Retrieve control information about a sublist you are monitoring. See also IXLLSTC REQUEST=READ_EMCONTROLS. See <a href="#">Chapter 68, “IXLLIST REQUEST=READ_EMCONTROLS,” on page 1197.</a>
READ_EQCONTROLS	Retrieve control information about your event queue. See also IXLLSTC REQUEST=READ_EQCONTROLS. See <a href="#">Chapter 69, “IXLLIST REQUEST=READ_EQCONTROLS,” on page 1209.</a>
READ_LCONTROLS	Retrieve list control and monitoring information for a specified list. See also IXLLSTC REQUEST=READ_LCONTROLS. See <a href="#">Chapter 70, “IXLLIST REQUEST=READ_LCONTROLS,” on page 1221.</a>
READ_LIST	Read multiple entries from a single list into the storage areas specified by BUFFER or BUFLIST and/or ADJAREA and ANSAREA. See also IXLLSTM REQUEST=READ_LIST. See <a href="#">Chapter 71, “IXLLIST REQUEST=READ_LIST,” on page 1237.</a>
READ_MULT	Read multiple entries from the coupling facility list structure into the storage areas specified by BUFFER or BUFLIST and/or ADJAREA and ANSAREA. See also IXLLSTM REQUEST=READ_MULT. See <a href="#">Chapter 72, “IXLLIST REQUEST=READ_MULT,” on page 1265.</a>
WRITE	Write data from the storage areas specified by BUFFER or BUFLIST and/or ADJAREA to a list entry in the coupling facility list structure. See also IXLLSTE REQUEST=WRITE. See <a href="#">Chapter 73, “IXLLIST REQUEST=WRITE,” on page 1291.</a>
WRITE_LCONTROLS	Modify list controls for the specified list. This includes the list authority, list limit, list description, list cursor, list key, and maximum list key. See also IXLLSTC REQUEST=WRITE_LCONTROLS. See <a href="#">Chapter 74, “IXLLIST REQUEST=WRITE_LCONTROLS,” on page 1323.</a>

## Environment

The following are the environment requirements for the caller.

Environment	Environment requirement
Minimum authorization:	Supervisor state and PSW key 0-7

Environment	Environment requirement
Dispatchable unit mode:	Task or SRB
Cross memory mode:	Any PASN, any HASN, any SASN. The primary address space must be the same as the primary address space at the time the connection service (IXLCONN) was issued. This address space is referred to as the connector's address space.  The requestor's address space refers to the home address space of the unit of work that is issuing the IXLLIST request. The requestor's address space may or may not be the same as the connector's address space.
AMODE:	31-bit
ASC mode:	Primary or access register (AR)
Interrupt status:	Enabled or disabled for I/O and external interrupts
Locks:	Disabled callers must be legally disabled by holding the CPU lock and cannot hold other disabled locks. Enabled callers must not hold any locks. If MODE=SYNCSUSPEND is specified, the caller must be enabled.
Control parameters:	See “Restrictions” on page 964

## Programming Requirements

- If your program is in AR mode, issue the SYSSTATE ASCENV=AR macro before IXLLIST. SYSSTATE ASCENV=AR tells the system to generate code appropriate for AR mode.
- Include the IXLYCON mapping macro in your program. This macro provides a list of equate symbols for users of XES services and exits.
- Include mapping macros IXLYEMC, IXLYLAA, IXLYLCTL, IXLYLMI, and IXLYMSRI in your program as necessary. Table 55 on page 963 lists these macros, the information and areas they map, and the particular IXLLIST requests and parameters they apply to.

Table 55. Mapping Macros for IXLLIST			
Mapping Macro	Information	Area	Request/Parameter
IXLYEMC	Event monitor controls	BUFLIST buffers, BUFFER area	DEQ_EVENTQ
IXLYLAA	Answer area output	ANSAREA area	All requests
IXLYLCTL	List entry controls	Field LAALCTL of IXLYLAA	READ, WRITE, MOVE, DELETE, READ_LIST, READ_MULT
		Field LAARLRLCTLs of IXLYLAA, BUFLIST buffers, BUFFER area	READ_LIST & READ_MULT (when TYPE=ECONTROLS)
IXLYLMI	List monitoring information	BUFLIST buffers, BUFFER area	READ_LCONTROLS
IXLYMSRI	Monitor sublists registration information	BUFLIST buffers, BUFFER area	MONITOR_SUBLISTS

## Restrictions

---

- If you specify BUFLIST and PAGEABLE=YES, all of the buffers in the list must be in the same area of storage; you cannot mix common and private storage addresses for the buffers in the list.
- This service cannot be invoked by callers running as a disabled interrupt exit (DIE).
- The caller's parameter list must be addressable in the caller's primary address space.
- If the caller is running in AR ASC mode and specifies a parameter using register notation, the access register corresponding to the general register must appropriately qualify the general register.
- The virtual storage areas specified by the ADJAREA, ANSAREA, and MOSVECTOR parameters must be addressable in the caller's primary address space or from the caller's PASN access list.
- The virtual storage areas specified by parameters other than ADJAREA, ANSAREA, and MOSVECTOR can be addressable in the caller's primary, secondary, or home address spaces, from the PASN access list, or from the dispatchable unit access list (DU-AL).
- If the caller is disabled then the parameter list and all storage areas addressed by the parameters must reside in either nonpageable or disabled reference storage.

## Input Register Information

---

Before issuing the IXLLIST macro, the caller does not have to place any information into any register unless using it in register notation for a particular parameter, or using it as a base register.

## Output Register Information

---

When control returns to the caller, the GPRs contain:

### Register

#### Contents

**0**

Reason code

**1**

Used as work register

**2-13**

Unchanged

**14**

Used as work register

**15**

Return code

When control returns to the caller, the ARs contain:

### Register

#### Contents

**0-1**

Used as work register

**2-13**

Unchanged

**14-15**

Used as work register

## Performance Implications

---

See [z/OS MVS Programming: Sysplex Services Guide](#) for performance information.

## Understanding IXLLIST Version Support

The IXLLIST macro supports multiple versions — the version number corresponds to the level of the IXLLIST macro in which a keyword or function was introduced. The higher the version number, the more recent the version of the IXLLIST macro. Some IXLLIST keywords are supported for multiple versions, but have different functions for each version.

- Keywords not specifically noted here are supported by all versions starting with version 0 and higher of the IXLLIST macro.
- The following IXLLIST keywords were available in the version 0 level of the macro, but have expanded function in the version 1 level: AUTHCOMP and NEWAUTH
- The following IXLLIST keywords are supported by all versions starting with version 1 and higher of the IXLLIST macro.
  - KEYCOMP
  - LISTKEY
  - LISTKEYINC
  - MAXLISTKEY
  - SETCURSOR
- The following keywords and functions are supported by all versions starting with version 2 and higher of the IXLLIST macro.
  - ENDINDEX
  - MOSVECTOR
  - REQUEST=MONITOR\_SUBLIST
  - REQUEST=MONITOR\_SUBLISTS
  - REQUEST=MONITOR\_EVENTQ
  - REQUEST=READ\_EMCONTROLS
  - REQUEST=READ\_EQCONTROLS
  - REQUEST=DEQ\_EVENTQ
  - STARTINDEX
  - UNC
- The following IXLLIST keyword is supported by all versions starting with version 3 and higher of the IXLLIST macro.
  - EXTRESTOKEN

Specify the version of the parameter list that you want generated with the PLISTVER keyword. See Chapter 2, “Specifying a Macro Version Number,” on page 5 for considerations when specifying the version of the parameter list with PLISTVER.

## Coding Equivalent Functions

Table 56 on page 965 displays examples of using version 1 keywords to achieve the equivalent function that was provided in version 0.

Table 56. IXLLIST Version Support	
Version 1 Keyword	Equivalent Function to Version 0
AUTHCOMPTYPE	REQUEST=WRITE_LCONTROLS, AUTHCOMP=authcomp,AUTHCOMPTYPE=EQUAL
CURSORUPDTYPE	REQUEST=..., UPDATECURSOR=YES,CURSORUPDTYPE=NEXT
LISTKEYTYPE	REQUEST=WRITE/MOVE, LISTKEYTYPE=NOLISTKEY
VERSCOMPTYPE	REQUEST=..., VERSCOMP=verscomp,VERSCOMPTYPE=EQUAL



## Chapter 57. IXLLIST REQUEST=DELETE

### Description

A DELETE request allows you to delete a single list entry from the list structure. Once removed from the list, the storage resources associated with the entry can be reused.

You can also specify that the designated entry be deleted only if it contains data that satisfies a version number comparison.

The entry you designate must exist on the list you specify, or the request fails with no change to the structure.

With a coupling facility of CFLEVEL=1 or higher, the following additional functions are supported for a DELETE request:

- You can specify that a list entry is to be deleted only after a successful comparison of the list authority value for the target list with a user specified value. You specify the list authority comparison requirements using AUTHCOMP and AUTHCOMPTYPE. If the request is successful, you also can update the list authority value to a new value that you specify with NEWAUTH.
- Version number comparison is enhanced to allow for an additional equal-or-less-than-equal comparison operation.
- There are additional list cursor options. You can update the list cursor conditionally, and you can set the list cursor to point to the current entry instead of the previous or next entry.

On the DELETE request, you can optionally read data from the entry by specifying DATAOPER=READ. Before it deletes the entry, the system reads any combination of the following types of data for the designated entry:

- List entry controls
- Entry data
- Adjunct data

Whether a particular data type is returned also depends on whether the local storage area to contain that data is specified. Listed below is the data that is returned and the storage area that should be specified to contain that returned data:

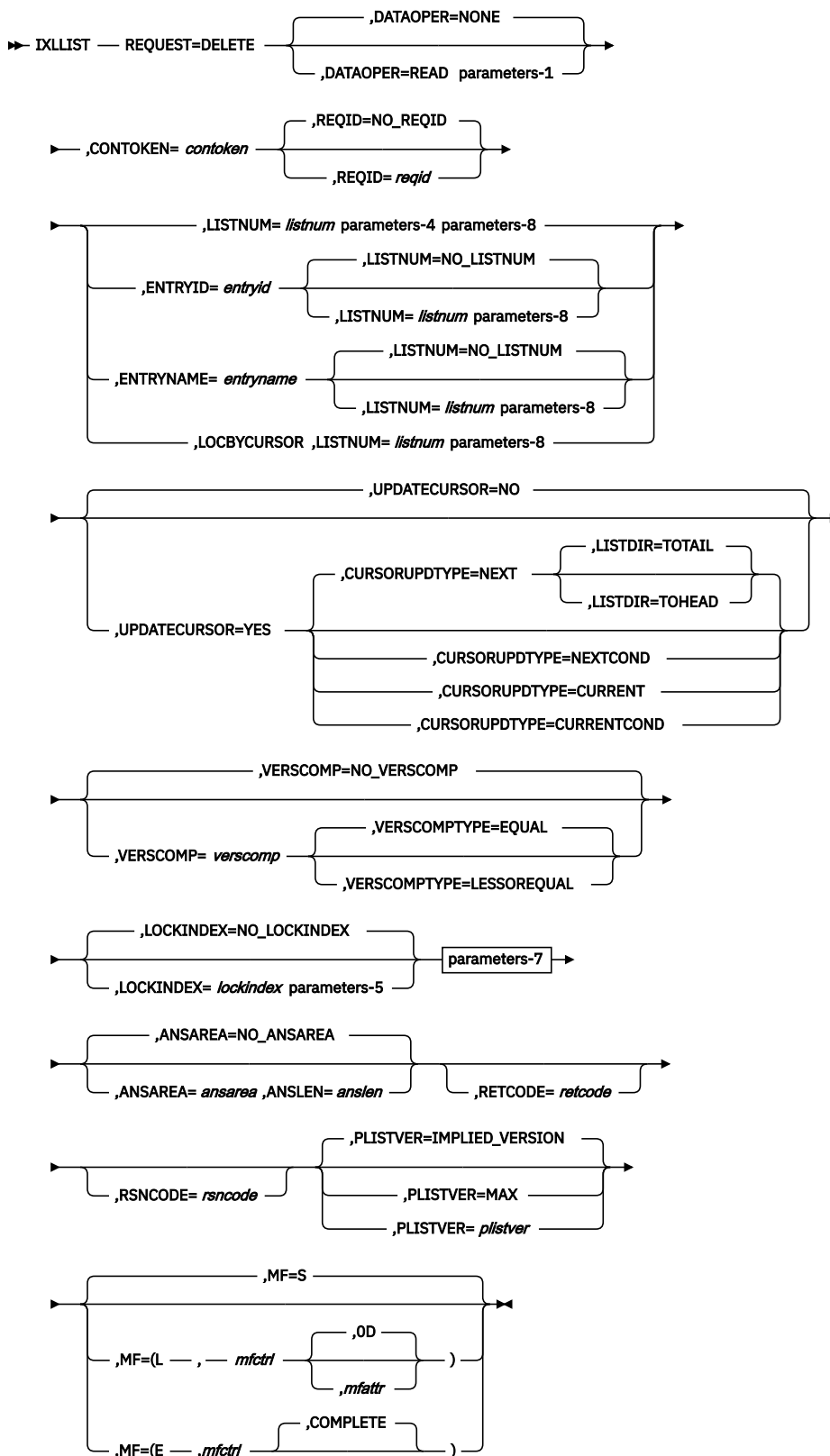
- List entry controls, the number of list entries or elements remaining on the list, and the total number of allocated entries in the structure are returned to the answer area (ANSAREA).
- Entry data is returned to the buffer (BUFFER) or buffer list (BUFLIST). (DATAOPER=READ must be specified.)
- Adjunct data is returned to the adjunct area (ADJAREA). (DATAOPER=READ must be specified.)

Additionally, you can perform locking functions with a DELETE request by specifying LOCKINDEX. The lock entry designated by LOCKINDEX will be operated on as specified by LOCKOPER.

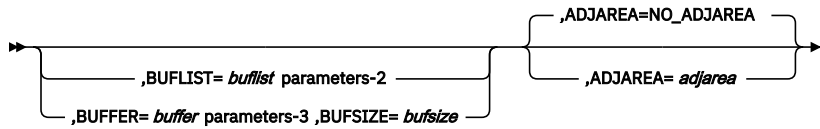
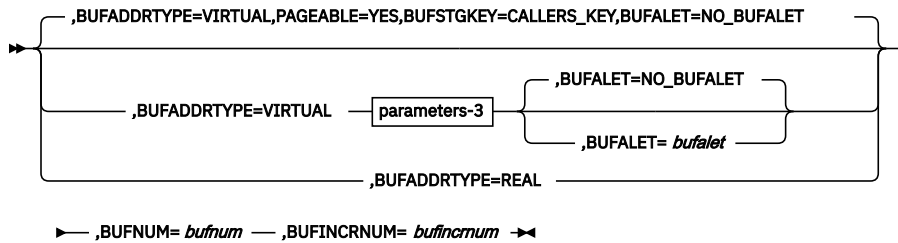
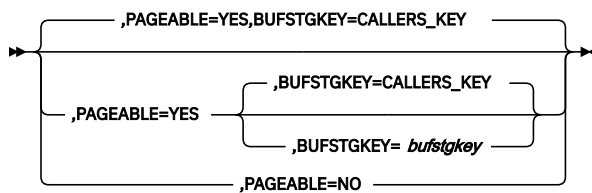
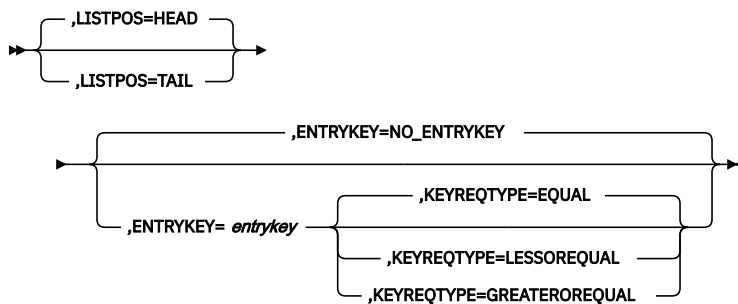
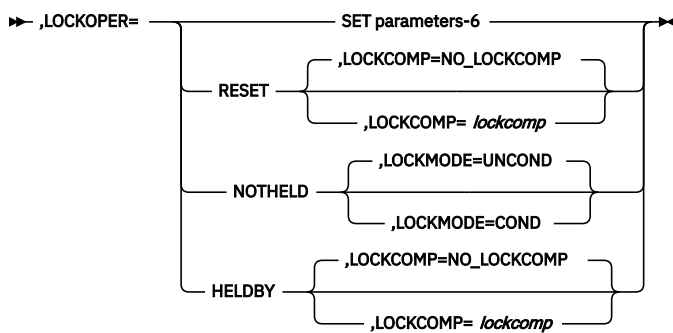
### Syntax Diagram

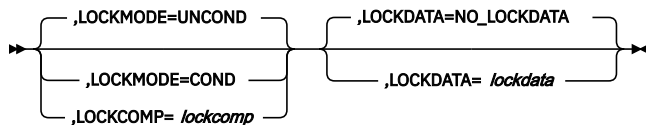
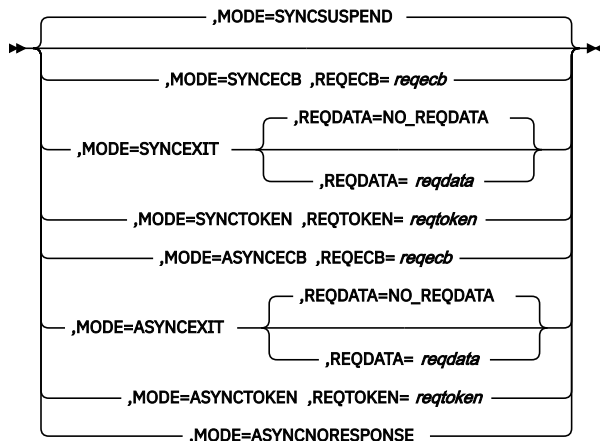
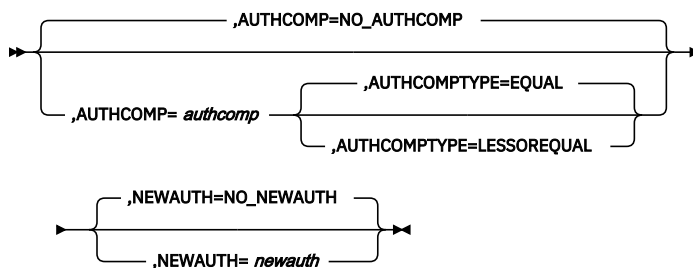
The syntax diagram for IXLLIST REQUEST=DELETE is as follows:

## main diagram





**parameters-1****parameters-2****parameters-3****parameters-4****parameters-5**

**parameters-6****parameters-7****parameters-8****Note:**

1. In the | parameters-1 | fragment, one of the following must be specified:

- BUFLIST=*buflist*
- BUFFER=*buffer*
- ADJAREA=*adjarea*

In addition, ADJAREA=*adjarea* can be specified with either BUFLIST=*buflist* or BUFFER=*buffer*.

2. If MODE=SYNCTOKEN or MODE=ASYNCTOKEN is specified, ANSAREA= *ansarea* and ANSLLEN=*anslen* are required.
3. If MODE=ASYNCRESPONSE is specified, BUFFER, BUFLIST, and LOCKINDEX may not be specified.

## Parameter Descriptions

The parameter descriptions for REQUEST=DELETE are listed in alphabetical order. Default values are underlined:

**REQUEST=DELETE**

Use this input parameter to delete a single list entry from the list structure.

**,ADJAREA=NO\_ADJAREA****,ADJAREA=adjarea**

Use this output parameter to specify a storage area to contain the adjunct data that was read from the designated entry.

Specify ADJAREA only for structures that support adjunct data. (Adjunct areas for a structure are established through the IXLCONN macro.)

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 64-byte area to contain the adjunct data.

**,ANSAREA=NO\_ANSAREA****,ANSAREA=ansarea**

Use this output parameter to specify an answer area to contain information returned from the request. The format of the answer area is described by mapping macro IXLYLAA.

On a successful completion of the request, the following information is returned to the answer area:

- List entry controls of the designated entry (field LAALCTL).
- The number of list entries or elements remaining on the list (field LAALISTCNT).
- The total number of allocated entries in the structure (field LAATOTALCNT).
- The total number of allocated elements in the structure (field LAATOTALELECNT).

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an area (with a length of ANSLEN) where the information about the completion of the request will be put.

**,ANSLEN=anslen**

Use this input parameter to specify the size of the storage area specified by ANSAREA.

Check the prologue of the IXLYLAA mapping macro for the minimum required size of the answer area, or use the length field (LAA\_LEN) in the LAA.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 2-byte field that contains the size, in bytes, of the answer area.

**,AUTHCOMP=NO\_AUTHCOMP****,AUTHCOMP=authcomp**

Use this input parameter to specify a value to be compared to the list authority value of the list specified by LISTNUM. You must supply the LISTNUM parameter.

If the comparison does not meet the condition specified by the AUTHCOMPTYPE parameter (EQUAL or LESSOREQUAL), the request fails.

**Note:** The AUTHCOMP parameter is valid only for list structures allocated in a coupling facility with CFLEVEL=1 or higher.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12 2 to ) of the 16-byte field that contains the list authority value.

**,AUTHCOMPTYPE=EQUAL****,AUTHCOMPTYPE=LESSOREQUAL**

Use this input parameter specify how the list audit comparison is to be performed.

**EQUAL**

The list authority for the list specified by LISTNUM must be equal to the value specified for AUTHCOMP.

**LESSOREQUAL**

The list authority for the list specified by LISTNUM must be less than or equal to the value specified for AUTHCOMP.

**Note:** The AUTHCOMPTYPE parameter is valid only for list structures allocated in a coupling facility with CFLEVEL=1 or higher.

**,BUFADDRTYPE=VIRTUAL****,BUFADDRTYPE=REAL**

Use this input parameter to specify whether the buffer addresses specified in the BUFLIST list are virtual storage or real storage addresses.

**VIRTUAL**

The buffer addresses are virtual storage addresses. The virtual storage can be pageable or nonpageable. See the PAGEABLE parameter for information about managing storage binds when specifying virtual storage addresses.

**REAL**

The buffer addresses are real storage addresses.

It is the caller's responsibility to manage the binds between the data buffer virtual storage and the real storage addresses provided. The caller must ensure that the data buffer virtual storage remains bound to the real storage addresses provided until the request completes.

**,BUFALET=NO\_BUFALET****,BUFALET=*bufalet***

Use this input parameter to specify an access list entry token (ALET) to be used in referencing all of the buffers specified by BUFLIST.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 4-byte field that contains the ALET.

**,BUFFER=*buffer***

Use this output parameter to specify a buffer area to hold the entry data that is read from the designated entry. Only 31-bit addressable (below 2GB) virtual storage areas are supported for the BUFFER specification.

You can define the buffer size to be a total of up to 65536 bytes. Depending on the size you select, the following restrictions apply:

- If you specify a buffer size of less than or equal to 4096 bytes, you must ensure that the buffer:
  - Is 256, 512, 1024, 2048, or 4096 bytes.
  - Starts on a 256-byte boundary.
  - Does not cross a 4096-byte boundary.
- If you specify a buffer size of greater than 4096 bytes, you must ensure that the buffer:
  - Is a multiple of 4096 bytes.
  - Is less than or equal to 65536 bytes.
  - Starts on a 4096-byte boundary.

See the BUFSIZE parameter description for defining the size of the buffer.

**Note:** You cannot code BUFFER with MODE=ASYNCRESPONSE.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an area (with a size of BUFSIZE) that will contain the entry data.

**,BUFINCRNUM=*bufincrnum***

Use this input parameter to specify the number of 256-byte segments comprising each buffer in the BUFLIST list.

Valid BUFINCRNUM values are 1, 2, 4, 8, or 16, which correspond to BUFLIST buffer sizes of 256, 512, 1024, 2048, and 4096 bytes respectively.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 1-byte field that contains 1, 2, 4, 8, or 16.

**,BUFLIST=*buflist***

Use this output parameter to specify a list of buffers to hold the entry data that is read from the designated entry. BUFLIST specifies a 128-byte storage area that consists of a list of 0 to 16 buffer addresses.

**The 128-byte storage area must:**

- Consist of 0 to 16 elements.
- Each element must consist of an 8-byte field in which:
  - The left (high-order) four bytes are reserved and
  - The right (low-order) four bytes contain the address of a buffer.

**The BUFLIST buffers must:**

- Reside in the same address space or data space.
- Be the same size: either 256, 512, 1024, 2048, or 4096 bytes.
- Start on a 256-byte boundary and not cross a 4096-byte boundary.

**Note:** The buffers do not have to be contiguous in storage. List services treats BUFLIST buffers as a single buffer even if the buffers are not contiguous.

See the BUFNUM and BUFINCRNUM parameter descriptions for specifying the number and size of buffers.

**Note:** You cannot code BUFLIST with MODE=ASYNCRESPONSE.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 128-byte area that contains a list of buffer addresses.

**,BUFNUM=*bufnum***

Use this input parameter to specify the number of buffers in the BUFLIST list. Valid BUFNUM values are 0 - 16. A value of zero indicates that no data is to be read into the buffers.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 1-byte field that contains the number of buffers (0 - 16) in the list (BUFLIST).

**,BUFSIZE=*bufsize***

Use this input parameter to specify the size of the BUFFER area. See the BUFFER parameter description for valid buffer sizes.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a fullword field that contains the size of the buffer (BUFFER) in bytes.

**,BUFSTGKEY=CALLERS\_KEY****,BUFSTGKEY=*bufstgkey***

Use this input parameter to specify a storage key that you define and use when referencing the buffers specified by BUFLIST or the buffer that is specified by BUFFER.

If you do not specify BUFSTGKEY, or if you specify BUFSTGKEY=CALLERS\_KEY, all references to one or more buffers are performed by using the caller's PSW key.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of an 8-bit field that contains the storage key in the format B'kkkkxxxx', where kkkk is the key and xxxx is ignored.

**,CONTOKEN=*contoken***

Use this input parameter to specify the connect token that was returned by the IXLCONN service. The connect token uniquely identifies your connection to the list structure, and must be specified on each IXLLIST invocation.

The connect token is available in the IXLCONN answer area that is mapped by IXLYCONA.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 16-byte field that contains the connect token.

**,CURSORUPDTYPE=NEXT****,CURSORUPDTYPE=NEXTCOND****,CURSORUPDTYPE=CURRENT****,CURSORUPDTYPE=CURRENTCOND**

Use this input parameter to specify how the list cursor is to be updated when UPDATECURSOR=YES is specified.

**Note:** The NEXTCOND, CURRENT, and CURRENTCOND values are valid only for list structures allocated in a coupling facility with CFLEVEL=1 or higher.

#### **CURSORUPDTYPE=NEXT**

Update the list cursor to point to the list entry before or after the target entry.

- For MOVE requests that specify DATAOPER=WRITE and ENTRYTYPE=ANY and that result in the creation of a new entry, the cursor for the list specified by MOVETOLIST is updated. The direction of the cursor update depends on the value that is specified for MOVETOPOS.
- For all other requests, the cursor for the source list is updated. The direction of the cursor update depends on the value that is specified for LISTPOS or LISTDIR.

#### **CURSORUPDTYPE=NEXTCOND**

Update the list cursor to point to the list entry before or after the target entry only if the cursor points to the target list entry, and that list entry is moved or deleted.

Set the list cursor direction with SETCURSOR on a WRITE\_LCONTROLS request.

The list cursor is reset to binary zeros if either:

- The entry is the last entry on the list and the list cursor direction is set to a head-to-tail direction.
- The entry is the first entry on the list and the list cursor is set to a tail-to-head direction.

#### **CURSORUPDTYPE=CURRENT**

Set the list cursor to point to the list entry processed for the request.

If the list entry is deleted or moved to another list, then the list cursor is reset to binary zeros.

#### **CURSORUPDTYPE=CURRENTCOND**

Set the list cursor to point to the list entry processed only if the list cursor value currently is zero and the target list entry is not deleted or moved to another list.

If the list entry is deleted or moved to another list, then the list cursor will remain binary zeros.

#### **,DATAOPER=NONE**

#### **,DATAOPER=READ**

Use this input parameter to specify an operation to be performed on the designated entry in addition to deleting it.

##### **NONE**

No additional operation is performed.

##### **READ**

Read and delete the entry data and/or adjunct data:

- If you have specified BUFFER or BUFLIST, the entry data is read into whichever buffer area you have specified.
- If you have specified ADJAREA, the adjunct data is read into the adjunct area specified.

#### **,ENTRYID=entryid**

Use this input parameter to designate the entry identifier of the entry to be deleted.

Each entry identifier, which is assigned by list services when the entry is created, is unique within the structure.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 12-byte field that contains the entry identifier.

#### **,ENTRYKEY=NO\_ENTRYKEY**

#### **,ENTRYKEY=entrykey**

Use this input parameter with LISTNUM to designate the entry key of the entry to be deleted.

If there is a sublist of entries on the list all with the same key as the ENTRYKEY key, the LISTPOS parameter determines whether the entry at the HEAD or TAIL of the sublist is designated.

Specify ENTRYKEY only for structures that support keyed entries.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-byte field that contains the entry key.

**,ENTRYNAME=entryname**

Use this input parameter to designate the entry name of the entry to be deleted.

Specify ENTRYNAME only for structures that support named entries. Each entry name is unique within the structure.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-byte field that contains the entry name.

**,KEYREQTYPE=EQUAL**

**,KEYREQTYPE=LESSOREQUAL**

**,KEYREQTYPE=GREATEROREQUAL**

Use this input parameter to specify how, if at all, the entry key of the entry to be read can differ from the entry key that is specified by ENTRYKEY. KEYREQTYPE applies only to locating an existing entry; it does not determine the key of a new entry.

**EQUAL**

The entry must have a key that equals the ENTRYKEY key.

**LESSOREQUAL**

The entry must have a key that is less than or equal to the ENTRYKEY key.

**GREATEROREQUAL**

The entry must have a key that is greater than or equal to the ENTRYKEY key.

**Note:**

1. When no entries on the list meet the requirements of the KEYREQTYPE, the request is failed with a return code X'8' and reason code IXLRSCODEOENTRY.
2. When LESSOREQUAL or GREATEROREQUAL is specified, if no entries on the list have an entry key value equal to the specified value, but entries exist with an entry key value greater than (if GREATEROREQUAL was specified), or less than (if LESSOREQUAL was specified) the entry key value that is specified, the entry with an entry key value closest to the value specified will be selected. When multiple entries have the same entry key value, LISTPOS is used to resolve whether the first, or last entry with the entry key value is selected.

**,LISTDIR=TOTAIL**

**,LISTDIR=TOHEAD**

Use this input parameter with UPDATECURSOR to specify the direction in which the list cursor is moved as a result of this request. See the UPDATECURSOR parameter for a full description of LISTDIR.

**,LISTNUM=NO\_LISTNUM**

**,LISTNUM=listnum**

Use this input parameter to specify the number of the list on which the entry to be deleted resides. Use LISTNUM in the following ways:

- With LISTPOS and/or ENTRYKEY to identify the entry to be deleted
- With either ENTRYID or ENTRYNAME to verify that the designated entry resides on the LISTNUM list before proceeding with the request
- With LOCBYCURSOR to identify the entry to be deleted

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 4-byte field that contains the number of the list.

**,LISTPOS=HEAD**

**,LISTPOS=TAIL**

Use this input parameter with LISTNUM to designate the entry to be deleted. The designated entry is at the HEAD or the TAIL of a list or sublist:

- If you specify ENTRYKEY and the list contains a sublist of one or more entries with the same key (or that satisfies the KEYREQTYPE criteria), then:

- LISTPOS=HEAD designates the entry at the head of the sublist.
- LISTPOS=TAIL designates the entry at the tail of the sublist.
- If you do not specify ENTRYKEY, then:
  - LISTPOS=HEAD designates the entry at the head of the list.
  - LISTPOS=TAIL designates the entry at the tail of the list.

**,LOCBYCURSOR**

Use this input parameter to designate the list entry to be deleted. The designated entry is the entry to which the LISTNUM list cursor is pointing.

Be aware that the list cursor could have been reset to zeros by a previous request that, for instance, deleted or moved the entry to which the list cursor points, or updated the list cursor after the last entry on the list had been processed. See *z/OS MVS Programming: Sysplex Services Guide* for more information about using the list cursor.

**,LOCKCOMP=NO LOCKCOMP****,LOCKCOMP=lockcomp**

This parameter has slightly different meanings based on the value that is specified for the LOCKOPER parameter. Generally, this input parameter specifies a connection identifier to be verified as the owner of the lock specified by LOCKINDEX. This verification is a prerequisite to the successful completion of the request.

When LOCKCOMP is specified, the completion of the request is conditional on there being no contention for the lock. If contention exists, the request fails .

The connection identifier is available from the IXLCONN answer area, which is mapped which is by IXLYCONA, in field CONACONID.

The effect of LOCKCOMP is based on the LOCKOPER value specified. See the description under LOCKOPER to see how each request is affected by this parameter.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 1-byte field that contains the connection identifier.

**,LOCKDATA=NO LOCKDATA****,LOCKDATA=lockdata**

Use this input parameter to specify information that is to be passed to your notify exit when another user requests the LOCKINDEX lock after you have obtained the lock by using LOCKOPER=SET. This user-defined information will be passed to your notify exit when the other user issues a request specifying either one of the following:

- LOCKOPER=SET
- LOCKOPER=NOTHELD with LOCKMODE=UNCOND

**To Code:** Specify the RS-type name or address (by using a register 2 - 12 2 to ) of an 8-byte field that contains the user-defined information.

**,LOCKINDEX=NO LOCKINDEX****,LOCKINDEX=lockindex**

Use this input parameter to specify the index of the lock to be operated on as specified by LOCKOPER. The lock indexes begin with zero.

**Note:** You cannot code LOCKINDEX with MODE=ASYNCSNORESPONSE.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 4-byte field that contains the lock index.

**,LOCKMODE=UNCOND****,LOCKMODE=COND**

Use this input parameter to specify how contention for the lock that is specified by LOCKINDEX should be handled if your request causes lock contention.



**UNCOND**

If the specified lock is held by another user, your request is either:

- Suspended until the lock becomes available (if MODE=SYNCSUSPEND is specified).
- Performed asynchronously with notification to you when the request completes (if any MODE value other than SYNCSUSPEND is specified).

**COND**

The lock operation is performed conditionally; that is, only if there is no contention for the lock. If the specified lock is held by another user, the request is terminated with no change to the structure, and return and reason codes describing the termination are returned to the caller.

**,LOCKOPER=SET**

**,LOCKOPER=RESET**

**,LOCKOPER=NOTHELD**

**,LOCKOPER=HELDDBY**

Use this input parameter to specify the type of operation to be performed on the lock specified by LOCKINDEX.

**SET**

Set the lock.

- When LOCKCOMP is not specified, your connection gets the lock, providing no other connection holds the lock. If another connection holds the lock, the request either continues once the lock is released or fails, depending on the LOCKMODE value you specify.
- When LOCKCOMP is specified, if the connection specified by LOCKCOMP holds the lock, the lock is transferred to your connection.

**RESET**

Reset the lock.

- When LOCKCOMP is not specified, the lock is released if it is held by your connection.
- When LOCKCOMP is specified, the lock is released if it is held by the connection that is specified by LOCKCOMP.

**NOTHELD**

The request is performed only if the lock is not held by any connection. This option also ensures that the lock remains free for the duration of the request. If another connection holds the lock, your task is suspended until the lock is released or your request fails, depending on the LOCKMODE value you specify. See the LOCKMODE description for how to handle possible lock contention.

**HELDDBY**

Processing is determined by which connector is holding the lock.

- When LOCKCOMP is not specified, the request is performed only if your connection holds the lock.
- When LOCKCOMP is specified, the request is performed only if the lock is held by the connection that is specified by LOCKCOMP.

**,MF=S**

**,MF=(L,mfctrl)**

**,MF=(L,mfctrl,mfattr)**

**,MF=(L,mfctrl,0D)**

**,MF=(M,mfctrl)**

**,MF=(M,mfctrl,COMPLETE)**

**,MF=(M,mfctrl,NOCHECK)**

**,MF=(E,mfctrl)**

**,MF=(E,mfctrl,COMPLETE)**

**,MF=(E,mfctrl,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

#### **,mfctrl**

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

#### **,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

#### **,COMPLETE**

#### **,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

#### **COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if SMILE=*var* were an optional parameter and the default is SMILE=NO\_SMILE then it would not be documented. However, if the default was SMILE=:), then it would be documented because a value would be the default.

#### **NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

#### **,MODE=SYNCSUSPEND**

#### **,MODE=SYNCECB**

#### **,MODE=SYNCEXIT**

#### **,MODE=SYNCTOKEN**

#### **,MODE=ASYNCECB**

#### **,MODE=ASYNCEXIT**

#### **,MODE=ASYNCTOKEN**

#### **,MODE=ASYNCRESPONSE**

Use this input parameter to specify:

- Whether the request is to be performed synchronously or asynchronously
- How you want to be notified of request completion

#### **MODE=SYNCSUSPEND**

The request is suspended until it can complete synchronously. To use this option, your program must be enabled for I/O and external interrupts.

#### **MODE=SYNCECB**

The request processes synchronously if possible. If the request processes asynchronously, the ECB specified by REQECB is posted when the request completes.

**MODE=SYNCEXIT**

The request processes synchronously if possible. If the request processes asynchronously, your complete exit is given control when the request completes.

**MODE=SYNCTOKEN**

The request processes synchronously if possible. If the request processes asynchronously, an asynchronous request token is returned in the area specified by REQTOKEN on invocation of the request. Use the returned request token on the IXLFCOMP macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=SYNCTOKEN is specified.

**MODE=ASYNCECB**

The request processes asynchronously. The ECB specified by REQECB is posted when the request completes.

**MODE=ASYNCEXIT**

The request processes asynchronously. Your complete exit is given control when the request completes.

**MODE=ASYNCTOKEN**

The request processes asynchronously. An asynchronous request token is returned in the area specified by REQTOKEN upon invocation of the request. Use the returned request token on the IXLFCOMP macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=ASYNCTOKEN is specified.

**MODE=ASYNCSNORESPONSE**

The request processes asynchronously. No notification of request completion is provided. The answer area (ANSAREA) fields will not contain valid information when this mode is specified.

**Note:** You cannot code MODE=ASYNCSNORESPONSE with LOCKINDEX, BUFFER, or BUFLIST.

**,NEWAUTH=NO\_NEWAUTH****,NEWAUTH=newauth**

Use this input parameter to specify a list authority value for the list specified by LISTNUM. If NEWAUTH is omitted, the list authority for the designated list is unchanged.

When the structure is allocated, the authority value for each list is initialized to binary zeros.

**Note:** The NEWAUTH parameter is valid only for list structures allocated in a coupling facility with CFLEVEL=1 or higher, except on WRITE\_LCONTROLS requests.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12 2 to ) of the 16-byte field that contains the list authority value.

**,PAGEABLE=YES****,PAGEABLE=NO**

Use this input parameter to identify whether the storage areas specified by BUFFER or BUFLIST are in pageable or potentially pageable storage.

**YES**

Specify this option to indicate that the BUFFER or BUFLIST buffers reside in pageable virtual storage. XES performs the required page fixing to fix the buffers in real storage while the cache or list request transfers data to or from the coupling facility.

This includes storage obtained from pageable subpools, disabled reference (DREF) storage, and may include storage that has the potential to become pageable during the processing of a request. (An example is address space storage owned by any swappable address space, for which a PGSER FIX has been successfully processed, but for which the owning address space gets swapped during processing of a cache or list request.) This does not include implicitly non-pageable storage (for example, storage obtained from non-pageable subpools).

The system takes responsibility for managing binds to central storage for the duration of the cache or list request, regardless of what address space owns the storage or whether the storage-owning address space is swappable or nonswappable. The storage can be owned by any address space.

High shared virtual storage areas (above 2GB) may not be used.

**NO**

Specify this option to indicate that the BUFFER or BUFLIST buffers reside in non-pageable virtual storage. XES does not page fix the buffers in real storage.

This includes implicitly non-pageable storage areas (for example, storage obtained from non-pageable subpools), and may include storage that has the potential to become pageable during the processing of a request (An example is address space storage owned by any swappable address space, for which a PGSER FIX has been successfully processed, but for which the owning address space gets swapped-out during processing of a cache or list request.)

The system takes responsibility for managing binds to central storage for the duration of the cache or list request, if and only if the non-pageable storage is owned by either the requester's address space or the connector's address space. If the storage is owned by any other address space, then the invoker is responsible for ensuring that the virtual storage remains non-pageable for the duration of the request (including the case in which the storage is owned by a swappable address space that is swapped during processing of an IXLCACHE or IXLLIST request). Subject to this consideration, the storage can be owned by any address space. See *z/OS MVS Programming: Sysplex Services Guide*.

**,PLISTVER=IMPLIED\_VERSION****,PLISTVER=MAX****,PLISTVER=plistver**

Use this input parameter to specify the version of the macro. See “[Understanding IXLLIST Version Support](#)” on page 965 for a description of the options available with PLISTVER.

**,REQDATA=NO\_REQDATA****,REQDATA=reqdata**

Use this input parameter with MODE=SYNCEXIT or MODE=ASYNCEXIT to pass any data you choose to the complete exit. The exit will get control only if the request is processed asynchronously.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the data to be passed to the complete exit.

**,REQECB=req ECB**

Use this output parameter with either MODE=SYNCECB or MODE=ASYNCECB to specify the address of an ECB, which is to be posted when the request completes if the request was processed asynchronously.

Before coding REQECB, you must ensure that:

- You initialize the ECB before you issue the request.
- The ECB resides in either common storage or the home address space where IXLCONN was issued.
- Any tasks that wait for the ECB to be posted reside in the home address space where IXLCONN was issued.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that contains the address of the ECB to be posted when the request completes. The ECB must be aligned on a fullword boundary.

**,REQID=NO\_REQID****,REQID=reqid**

Use this input parameter to specify a user-defined request identifier to be associated with the request. You can specify this request identifier on the IXLPURGE macro to cancel a request that has not yet been processed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the user-defined request identifier.

**,REQTOKEN=reqtoken**

Use this output parameter with either MODE=SYNCTOKEN or MODE=ASYNCTOKEN to specify the address of a storage area to receive the request token that is returned when the request will be

processed asynchronously. This token, which uniquely identifies the request, must be used as input to the IXLFCOMP macro, which you use to determine if the request has completed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field where the system will put the request token.

**,RETCODE=retcode**

Use this output parameter to specify a field to contain the return code. (The return code is also returned in register 15.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the return code when the request has completed.

**,RSNCODE=rsncode**

Use this output parameter to specify a field to contain the reason code returned, if applicable. (The reason code is also returned in register 0.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the reason code (if any) when the request has completed.

**,UPDATECURSOR=NO**

**,UPDATECURSOR=YES**

Use this input parameter to specify whether the list cursor of the list containing the designated entry is to be updated to point to another entry on the list.

**NO**

The list cursor is not updated.

**YES**

The list cursor is updated to point to the entry that is adjacent to the entry that was deleted.

- If LISTDIR=TOHEAD, the list cursor will point to the adjacent entry that is closest to the head of the list.
- If LISTDIR=TOTAIL, the list cursor will point to the adjacent entry that is closest to the tail of the list.

The list cursor is set to binary zeros if its new position would be before the first entry or after the last.

**,VERSCOMP=NO\_VERSCOMP**

**,VERSCOMP=verscomp**

Use this input parameter to specify a version number to be compared to the version number of the designated entry to be deleted. The entry is deleted only if its version number meets the condition specified by the VERSCOMPTYPE parameter. If the designated entry does not have the specified version number, the request is terminated with no change to the structure.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-byte field that contains the version number.

**,VERSCOMPTYPE=EQUAL**

**,VERSCOMPTYPE=LESSOREQUAL**

Use this input parameter to specify how a list entry version comparison as specified by VERSCOMP is to be performed.

**Note:** The VERSCOMPTYPE parameter is valid only for list structures allocated in a coupling facility with CFLEVEL=1 or higher.

**VERSCOMPTYPE=EQUAL**

The version number for the list entry must be equal to the value specified for VERSCOMP.

**VERSCOMPTYPE=LESSOREQUAL**

The version number for the list entry must be less than or equal to the value specified for VERSCOMP.

## ABEND Codes

Abend X'026' (See *z/OS MVS System Codes* for more information on this abend.)

## Return and Reason Codes

When the system returns control to your program:

- GPR 15 (and RETCODE, if specified) contains a return code.
- If the return code is not zero, GPR 0 (and RSNCODE, if specified) contains a reason code.

**Note:** The return and reason codes will also be put into the answer area (mapped by IXLYLAA), if there is one, when the request completes.

The IXLYCON macro provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

<b>0</b>	IXLRETCODEOK
<b>4</b>	IXLRETCODEWARNING
<b>8</b>	IXLRETCODEPARMERROR
<b>C</b>	IXLRETCODEENVERROR
<b>10</b>	IXLRETCODECOMPERROR

The following table contains hexadecimal return and reason codes, the equate symbols associated with each reason code, and the meaning and suggested action for each return and reason code.

Table 57. Return and Reason Codes for IXLLIST REQUEST=DELETE Macro		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0	None.	<p><b>Meaning:</b> If you specified a MODE value of ASYNCECB, ASYNCEXIT, ASYNCTOKEN, or ASYNCNORESPONSE, the request has been successfully initiated. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed. For any other MODE value, the request has successfully completed.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=ASYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=ASYNCEXIT, the connection's complete exit will be given control when the request has completed.</li> <li>• If you specified MODE=ASYNCTOKEN, use REQTOKEN when calling IXLFCOMP to determine when the request has completed.</li> </ul>

Table 57. Return and Reason Codes for IXLLIST REQUEST=DELETE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	xxxx0402	<p><b>Equate Symbol:</b> IXLRNCODEASYNCH</p> <p><b>Meaning:</b> The request could not be processed synchronously. It will be processed asynchronously. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed.</p> <ul style="list-style-type: none"> <li>• If MODE=SYNCECB was specified, the ECB specified for REQECB will be posted when the request has finished.</li> <li>• If MODE=SYNCEXIT was specified, the connection's complete exit will be given control when the request has finished.</li> <li>• If MODE=SYNCTOKEN was specified, an asynchronous request token was returned in the area specified by REQTOKEN. This request token may be specified on the IXLFCOMP macro to determine when the request has finished.</li> </ul> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=SYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=SYNCEXIT, the connection's complete exit will be given control when the request has completed.</li> <li>• If you specified MODE=SYNCTOKEN, use REQTOKEN when calling IXLFCOMP to determine when the request has completed.</li> </ul>

Table 57. Return and Reason Codes for IXLLIST REQUEST=DELETE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	xxxx040D	<p><b>Equate Symbol:</b> IXLRNCODEBADREADADJDATA</p> <p><b>Meaning:</b> Program error. The request specified that adjunct data was to be read, but the storage area specified by ADJAREA is not addressable. All other requested data was retrieved, and the designated entry was deleted. The following fields in the answer area have been filled in:</p> <ul style="list-style-type: none"> <li>• LAATOTALCNT - The total count of allocated entries in the list structure.</li> <li>• LAATOTALELECNT - The total count of allocated elements in the list structure.</li> <li>• LAALISTCNT - The count of entries or elements residing on the processed list.</li> <li>• LAALCTL (if the request completed prematurely as well) - The list entry controls for the first UNPROCESSED entry in the list sequence.</li> </ul> <p><b>Note:</b> Only the adjunct data for the first entry in the list is placed in the ADJAREA. The buffers should contain the adjunct data for the other entries in the list.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the ADJAREA address.</li> <li>• ADJAREA must be addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are calling IXLLIST while disabled, ADJAREA must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLLIST in AR-mode and you specified the ADJAREA address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are calling IXLLIST in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLIST macro.</li> </ul> <p>Correct the address specified by ADJAREA, and rerun the request asking for adjunct data only.</p>
4	xxxx0410	<p><b>Equate Symbol:</b> IXLRNCODELOCKCOND</p> <p><b>Meaning:</b> For a LOCKOPER=HELD BY request, or a LOCKMODE=COND request, or a request that specified LOCKCOMP, the request could not be completed successfully because the specified lock is not currently held as required. The connection identifier of the lock owner is returned in the answer area (LAACONID field).</p> <p><b>Action:</b> Retry the request, or obtain the lock as required, and retry the request. If you are unable to get the lock, check the ID in the LAACONID field and determine if some recovery is necessary.</p>



Table 57. Return and Reason Codes for IXLLIST REQUEST=DELETE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	xxxx0412	<p><b>Equate Symbol:</b> IXLRSNCODELOCKHELDDBYSYS</p> <p><b>Meaning:</b> The lock is not held by any connection, but instead is held by the system. For a request that specified any of the following, the request could not be completed successfully because the specified lock is not currently held as required:</p> <ul style="list-style-type: none"> <li>• LOCKOPER=SET or LOCKOPER=NOTHELD with either of: <ul style="list-style-type: none"> <li>– LOCKMODE=COND</li> <li>– LOCKCOMP</li> </ul> </li> <li>• LOCKOPER=HELDDBY</li> </ul> <p><b>Action:</b> Retry the request, or obtain the lock as required, and retry the request.</p>
8	xxxx0801	<p><b>Equate Symbol:</b> IXLRSNCODEBADPARMLIST</p> <p><b>Meaning:</b> Program error. The parameter list for this request is not addressable. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Verify that:</p> <ul style="list-style-type: none"> <li>• The parameter list address is uncorrupted.</li> <li>• The parameter list is addressable in the caller's primary address space.</li> <li>• If you are issuing this macro while disabled, the parameter list resides in either page-fixed or DREF storage.</li> <li>• If you are issuing this macro in AR-mode and you specified the parameter list address using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If you are issuing this macro in AR-mode, SYSSTATE ASCENV=AR must be issued before issuing this macro.</li> </ul>
8	xxxx0804	<p><b>Equate Symbol:</b> IXLRSNCODEBADVERSIONNUM (or IXLRSNCODEBADVERSION#)</p> <p><b>Meaning:</b> The version number in the macro parameter list is not compatible with the level of XES currently being used. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that your program did not overlay the parameter list storage.</li> <li>• Verify that your program was assembled with the correct macro library for the release of MVS on which your program is running.</li> </ul>

Table 57. Return and Reason Codes for IXLLIST REQUEST=DELETE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx080A	<p><b>Equate Symbol:</b> IXLRSNCODEBADCONTOKEN</p> <p><b>Meaning:</b> Program error. The specified connect token (CONTOKEN) was not valid for one of the following reasons:</p> <ol style="list-style-type: none"> <li>1. The user with the connection identifier represented by the token has disconnected from the structure.</li> <li>2. The connector's task (the task that issued IXLCONN) ended.</li> <li>3. The specified token is not the token that was returned from IXLCONN.</li> <li>4. The request was issued from an address space other than the address space in which IXLCONN was issued.</li> <li>5. The connect token was invalidated during rebuild.</li> <li>6. The connect token was invalidated by XES.</li> </ol> <p><b>Note:</b> The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Take the action with the corresponding meaning.</p> <ol style="list-style-type: none"> <li>1. Discontinue use of the structure. Perform recovery and cleanup for the structure and check the protocol for the use of the structure to determine why you did not know of the disconnection.</li> <li>2. Discontinue use of the structure. Perform recovery and cleanup for the structure and check the protocol for the use of the structure to determine why you did not know that the task ended.</li> <li>3. The contoken is returned in the CONACONTOKEN field of the answer area specified on the IXLCONN request.</li> <li>4. Issue your request from the same address space the IXLCONN was issued in.</li> <li>5. Wait for the rebuild to complete, and try again.</li> <li>6. Discontinue use of the structure. Perform recovery and cleanup for the structure.</li> </ol>
8	xxxx0824	<p><b>Equate Symbol:</b> IXLRSNCODEWRONGSTRTYPE</p> <p><b>Meaning:</b> Program error. The connection specified by CONTOKEN is not to a list structure.</p> <p><b>Action:</b> Specify the appropriate CONTOKEN value, and rerun your program. The CONTOKEN is returned in the answer area provided on the IXLCONN invocation. The type of structure built is passed on the TYPE parameter of the IXLCONN macro.</p>

Table 57. Return and Reason Codes for IXLLIST REQUEST=DELETE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0825	<p><b>Equate Symbol:</b> IXLRSNCODENOENTRY</p> <p><b>Meaning:</b> Program error. The designated list entry does not exist; therefore, no entries were processed.</p> <p><b>Action:</b> None necessary. However, if this return code and reason code are unexpected, you should check the way the list entry was created. Examine the parameters specified for locating the list entry on the invocation of this macro.</p>
8	xxxx0833	<p><b>Equate Symbol:</b> IXLRSNCODEBADPGBLATTR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list is specified as being pageable (PAGEABLE=YES), but is not.</p> <p><b>Action:</b> Change the buffer area(s) to pageable storage, or specify PAGEABLE=NO. See the PAGEABLE parameter description for specification instructions.</p>
8	xxxx0834	<p><b>Equate Symbol:</b> IXLRSNCODEBADNONPGBLATTR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list, is specified as being nonpageable (PAGEABLE=NO), but is either pageable or not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The type of storage in the buffer area corresponds to the value you specified on the PAGEABLE parameter. See the PAGEABLE parameter description for specification instructions.</li> <li>• The correct buffer address was used.</li> <li>• The buffer area was not previously freed.</li> <li>• If you are calling IXLLIST while disabled, the buffers must reside in either page-fixed or DREF storage.</li> <li>• The buffer areas were allocated in a storage key that matches the key specified by the BUFSTGKEY parameter, or, if BUFSTGKEY is omitted, the caller's PSW key.</li> <li>• If BUFLIST was specified and your program is running in AR-mode: <ul style="list-style-type: none"> <li>– If the BUFFER or BUFLIST parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>– The BUFALET specification is correct.</li> <li>– You specified SYSSTATE ASCENV=AR before issuing the IXLLIST macro.</li> </ul> </li> </ul>

Table 57. Return and Reason Codes for IXLLIST REQUEST=DELETE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0835	<p><b>Equate Symbol:</b> IXLRNCODEBADDATAADDR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list, is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The correct buffer address was used.</li> <li>• The buffer area was not previously freed.</li> <li>• The buffer area was allocated in a storage key that matches the key specified using the BUFSTGKEY parameter, or, if BUFSTGKEY is omitted, the caller's PSW key.</li> <li>• If BUFLIST was specified and your program is running in AR mode: <ul style="list-style-type: none"> <li>– The BUFALET specification is correct.</li> <li>– You specified SYSSTATE ASCENV=AR before issuing the macro.</li> </ul> </li> </ul>
8	xxxx0836	<p><b>Equate Symbol:</b> IXLRNCODEBADREALADDR</p> <p><b>Meaning:</b> Program error. Real storage addresses were provided in the BUFLIST list, but one of the buffers is not addressable in central storage.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that BUFADDRTYPE was specified as you intended.</li> <li>• Ensure that the buffer addresses specified by BUFLIST are valid.</li> </ul>
8	xxxx0838	<p><b>Equate Symbol:</b> IXLRNCODEBADANSAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by ANSAREA is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The answer area address specified by ANSAREA is valid.</li> <li>• If the caller is running in AR-mode and the ANSAREA parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• The address specified for ANSAREA is addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are calling IXLLIST while disabled, ANSAREA must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLLIST in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLIST macro.</li> </ul>

Table 57. Return and Reason Codes for IXLLIST REQUEST=DELETE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0839	<p><b>Equate Symbol:</b> IXLRSNCODEBADREQTOKENAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by REQTOKEN is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The request token area specified by REQTOKEN is valid.</li> <li>• If the caller is running in AR-mode and the REQTOKEN parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If you are calling IXLLIST while disabled, REQTOKEN must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLLIST in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLIST macro.</li> </ul>
8	xxxx083D	<p><b>Equate Symbol:</b> IXLRSNCODEBADANSLEN</p> <p><b>Meaning:</b> Program error. The length of the answer area, as specified by ANSLEN, is not sufficient to contain answer area information. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Increase the size of the answer area, and rerun your program. The minimum size of the ANSAREA is in the LAA_LEN field of the IXLYLAA mapping macro.</p>
8	xxxx083F	<p><b>Equate Symbol:</b> IXLRSNCODEBADENTRYVERSION</p> <p><b>Meaning:</b> The designated entry has a version number that failed the comparison specified by VERSCOMPTYPE. The list entry controls for the entry are returned in the answer area (field LAALCTL).</p> <p><b>Action:</b> None necessary. However, if this return code and reason code are unexpected, you might want to verify the value specified in VERSCOMP and look at the version returned in the answer area.</p>
8	xxxx0840	<p><b>Equate Symbol:</b> IXLRSNCODEBADENTRYLIST</p> <p><b>Meaning:</b> The entry designated by ENTRYID or ENTRYNAME exists in the structure, but does not reside on the list specified by LISTNUM. The list entry controls for the entry are returned in the answer area (field LAALCTL).</p> <p><b>Action:</b> None necessary. However, if you did not expect this return and reason code, you might want to verify what list this entry is on. Maybe the entry was moved, or you designated the wrong list in LISTNUM. Check the protocol for using this list.</p> <p>The information returned in the LAALCTL field of the answer area contains the number of the list that this list entry is on as well as other control information.</p>

Table 57. Return and Reason Codes for IXLLIST REQUEST=DELETE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0842	<p><b>Equate Symbol:</b> IXLRSNCODEPERSISTENTLOCK</p> <p><b>Meaning:</b> Program error. The request specifying LOCKOPER=SET with LOCKMODE=UNCOND, or LOCKOPER=NOTHELD with LOCKMODE=UNCOND failed because the lock is held by a connection that is in the failed-persistent state. The connection identifier of the lock owner is returned in the answer area (field LAACONID).</p> <p><b>Action:</b> Either perform recovery for the connection, or wait until recovery for the connection is performed.</p>
8	xxxx0845	<p><b>Equate Symbol:</b> IXLRSNCODENONAMES</p> <p><b>Meaning:</b> Program error. A list entry was designated by entry name, but the structure does not support entry names.</p> <p><b>Action:</b> Ensure that you are connected to the intended structure. Ensure that the correct specification was made for REFOPTION on the IXLCONN macro. You may want to issue IXLMG to get more information about the structure.</p>
8	xxxx0846	<p><b>Equate Symbol:</b> IXLRSNCODEBADLOCKINDEX</p> <p><b>Meaning:</b> Program error. The specified LOCKINDEX exceeds the size of the lock table for the structure.</p> <p><b>Action:</b> Correct LOCKINDEX to specify an index that is contained within the lock table. The maximum value for the LOCKINDEX is one less than the value specified by the LOCKENTRIES parameter on the IXLCONN macro.</p>
8	xxxx0847	<p><b>Equate Symbol:</b> IXLRSNCODEBADLISTNUMBER</p> <p><b>Meaning:</b> Program error. The specified LISTNUM value exceeds the number of lists for the structure.</p> <p><b>Action:</b> Correct LISTNUM to specify a list number that exists in the structure. The number of lists allocated for a structure is determined on the LISTHEADERS parameter of the IXLCONN macro. The list numbers go from 0 to n-1, where n is the number of lists specified.</p>
8	xxxx0848	<p><b>Equate Symbol:</b> IXLRSNCODEBADRESET</p> <p><b>Meaning:</b> Program error. LOCKOPER=RESET was specified for a lock not currently held by the invoker. The value of the connection ID holding the lock is returned in the answer area (field LAACONID).</p> <p><b>Action:</b> Check your code to ensure that your connection did not already reset the lock.</p>

Table 57. Return and Reason Codes for IXLLIST REQUEST=DELETE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx084A	<p><b>Equate Symbol:</b> IXLRNOCODENOKEYS</p> <p><b>Meaning:</b> Program error. The structure does not support the use of entry keys. The IXLLIST request type either required the structure to support entry keys or designated a list entry or list position by list number and entry key.</p> <p><b>Action:</b> Ensure that you are connected to the intended structure. The use of keys for a structure is determined by the REFOPTION keyword on the IXLCONN macro.</p>
8	xxxx084B	<p><b>Equate Symbol:</b> IXLRNOCODENOLOCKS</p> <p><b>Meaning:</b> Program error. A locking operation was requested, but the structure does not contain a lock table.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• You are connected to the intended structure.</li> <li>• You intended to perform a locking operation.</li> </ul> <p>The use of locks is determined by the LOCKENTRIES keyword on the IXLCONN macro.</p>
8	xxxx0851	<p><b>Equate Symbol:</b> IXLRNOCODENOSUSPENDISABLE</p> <p><b>Meaning:</b> Program error. The request failed because MODE=SYNCSUSPEND was specified, but the caller is disabled and cannot be suspended.</p> <p><b>Action:</b> Either specify another MODE value, or become enabled (release the CPU lock), then reissue the request.</p>
8	xxxx0854	<p><b>Equate Symbol:</b> IXLRNOCODEBADLOCKCOMP</p> <p><b>Meaning:</b> Program error. The connection identifier specified for LOCKCOMP is not valid.</p> <p><b>Action:</b> The connection identifier is returned in the answer area (mapped by IXLYCONA) when the IXLCONN macro was issued.</p>
8	xxxx0859	<p><b>Equate Symbol:</b> IXLRNOCODEBADLISTAUTH</p> <p><b>Meaning:</b> The list authority specified for AUTHCOMP failed the comparison specified by AUTHCOMPTYPE for the list authority of the specified list. The current list authority (field LAALISTAUTH) and description (field LAALISTDESC) are returned in the answer area.</p> <p><b>Action:</b> None required; however you might want to take some action depending on your application. The list authority is set to binary zeros when the structure is allocated. When IXLLIST is issued with the NEWAUTH keyword, the list authority is changed if the correct comparison list authority value is specified. Check the answer area to determine the list authority value for the list specified. Verify that the correct list number was specified.</p>

Table 57. Return and Reason Codes for IXLLIST REQUEST=DELETE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0864	<p><b>Equate Symbol:</b> IXLRNCODEBADBUFSIZE</p> <p><b>Meaning:</b> Program error. The size of the BUFFER area or the buffer areas specified by BUFLIST is not large enough to contain the data for the first entry to be read in the list. No data is returned.</p> <p><b>Action:</b> If more space is available, specify a larger buffer size, or more buffers, and reissue the request. Check the information returned in the answer area (mapped by IXLYLAA) in the field LAALCTL (mapped by IXLYLCTL).</p>
8	xxxx0865	<p><b>Equate Symbol:</b> IXLRNCODEBADBUFSPEC</p> <p><b>Meaning:</b> Program error. There is an error in the buffer specification.</p> <p><b>Action:</b> Check the following:</p> <ul style="list-style-type: none"> <li>• If BUFLIST was specified, check the requirements for BUFLIST, BUFNUM, and BUFINCRNUM.</li> <li>• If BUFFER was specified, check the requirements for BUFFER and BUFSIZE.</li> <li>• Buffer pointer(s) in BUFLIST</li> <li>• Buffer boundaries.</li> </ul>
8	xxxx0866	<p><b>Equate Symbol:</b> IXLRNCODEBADBUFKEY</p> <p><b>Meaning:</b> Program error. The buffer storage key specified by BUFSTGKEY is incorrect or BUFSTGKEY was not specified and the PSW key does not match the key of the buffers.</p> <p>The data cannot be stored in the specified buffer area.</p> <p><b>Action:</b> Check the following:</p> <ul style="list-style-type: none"> <li>• Determine if the key of the storage being used for the buffers is different from the PSW key.</li> <li>• If BUFSTGKEY was specified, verify that the key was put in the correct bits (see the explanation of this parameter for more information).</li> <li>• If you are calling IXLLIST in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLIST macro.</li> </ul>
8	xxxx0867	<p><b>Equate Symbol:</b> IXLRNCODEBADBUFLIST</p> <p><b>Meaning:</b> Program error. The 128-byte storage area specified by BUFLIST is not addressable.</p> <p><b>Action:</b> Ensure that the address specified by BUFLIST is valid.</p>
8	xxxx08AD	<p><b>Equate Symbol:</b> IXLRNCODEBADHIGHSHAREDVIRT</p> <p><b>Meaning:</b> Program error. The request specified a high shared virtual storage area (above 2GB).</p> <p><b>Action:</b> None required.</p>



Table 57. Return and Reason Codes for IXLLIST REQUEST=DELETE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C06	<p><b>Equate Symbol:</b> IXLRSNCODENOCNN</p> <p><b>Meaning:</b> Environmental error. This system does not have connectivity to the coupling facility that contains the list structure. Possible reasons for this are:</p> <ul style="list-style-type: none"> <li>• The operator issued VARY PATH,OFFLINE.</li> <li>• The operator issued CONFIG CHP,OFFLINE.</li> <li>• Hardware errors to the coupling facility.</li> <li>• Facility or path failure to the coupling facility.</li> </ul> <p><b>Action:</b> Begin rebuilding the structure on a different coupling facility, or disconnect from the structure.</p>
C	xxxx0C13	<p><b>Equate Symbol:</b> IXLRSNCODEREQPURGED</p> <p><b>Meaning:</b> Environmental error. The request was purged prior to completion of the request. Possible reasons include:</p> <ul style="list-style-type: none"> <li>• The connector failed.</li> <li>• The connector disconnected.</li> <li>• The requestor failed.</li> <li>• The request was purged by IXLPURGE.</li> <li>• Requests were purged when the connector provided an IXLEERSP response for the Rebuild Stop or Rebuild Cleanup event.</li> <li>• The secondary address space was no longer valid.</li> </ul> <p><b>Action:</b> None if this is expected. Otherwise, determine why the connector failed.</p>
C	xxxx0C14	<p><b>Equate Symbol:</b> IXLRSNCODESTATUSUNKNOWN</p> <p><b>Meaning:</b> Environmental error. The request has completed, but the final disposition of the request cannot be determined. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the validity of your data by comparing the expected results with what is in the coupling facility.</li> <li>• Consider running rebuild processing to move the structure to attempt to protect it from further errors of this kind.</li> </ul>
C	xxxx0C25	<p><b>Equate Symbol:</b> IXLRSNCODESTRFAILURE</p> <p><b>Meaning:</b> Environmental error. The list structure failed prior to completion of the request.</p> <p><b>Action:</b> Either rebuild or disconnect from the structure.</p>

Table 57. Return and Reason Codes for IXLLIST REQUEST=DELETE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0CA0	<b>Equate Symbol:</b> IXLRSNCODEQUIESCEDSUSPENDFAIL <b>Meaning:</b> Environmental error. The request is failed because the structure is quiesced for a system-managed process and SUSPEND=FAIL is specified on the IXLCONN. <b>Action:</b> None, if this is expected.
C	xxxxFFFF	<b>Equate Symbol:</b> IXLRSNCODENOTAVAILABLE <b>Meaning:</b> Environmental error. XES functions are not available. This could occur because the coupling facility hardware necessary to provide XES functions is not present. <b>Action:</b> Re-IPL the system, or follow your particular failure management protocol.
10	xxxx10xx	<b>Meaning:</b> System error. XES processing failure. The state of the involved structure and the disposition of the request are unpredictable. The answer area (ANSAREA) fields do not contain valid information. <b>Action:</b> Contact the IBM support center.

## Chapter 58. IXLLIST REQUEST=DELETE\_ENTRYLIST

### Description

A DELETE\_ENTRYLIST request allows you to delete multiple entries that are designated in an entry list contained in the storage area specified by BUFLIST or BUFFER. To designate the entries to be deleted, put either their entry identifier or entry name into the buffer (BUFFER) or buffer list buffers (BUFLIST). The LISTTYPE parameter indicates how the entries to be deleted are listed in the buffer. They may be listed by either entry identifier or by entry name, but not a combination of the two.

You can also request that a subset of the entries identified in the entry list be deleted by specifying the indexes into the entry list identifying the entries with which processing should begin (FIRSTELEM) and end (LASTELEM).

You can also restrict processing to only those entries that satisfy either one or both of the following criteria:

- Reside on a certain list (which you specify using LISTNUM)
- Contain data that satisfies a version number comparison (which you specify using VERSCOMP).

If you do not specify either LISTNUM or VERSCOMP, every entry between FIRSTELEM and LASTELEM will be deleted.

With a coupling facility of CFLEVEL=1 or higher, you can further restrict processing to only those entries that satisfy the following additional criteria:

- A successful comparison of the list authority value for the target list with a user-specified value. You specify the list authority comparison requirements using AUTHCOMP and AUTHCOMPTYPE.
- A successful comparison of the entry key value with a user-specified value (KEYCOMP). If the comparison is not successful, the current list entry is not deleted and processing continues with the next entry to be considered.
- An successful comparison of the version number, which is enhanced to allow for an additional equal-or-less-than-equal comparison operation.

When the DELETE\_ENTRYLIST request completes, the number of deleted entries is returned in the answer area (ANSAREA).

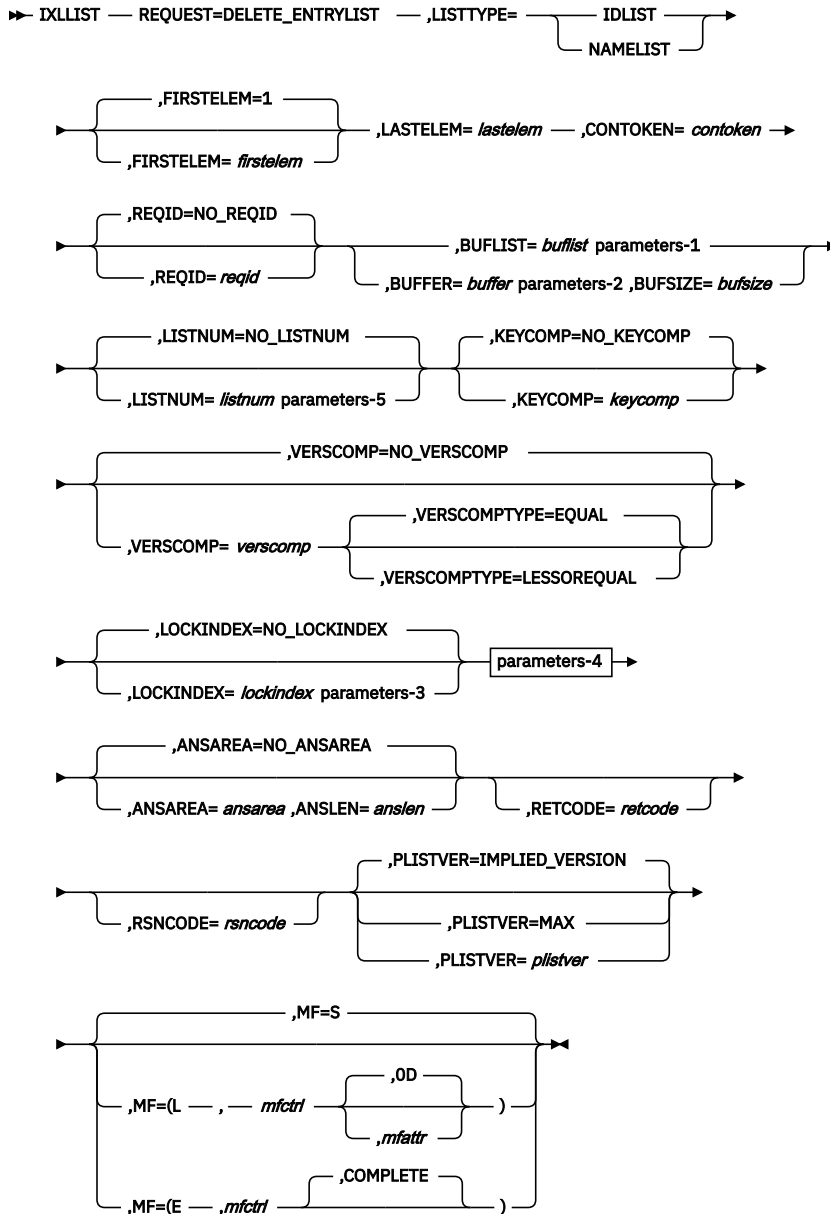
When an entry identified in the entry list does not exist, or does not exist with the indicated version number and/or list number, processing is halted and the index of the failing entry is returned in the answer area. All the entries on the list preceding this entry are deleted; all succeeding entries have not yet been deleted. To continue processing, update FIRSTELEM and reissue the macro.

If the request completes prematurely because it exceeds the model-dependent time-out criteria, the number of deleted entries thus far and the index of the next entry to be deleted is returned in the answer area. You can update FIRSTELEM to this returned index (LAAFAILINDEX) on a subsequent DELETE\_ENTRYLIST request to finish deleting the entries.

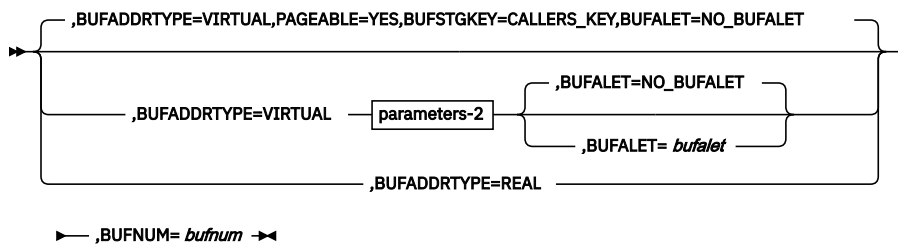
### Syntax Diagram

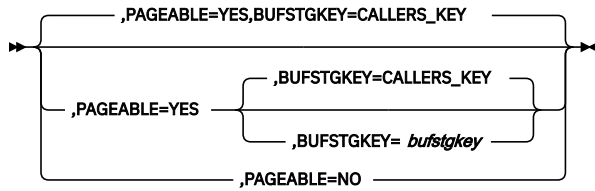
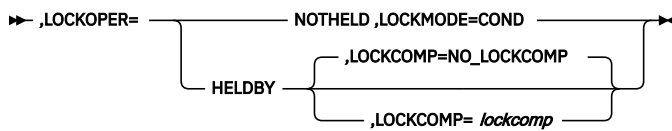
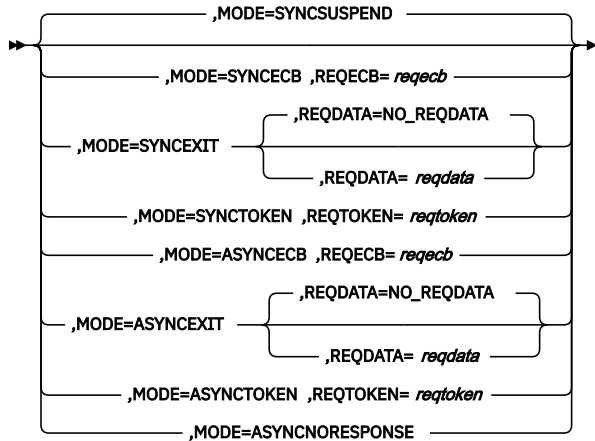
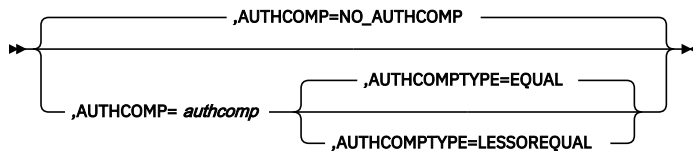
The syntax diagram for IXLLIST REQUEST=DELETE\_ENTRYLIST is as follows:

## main diagram



## parameters-1



**parameters-2****parameters-3****parameters-4****parameters-5****Note:**

1. If MODE=SYNCTOKEN or MODE=ASYNCTOKEN is specified, ANSAREA= *ansarea* and ANSLEN=*anslen* are required.
2. If MODE=ASYNCSUSPEND is specified, BUFFER, BUFLIST, and LOCKINDEX may not be specified.

## Parameter Descriptions

The parameter descriptions for REQUEST=DELETE\_ENTRYLIST are listed in alphabetical order. Default values are underlined:

**REQUEST=DELETE\_ENTRYLIST**

Use this input parameter to delete multiple entries that are identified in an entry list.

**,ANSAREA=NO\_ANSAREA****,ANSAREA=ansarea**

Use this output parameter to specify an answer area to contain information returned from the request. The format of the answer area is described by mapping macro IXLYLAA.

When the request completes successfully, the answer area contains the number of deleted entries (field LAADELCNT).

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an area (with a length of ANSLEN) to contain the information returned by the request.

**,ANSLEN=anslen**

Use this input parameter to specify the size of the storage area specified by ANSAREA.

Check the prologue of the IXLYLAA mapping macro for the minimum required size of the answer area, or use the length field (LAA\_LEN) in the LAA.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 2-byte field that contains the size, in bytes, of the answer area.

**,AUTHCOMP=NO\_AUTHCOMP****,AUTHCOMP=authcomp**

Use this input parameter to specify a value to be compared to the list authority value of the list specified by LISTNUM. You must supply the LISTNUM parameter.

If the comparison does not meet the condition specified by the AUTHCOMPTYPE parameter (EQUAL or LESSOREQUAL), the request fails.

**Note:** The AUTHCOMP parameter is valid only for list structures allocated in a coupling facility with CFLEVEL=1 or higher.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12 2 to ) of the 16-byte field that contains the list authority value.

**,AUTHCOMPTYPE=EQUAL****,AUTHCOMPTYPE=LESSOREQUAL**

Use this input parameter specify how the list audit comparison is to be performed.

**EQUAL**

The list authority for the list specified by LISTNUM must be equal to the value specified for AUTHCOMP.

**LESSOREQUAL**

The list authority for the list specified by LISTNUM must be less than or equal to the value specified for AUTHCOMP.

**Note:** The AUTHCOMPTYPE parameter is valid only for list structures allocated in a coupling facility with CFLEVEL=1 or higher.

**,BUFADDRTYPE=VIRTUAL****,BUFADDRTYPE=REAL**

Use this input parameter to specify whether the buffer addresses specified in the BUFLIST list are virtual storage or real storage addresses.

**VIRTUAL**

The buffer addresses are virtual storage addresses. The virtual storage can be pageable or nonpageable. See the PAGEABLE parameter for information about managing storage binds when specifying virtual storage addresses.

**REAL**

The buffer addresses are real storage addresses.

It is the caller's responsibility to manage the binds between the data buffer virtual storage and the real storage addresses provided. The caller must ensure that the data buffer virtual storage remains bound to the real storage addresses provided until the request completes.

**,BUFALET=NO\_BUFALET****,BUFALET=*bufalet***

Use this input parameter to specify an access list entry token (ALET) to be used in referencing all of the buffers specified by BUFLIST.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 4-byte field that contains the ALET.

**,BUFFER=*buffer***

Use this input parameter to specify a buffer area to hold either the entry identifiers or the entry names of the list entries to be deleted. Whether you place entry identifiers or entry names in the buffer depends on the LISTTYPE value you specify:

- If you specify LISTTYPE=IDLIST, the buffer should contain list entry identifiers.
- If you specify LISTTYPE=NAMELIST, the buffer should contain list entry names.

Only 31-bit addressable (below 2GB) virtual storage areas are supported for the BUFFER specification.

You must ensure that the storage area specified by BUFFER:

- Is a multiple of 4096 bytes.
- Is less than or equal to 65536 bytes.
- Starts on a 4096-byte boundary.

The buffer should be formatted according to the LISTTYPE value you specify:

- If LISTTYPE=IDLIST, format the buffer into units of 12 bytes each. Each 12-byte unit should contain the entry identifier of one entry to be deleted.
- For LISTTYPE=NAMELIST, format the buffer into units of 16-bytes each. Each 16-byte unit should contain the entry name of one entry to be deleted.

**Note:** You cannot code BUFFER with MODE=ASYNCRESPONSE.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an area (with a length of BUFSIZE) that contains the entry identifiers or entry names.

**,BUFLIST=*buflist***

Use this input parameter to specify a list of buffers to hold the entry identifiers or the entry names of the list entries to be deleted. Whether you place entry identifiers or entry names in the buffers depends on the LISTTYPE value you specify:

- If you specify LISTTYPE=IDLIST, the buffers should contain list entry identifiers.
- If you specify LISTTYPE=NAMELIST, the buffers should contain list entry names.

BUFLIST specifies a 128-byte storage area that consists of a list of 0 to 16 buffer addresses.

**The 128-byte storage area must:**

- Consist of 0 to 16 elements.
- Each element must consist of an 8-byte field in which:
  - The left (high-order) four bytes are reserved and
  - The right (low-order) four bytes contain the address of a buffer.

**The BUFLIST buffers must:**

- Reside in the same address space or data space.
- Be 4096 bytes in length.
- Start on a 4096-byte boundary.

**Note:** The buffers do not have to be contiguous in storage. (List services treats BUFLIST buffers as a single, contiguous buffer, even if the buffers are not contiguous.)

See the BUFNUM parameter description for specifying the number of buffers in the buffer list.

The buffers must be arranged according to the LISTTYPE value you specify:

- If LISTTYPE=IDLIST, each buffer should contain elements of 12 bytes each. Each 12-byte element should contain the entry identifier of one entry to be deleted.
- For LISTTYPE=NAMELIST, each buffer should contain elements of 16 bytes each. Each 16-byte element should contain the entry name of one entry to be deleted.

**Note:** You cannot code BUFLIST with MODE=ASYNCHRESPONSE.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 128-byte area that contains a list of buffer addresses.

**,BUFNUM=*bufnum***

Use this input parameter to specify the number of buffers in the BUFLIST list. Valid BUFNUM values are from 0 to 16. A value of zero indicates that no entries will be deleted.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 1-byte field that contains the number of buffers in the buffer list.

**,BUFSIZE=*bufsize***

Use this input parameter to specify the size of the BUFFER area. See the BUFFER parameter description for valid buffer sizes.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a fullword field that contains the size of the buffer (BUFFER) in bytes.

**,BUFSTGKEY=CALLERS\_KEY**

**,BUFSTGKEY=*bufstgkey***

Use this input parameter to specify a storage key that you define and use when referencing the buffers specified by BUFLIST or the buffer that is specified by BUFFER.

If you do not specify BUFSTGKEY, or if you specify BUFSTGKEY=CALLERS\_KEY, all references to one or more buffers are performed by using the caller's PSW key.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of an 8-bit field that contains the storage key in the format B'kkkkxxxx', where kkkk is the key and xxxx is ignored.

**,CONTOKEN=*contoken***

Use this input parameter to specify the connect token that was returned by the IXLCONN service. The connect token uniquely identifies your connection to the list structure, and must be specified on each IXLLIST invocation.

The connect token is available in the IXLCONN answer area that is mapped by IXLYCONA.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 16-byte field that contains the connect token.

**,FIRSTELEM=1**

**,FIRSTELEM=*firstelem***

Use this input parameter to specify an index into the entry list. The FIRSTELEM index identifies where in the list of entries to begin processing. A FIRSTELEM index of 5, for example, indicates that the fifth entry in the list is the beginning of the list of entries to be deleted.

The value must identify one of the entry identifiers or entry names contained in the BUFFER area or BUFLIST buffers.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 2-byte field that contains the entry index.

**,KEYCOMP=NO\_KEYCOMP**

**,KEYCOMP=*keycomp***

Use this input parameter to specify a value to be compared to the entry key of each list entry in the designated list.

If the list entry that is to be processed does not have an entry key value that is equal to the specified KEYCOMP value, the entry is not processed. Processing continues with the next entry.



**Note:** The KEYCOMP parameter is valid only for list structures allocated in a coupling facility with CFLEVEL=1 or higher.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the 16-byte field that contains the value to be compared the entry key of the designated list entry.

**,LASTELEM=*lastelem***

Use this input parameter to specify an index into the entry list. The LASTELEM index identifies the entry that designates the end of the list or the last entry that you want to process. A LASTELEM index of 10, for example, indicates that the entry associated with the tenth entry identifier or entry name in the entry list will be the final entry to be deleted.

The LASTELEM index must be greater than or equal to the FIRSTELEM index, and must specify one of the entry identifiers or entry names contained in the BUFFER area or BUFLIST buffers.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 2-byte field that contains the entry index.

**,LISTNUM=NO LISTNUM**

**,LISTNUM=*listnum***

Use this input parameter to specify the number of the list on which the entries to be deleted must reside. Specifying a list number restricts processing to entries that reside on the designated list.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 4-byte field that contains the number of the list.

**,LISTTYPE=IDLIST**

**,LISTTYPE=NAMELIST**

Use this input parameter to specify whether the input data in the BUFFER area or BUFLIST buffers consists of a list of entry identifiers or entry names.

**IDLIST**

The contents of the BUFFER area or BUFLIST buffers is a list of entry identifiers.

**NAMELIST**

The contents of the BUFFER area or BUFLIST buffers is a list of entry names.

**Note:** You cannot specify LISTTYPE=NAMELIST for structures that do not support named entries.

**,LOCKCOMP=NO LOCKCOMP**

**,LOCKCOMP=*lockcomp***

This parameter has slightly different meanings based on the value that is specified for the LOCKOPER parameter. Generally, this input parameter specifies a connection identifier to be verified as the owner of the lock specified by LOCKINDEX. This verification is a prerequisite to the successful completion of the request.

When LOCKCOMP is specified, the completion of the request is conditional on there being no contention for the lock. If contention exists, the request fails .

The connection identifier is available from the IXLCONN answer area, which is mapped which is by IXLYCONA, in field CONACONID.

The effect of LOCKCOMP is based on the LOCKOPER value specified. See the description under LOCKOPER to see how each request is affected by this parameter.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 1-byte field that contains the connection identifier.

**,LOCKINDEX=NO LOCKINDEX**

**,LOCKINDEX=*lockindex***

Use this input parameter to specify the index of the lock to be operated on as specified by LOCKOPER. The lock indexes begin with zero.

**Note:** You cannot code LOCKINDEX with MODE=ASYNCRESPONSE.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 4-byte field that contains the lock index.

**,LOCKMODE=COND**

Use this input parameter specify that the lock operation for the lock specified by LOCKINDEe performed conditionally; that is, only if there is no contention for the lock. If the specified lock is held by another user, the request is terminated with no change to the structure, and return and reason codes describing the termination are returned to the caller.

**,LOCKOPER=NOTHELD****,LOCKOPER=HELD**

Use this input parameter to specify the type of operation to be performed on the lock specified by LOCKINDEX.

**LOCKOPER=NOTHELD**

The request is performed only if the lock is not held by any connection. This option also ensures that the lock remains free for the duration of the request. If another connection holds the lock, your task is suspended until the lock is released, or your request fails depending on the LOCKMODE value you specify. See the LOCKMODE description for how to handle possible lock contention.

**LOCKOPER=HELD**

- When LOCKCOMP is not specified, the request is performed only if your connection holds the lock.
- When LOCKCOMP is specified, the request is performed only if the lock is held by the connection that is specified by LOCKCOMP.

**,MF=S****,MF=(L,mfctrl)****,MF=(L,mfctrl,mfattr)****,MF=(L,mfctrl,0D)****,MF=(M,mfctrl)****,MF=(M,mfctrl,COMPLETE)****,MF=(M,mfctrl,NOCHECK)****,MF=(E,mfctrl)****,MF=(E,mfctrl,COMPLETE)****,MF=(E,mfctrl,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

**,mfctrl**

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

**,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

**,COMPLETE****,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

**COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if `SMILE=var` were an optional parameter and the default is `SMILE=NO_SMILE` then it would not be documented. However, if the default was `SMILE=-)`, then it would be documented because a value would be the default.

**NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

**,MODE=SYNCSUSPEND****,MODE=SYNCECB****,MODE=SYNCEXIT****,MODE=SYNCTOKEN****,MODE=ASYNCECB****,MODE=ASYNCEXIT****,MODE=ASYNCTOKEN****,MODE=ASYNCSUSPEND**

Use this input parameter to specify:

- Whether the request is to be performed synchronously or asynchronously
- How you want to be notified of request completion

**MODE=SYNCSUSPEND**

The request is suspended until it can complete synchronously. To use this option, your program must be enabled for I/O and external interrupts.

**MODE=SYNCECB**

The request processes synchronously if possible. If the request processes asynchronously, the ECB specified by REQECB is posted when the request completes.

**MODE=SYNCEXIT**

The request processes synchronously if possible. If the request processes asynchronously, your complete exit is given control when the request completes.

**MODE=SYNCTOKEN**

The request processes synchronously if possible. If the request processes asynchronously, an asynchronous request token is returned in the area specified by REQTOKEN on invocation of the request. Use the returned request token on the IXLFComp macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=SYNCTOKEN is specified.

**MODE=ASYNCECB**

The request processes asynchronously. The ECB specified by REQECB is posted when the request completes.

**MODE=ASYNCEXIT**

The request processes asynchronously. Your complete exit is given control when the request completes.

**MODE=ASYNCTOKEN**

The request processes asynchronously. An asynchronous request token is returned in the area specified by REQTOKEN upon invocation of the request. Use the returned request token on the IXLFComp macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=ASYNCTOKEN is specified.

### **MODE=ASYNCRESPONSE**

The request processes asynchronously. No notification of request completion is provided. The answer area (ANSAREA) fields will not contain valid information when this mode is specified.

**Note:** You cannot code MODE=ASYNCRESPONSE with LOCKINDEX, BUFFER, or BUFLIST.

### **,PAGEABLE=YES**

### **,PAGEABLE=NO**

Use this input parameter to identify whether the storage areas specified by BUFFER or BUFLIST are in pageable or potentially pageable storage.

#### **YES**

Specify this option to indicate that the BUFFER or BUFLIST buffers reside in pageable virtual storage. XES performs the required page fixing to fix the buffers in real storage while the cache or list request transfers data to or from the coupling facility.

This includes storage obtained from pageable subpools, disabled reference (DREF) storage, and may include storage that has the potential to become pageable during the processing of a request. (An example is address space storage owned by any swappable address space, for which a PGSER FIX has been successfully processed, but for which the owning address space gets swapped during processing of a cache or list request.) This does not include implicitly non-pageable storage (for example, storage obtained from non-pageable subpools).

The system takes responsibility for managing binds to central storage for the duration of the cache or list request, regardless of what address space owns the storage or whether the storage-owning address space is swappable or nonswappable. The storage can be owned by any address space.

High shared virtual storage areas (above 2GB) may not be used.

#### **NO**

Specify this option to indicate that the BUFFER or BUFLIST buffers reside in non-pageable virtual storage. XES does not page fix the buffers in real storage.

This includes implicitly non-pageable storage areas (for example, storage obtained from non-pageable subpools), and may include storage that has the potential to become pageable during the processing of a request (An example is address space storage owned by any swappable address space, for which a PGSER FIX has been successfully processed, but for which the owning address space gets swapped-out during processing of a cache or list request.)

The system takes responsibility for managing binds to central storage for the duration of the cache or list request, if and only if the non-pageable storage is owned by either the requester's address space or the connector's address space. If the storage is owned by any other address space, then the invoker is responsible for ensuring that the virtual storage remains non-pageable for the duration of the request (including the case in which the storage is owned by a swappable address space that is swapped during processing of an IXLCACHE or IXLLIST request). Subject to this consideration, the storage can be owned by any address space. See [z/OS MVS Programming: Sysplex Services Guide](#).

### **,PLISTVER=IMPLIED\_VERSION**

### **,PLISTVER=MAX**

### **,PLISTVER=plistver**

Use this input parameter to specify the version of the macro. See [“Understanding IXLLIST Version Support” on page 965](#) for a description of the options available with PLISTVER.

### **,REQDATA=NO\_REQDATA**

### **,REQDATA=reqdata**

Use this input parameter with MODE=SYNCEXIT or MODE=ASYNCEXIT to pass any data you choose to the complete exit. The exit will get control only if the request is processed asynchronously.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the data to be passed to the complete exit.

**,REQECB=reqecb**

Use this output parameter with either MODE=SYNCECB or MODE=ASYNCECB to specify the address of an ECB, which is to be posted when the request completes if the request was processed asynchronously.

Before coding REQECB, you must ensure that:

- You initialize the ECB before you issue the request.
- The ECB resides in either common storage or the home address space where IXLCONN was issued.
- Any tasks that wait for the ECB to be posted reside in the home address space where IXLCONN was issued.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that contains the address of the ECB to be posted when the request completes. The ECB must be aligned on a fullword boundary.

**,REQID=NO\_REQID****,REQID=reqid**

Use this input parameter to specify a user-defined request identifier to be associated with the request. You can specify this request identifier on the IXPURGE macro to cancel a request that has not yet been processed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the user-defined request identifier.

**,REQTOKEN=reqtoken**

Use this output parameter with either MODE=SYNCTOKEN or MODE=ASYNCTOKEN to specify the address of a storage area to receive the request token that is returned when the request will be processed asynchronously. This token, which uniquely identifies the request, must be used as input to the IXLFCOMP macro, which you use to determine if the request has completed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field where the system will put the request token.

**,RETCODE=retcode**

Use this output parameter to specify a field to contain the return code. (The return code is also returned in register 15.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the return code when the request has completed.

**,RSNCODE=rsncode**

Use this output parameter to specify a field to contain the reason code returned, if applicable. (The reason code is also returned in register 0.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the reason code (if any) when the request has completed.

**,VERSCOMP=NO\_VERSCOMP****,VERSCOMP=verscomp**

Use this input parameter to specify a version number to be compared to the version number of each entry in the list of entries to be deleted. Only those entries with a version that meets the condition specified by the VERSCOMPTYPE parameter will be deleted.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-byte field that contains the version number.

**,VERSCOMPTYPE=EQUAL****,VERSCOMPTYPE=LESSOREQUAL**

Use this input parameter to specify how a list entry version comparison as specified by VERSCOMP is to be performed.

**Note:** The VERSCOMPTYPE parameter is valid only for list structures allocated in a coupling facility with CFLEVEL=1 or higher.

**VERSCOMPTYPE=EQUAL**

The version number for the list entry must be equal to the value specified for VERSCOMP.

**VERSCOMPTYPE=LESSOREQUAL**

The version number for the list entry must be less than or equal to the value specified for VERSCOMP.

## ABEND Codes

---

Abend X'026' (See *z/OS MVS System Codes* for more information on this abend.)

## Return and Reason Codes

---

When the system returns control to your program:

- GPR 15 (and RETCODE, if specified) contains a return code.
- If the return code is not zero, GPR 0 (and RSNCODE, if specified) contains a reason code.

**Note:** The return and reason codes will also be put into the answer area (mapped by IXLYLAA), if there is one, when the request completes.

The IXLYCON macro provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

**0**

IXLRETCODEOK

**4**

IXLRETCODEWARNING

**8**

IXLRETCODEPARMERROR

**C**

IXLRETCODEENVERROR

**10**

IXLRETCODECOMPERROR

The following table contains hexadecimal return and reason codes, the equate symbols associated with each reason code, and the meaning and suggested action for each return and reason code.

Table 58. Return and Reason Codes for IXLLIST REQUEST=DELETE_ENTRYLIST Macro		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0	None.	<p><b>Meaning:</b> If you specified a MODE value of ASYNCECB, ASYNCEXIT, ASYNCTOKEN, or ASYNCSNORESPONSE, the request has been successfully initiated. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed. For any other MODE value, the request has successfully completed.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=ASYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=ASYNCEXIT, the connection's complete exit will be given control when the request has completed.</li> <li>• If you specified MODE=ASYNCTOKEN, use REQTOKEN when calling IXLFCOMP to determine when the request has completed.</li> </ul>

Table 58. Return and Reason Codes for IXLLIST REQUEST=DELETE\_ENTRYLIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	xxxx0402	<p><b>Equate Symbol:</b> IXLRSNCODEASYNCH</p> <p><b>Meaning:</b> The request could not be processed synchronously. It will be processed asynchronously. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed.</p> <ul style="list-style-type: none"> <li>• If MODE=SYNCECB was specified, the ECB specified for REQECB will be posted when the request has finished.</li> <li>• If MODE=SYNCEXIT was specified, the connection's complete exit will be given control when the request has finished.</li> <li>• If MODE=SYNCTOKEN was specified, an asynchronous request token was returned in the area specified by REQTOKEN. This request token may be specified on the IXLFCOMP macro to determine when the request has finished.</li> </ul> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=SYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=SYNCEXIT, the connection's complete exit will be given control when the request has completed.</li> <li>• If you specified MODE=SYNCTOKEN, use REQTOKEN when calling IXLFCOMP to determine when the request has completed.</li> </ul>
4	xxxx0409	<p><b>Equate Symbol:</b> IXLRSNCODETIMEOUT</p> <p><b>Meaning:</b> The request completed prematurely because it exceeded the coupling facility model dependent time-out criteria. The following information has been returned in the answer area:</p> <ul style="list-style-type: none"> <li>• The number of entries that were deleted (field LAADELCNT)</li> <li>• The index of the first unprocessed entry identifier or entry name (field LAAFAILINDEX)</li> </ul> <p><b>Action:</b> Reissue the request specifying for FIRSTLEM the index of the first unprocessed entry. For more information about premature completion, see <a href="#">z/OS MVS Programming: Sysplex Services Guide</a>.</p>
4	xxxx0410	<p><b>Equate Symbol:</b> IXLRSNCODELOCKCOND</p> <p><b>Meaning:</b> For a LOCKOPER=HELDDBY request, or a LOCKMODE=COND request, or a request that specified LOCKCOMP, the request could not be completed successfully because the specified lock is not currently held as required. The connection identifier of the lock owner is returned in the answer area (LAACONID field).</p> <p><b>Action:</b> Retry the request, or obtain the lock as required, and retry the request. If you are unable to get the lock, check the ID in the LAACONID field and determine if some recovery is necessary.</p>

Table 58. Return and Reason Codes for IXLLIST REQUEST=DELETE\_ENTRYLIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	xxxx0412	<p><b>Equate Symbol:</b> IXLRSNCODELOCKHELDDBYSYS</p> <p><b>Meaning:</b> The lock is not held by any connection, but instead is held by the system. For a request that specified any of the following, the request could not be completed successfully because the specified lock is not currently held as required:</p> <ul style="list-style-type: none"> <li>• LOCKOPER=SET or LOCKOPER=NOTHELD with either of: <ul style="list-style-type: none"> <li>– LOCKMODE=COND</li> <li>– LOCKCOMP</li> </ul> </li> <li>• LOCKOPER=HELDDBY</li> </ul> <p><b>Action:</b> Retry the request, or obtain the lock as required, and retry the request.</p>
8	xxxx0801	<p><b>Equate Symbol:</b> IXLRSNCODEBADPARMLIST</p> <p><b>Meaning:</b> Program error. The parameter list for this request is not addressable. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Verify that:</p> <ul style="list-style-type: none"> <li>• The parameter list address is uncorrupted.</li> <li>• The parameter list is addressable in the caller's primary address space.</li> <li>• If you are issuing this macro while disabled, the parameter list resides in either page-fixed or DREF storage.</li> <li>• If you are issuing this macro in AR-mode and you specified the parameter list address using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If you are issuing this macro in AR-mode, SYSSTATE ASCENV=AR must be issued before issuing this macro.</li> </ul>
8	xxxx0804	<p><b>Equate Symbol:</b> IXLRSNCODEBADVERSIONNUM (or IXLRSNCODEBADVERSION#)</p> <p><b>Meaning:</b> The version number in the macro parameter list is not compatible with the level of XES currently being used. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that your program did not overlay the parameter list storage.</li> <li>• Verify that your program was assembled with the correct macro library for the release of MVS on which your program is running.</li> </ul>



Table 58. Return and Reason Codes for IXLLIST REQUEST=DELETE\_ENTRYLIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx080A	<p><b>Equate Symbol:</b> IXLRNCODEBADCONTOKEN</p> <p><b>Meaning:</b> Program error. The specified connect token (CONTOKEN) was not valid for one of the following reasons:</p> <ol style="list-style-type: none"> <li>1. The user with the connection identifier represented by the token has disconnected from the structure.</li> <li>2. The connector's task (the task that issued IXLCONN) ended.</li> <li>3. The specified token is not the token that was returned from IXLCONN.</li> <li>4. The request was issued from an address space other than the address space in which IXLCONN was issued.</li> <li>5. The connect token was invalidated during rebuild.</li> <li>6. The connect token was invalidated by XES.</li> </ol> <p><b>Note:</b> The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Take the action with the corresponding meaning.</p> <ol style="list-style-type: none"> <li>1. Discontinue use of the structure. Perform recovery and cleanup for the structure and check the protocol for the use of the structure to determine why you did not know of the disconnection.</li> <li>2. Discontinue use of the structure. Perform recovery and cleanup for the structure and check the protocol for the use of the structure to determine why you did not know that the task ended.</li> <li>3. The contoken is returned in the CONACONTOKEN field of the answer area specified on the IXLCONN request.</li> <li>4. Issue your request from the same address space the IXLCONN was issued in.</li> <li>5. Wait for the rebuild to complete, and try again.</li> <li>6. Discontinue use of the structure. Perform recovery and cleanup for the structure.</li> </ol>
8	xxxx0824	<p><b>Equate Symbol:</b> IXLRNCODEWRONGSTRTYPE</p> <p><b>Meaning:</b> Program error. The connection specified by CONTOKEN is not to a list structure.</p> <p><b>Action:</b> Specify the appropriate CONTOKEN value, and rerun your program. The CONTOKEN is returned in the answer area provided on the IXLCONN invocation. The type of structure built is passed on the TYPE parameter of the IXLCONN macro.</p>

Table 58. Return and Reason Codes for IXLLIST REQUEST=DELETE\_ENTRYLIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx082B	<p><b>Equate Symbol:</b> IXLRSNCODEBADIDINDEX</p> <p><b>Meaning:</b> Program error. The value specified for either FIRSTLEM or LASTLEM was not valid. Some entries may have been processed. The answer area (mapped by IXLYLAA) contains:</p> <ul style="list-style-type: none"> <li>• The count of entries successfully deleted, field LAADLCNT.</li> <li>• The index into the entry list of the entry identifier or entry or entry name that was not valid, field LAAFAILINDEX.</li> </ul> <p><b>Action:</b> Ensure that the FIRSTLEM and LASTLEM values correspond to entries in the list of entries to be processed.</p>
8	xxxx0833	<p><b>Equate Symbol:</b> IXLRSNCODEBADPGBLATTR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list is specified as being pageable (PAGEABLE=YES), but is not.</p> <p><b>Action:</b> Change the buffer area(s) to pageable storage, or specify PAGEABLE=NO. See the PAGEABLE parameter description for specification instructions.</p>
8	xxxx0834	<p><b>Equate Symbol:</b> IXLRSNCODEBADNONPGBLATTR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list, is specified as being nonpageable (PAGEABLE=NO), but is either pageable or not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The type of storage in the buffer area corresponds to the value you specified on the PAGEABLE parameter. See the PAGEABLE parameter description for specification instructions.</li> <li>• The correct buffer address was used.</li> <li>• The buffer area was not previously freed.</li> <li>• If you are calling IXLLIST while disabled, the buffers must reside in either page-fixed or DREF storage.</li> <li>• The buffer areas were allocated in a storage key that matches the key specified by the BUFSTGKEY parameter, or, if BUFSTGKEY is omitted, the caller's PSW key.</li> <li>• If BUFLIST was specified and your program is running in AR-mode: <ul style="list-style-type: none"> <li>– If the BUFFER or BUFLIST parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>– The BUFALET specification is correct.</li> <li>– You specified SYSSTATE ASCENV=AR before issuing the IXLLIST macro.</li> </ul> </li> </ul>

Table 58. Return and Reason Codes for IXLLIST REQUEST=DELETE\_ENTRYLIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0835	<p><b>Equate Symbol:</b> IXLRNCODEBADDATAADDR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list, is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The correct buffer address was used.</li> <li>• The buffer area was not previously freed.</li> <li>• The buffer area was allocated in a storage key that matches the key specified using the BUFSTGKEY parameter, or, if BUFSTGKEY is omitted, the caller's PSW key.</li> <li>• If BUFLIST was specified and your program is running in AR mode: <ul style="list-style-type: none"> <li>– The BUFALET specification is correct.</li> <li>– You specified SYSSTATE ASCENV=AR before issuing the macro.</li> </ul> </li> </ul>
8	xxxx0836	<p><b>Equate Symbol:</b> IXLRNCODEBADREALADDR</p> <p><b>Meaning:</b> Program error. Real storage addresses were provided in the BUFLIST list, but one of the buffers is not addressable in central storage.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that BUFADDRTYPE was specified as you intended.</li> <li>• Ensure that the buffer addresses specified by BUFLIST are valid.</li> </ul>
8	xxxx0838	<p><b>Equate Symbol:</b> IXLRNCODEBADANSAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by ANSAREA is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The answer area address specified by ANSAREA is valid.</li> <li>• If the caller is running in AR-mode and the ANSAREA parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• The address specified for ANSAREA is addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are calling IXLLIST while disabled, ANSAREA must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLLIST in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLIST macro.</li> </ul>

Table 58. Return and Reason Codes for IXLLIST REQUEST=DELETE\_ENTRYLIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0839	<p><b>Equate Symbol:</b> IXLRSNCODEBADREQTOKENAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by REQTOKEN is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The request token area specified by REQTOKEN is valid.</li> <li>• If the caller is running in AR-mode and the REQTOKEN parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If you are calling IXLLIST while disabled, REQTOKEN must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLLIST in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLIST macro.</li> </ul>
8	xxxx083D	<p><b>Equate Symbol:</b> IXLRSNCODEBADANSLEN</p> <p><b>Meaning:</b> Program error. The length of the answer area, as specified by ANSLEN, is not sufficient to contain answer area information. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Increase the size of the answer area, and rerun your program. The minimum size of the ANSAREA is in the LAA_LEN field of the IXLYLAA mapping macro.</p>
8	xxxx0843	<p><b>Equate Symbol:</b> IXLRSNCODEBADENTRYID</p> <p><b>Meaning:</b> Program error. The entry corresponding to either an entry identifier (LISTTYPE=IDLIST) or entry name (LISTTYPE=NAMELIST) in BUFFER or the BUFLIST buffers does not exist. The index to the failing entry identifier or name was returned in the answer area (field LAAFAILINDEX). The count of entries deleted is also returned in the answer area (field LAADELCNT).</p> <p><b>Action:</b> Remove the failing entry from the list, or reissue the request. Update FIRSTELEM to point after the failing entry (FIRSTELEM = LAAFAILINDEX + 1) to the next entry on the list to be processed.</p>
8	xxxx0845	<p><b>Equate Symbol:</b> IXLRSNCODENONAMES</p> <p><b>Meaning:</b> Program error. A list entry was designated by entry name, but the structure does not support entry names.</p> <p><b>Action:</b> Ensure that you are connected to the intended structure. Ensure that the correct specification was made for REFOPTION on the IXLCONN macro. You may want to issue IXLMG to get more information about the structure.</p>

Table 58. Return and Reason Codes for IXLLIST REQUEST=DELETE\_ENTRYLIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0846	<p><b>Equate Symbol:</b> IXLRSNCODEBADLOCKINDEX</p> <p><b>Meaning:</b> Program error. The specified LOCKINDEX exceeds the size of the lock table for the structure.</p> <p><b>Action:</b> Correct LOCKINDEX to specify an index that is contained within the lock table. The maximum value for the LOCKINDEX is one less than the value specified by the LOCKENTRIES parameter on the IXLCONN macro.</p>
8	xxxx0847	<p><b>Equate Symbol:</b> IXLRSNCODEBADLISTNUMBER</p> <p><b>Meaning:</b> Program error. The specified LISTNUM value exceeds the number of lists for the structure.</p> <p><b>Action:</b> Correct LISTNUM to specify a list number that exists in the structure. The number of lists allocated for a structure is determined on the LISTHEADERS parameter of the IXLCONN macro. The list numbers go from 0 to n-1, where n is the number of lists specified.</p>
8	xxxx084B	<p><b>Equate Symbol:</b> IXLRSNCODENOLOCKS</p> <p><b>Meaning:</b> Program error. A locking operation was requested, but the structure does not contain a lock table.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• You are connected to the intended structure.</li> <li>• You intended to perform a locking operation.</li> </ul> <p>The use of locks is determined by the LOCKENTRIES keyword on the IXLCONN macro.</p>
8	xxxx0851	<p><b>Equate Symbol:</b> IXLRSNCODENOSUSPENDISABLE</p> <p><b>Meaning:</b> Program error. The request failed because MODE=SYNCSUSPEND was specified, but the caller is disabled and cannot be suspended.</p> <p><b>Action:</b> Either specify another MODE value, or become enabled (release the CPU lock), then reissue the request.</p>
8	xxxx0854	<p><b>Equate Symbol:</b> IXLRSNCODEBADLOCKCOMP</p> <p><b>Meaning:</b> Program error. The connection identifier specified for LOCKCOMP is not valid.</p> <p><b>Action:</b> The connection identifier is returned in the answer area (mapped by IXLYCONA) when the IXLCONN macro was issued.</p>

Table 58. Return and Reason Codes for IXLLIST REQUEST=DELETE\_ENTRYLIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0859	<p><b>Equate Symbol:</b> IXLRNCODEBADLISTAUTH</p> <p><b>Meaning:</b> The list authority specified for AUTHCOMP failed the comparison specified by AUTHCOMPTYPE for the list authority of the specified list. The current list authority (field LAALISTAUTH) and description (field LAALISTDESC) are returned in the answer area.</p> <p><b>Action:</b> None required; however you might want to take some action depending on your application. The list authority is set to binary zeros when the structure is allocated. When IXLLIST is issued with the NEWAUTH keyword, the list authority is changed if the correct comparison list authority value is specified. Check the answer area to determine the list authority value for the list specified. Verify that the correct list number was specified.</p>
8	xxxx0865	<p><b>Equate Symbol:</b> IXLRNCODEBADBUFSPEC</p> <p><b>Meaning:</b> Program error. There is an error in the buffer specification</p> <p><b>Action:</b> Check the following:</p> <ul style="list-style-type: none"> <li>• If BUFLIST was specified, check the requirements for BUFLIST and BUFNUM.</li> <li>• If BUFFER was specified, check the requirements for BUFFER and BUFSIZE.</li> <li>• Buffer pointer(s) in BUFLIST.</li> <li>• Buffer boundaries.</li> </ul>
8	xxxx0866	<p><b>Equate Symbol:</b> IXLRNCODEBADBUFKEY</p> <p><b>Meaning:</b> Program error. The buffer storage key specified by BUFSTGKEY is incorrect or BUFSTGKEY was not specified and the PSW key does not match the key of the buffers. The data cannot be fetched from the specified buffer area.</p> <p><b>Action:</b> Check the following:</p> <ul style="list-style-type: none"> <li>• Determine if the key of the storage being used for the buffers is different from the PSW key.</li> <li>• If BUFSTGKEY was specified, verify that the key was put in the correct bits (see the explanation of this parameter for more information).</li> <li>• If you are calling IXLLIST in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLIST macro.</li> </ul>
8	xxxx0867	<p><b>Equate Symbol:</b> IXLRNCODEBADBUFLIST</p> <p><b>Meaning:</b> Program error. The 128-byte storage area specified by BUFLIST is not addressable.</p> <p><b>Action:</b> Ensure that the address specified by BUFLIST is valid.</p>

Table 58. Return and Reason Codes for IXLLIST REQUEST=DELETE\_ENTRYLIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx08AD	<b>Equate Symbol:</b> IXLRNCODEBADHIGHSHAREDVIRT <b>Meaning:</b> Program error. The request specified a high shared virtual storage area (above 2GB). <b>Action:</b> None required.
C	xxxx0C06	<b>Equate Symbol:</b> IXLRNCODENOCOONN <b>Meaning:</b> Environmental error. This system does not have connectivity to the coupling facility that contains the list structure. Possible reasons for this are: <ul style="list-style-type: none"> <li>• The operator issued VARY PATH,OFFLINE.</li> <li>• The operator issued CONFIG CHP,OFFLINE.</li> <li>• Hardware errors to the coupling facility.</li> <li>• Facility or path failure to the coupling facility.</li> </ul> <b>Action:</b> Begin rebuilding the structure on a different coupling facility, or disconnect from the structure.
C	xxxx0C13	<b>Equate Symbol:</b> IXLRNCODEREQPURGED <b>Meaning:</b> Environmental error. The request was purged prior to completion of the request. Possible reasons include: <ul style="list-style-type: none"> <li>• The connector failed.</li> <li>• The connector disconnected.</li> <li>• The requestor failed.</li> <li>• The request was purged by IXLPURGE.</li> <li>• Requests were purged when the connector provided an IXLEERSP response for the Rebuild Stop or Rebuild Cleanup event.</li> <li>• The secondary address space was no longer valid.</li> </ul> <b>Action:</b> None if this is expected. Otherwise, determine why the connector failed.
C	xxxx0C14	<b>Equate Symbol:</b> IXLRNCODESTATUSUNKNOWN <b>Meaning:</b> Environmental error. The request has completed, but the final disposition of the request cannot be determined. The answer area (ANSAREA) fields do not contain valid information. <b>Action:</b> <ul style="list-style-type: none"> <li>• Verify the validity of your data by comparing the expected results with what is in the coupling facility.</li> <li>• Consider running rebuild processing to move the structure to attempt to protect it from further errors of this kind.</li> </ul>

Table 58. Return and Reason Codes for IXLLIST REQUEST=DELETE\_ENTRYLIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C25	<b>Equate Symbol:</b> IXLRSNCODESTRFAILURE <b>Meaning:</b> Environmental error. The list structure failed prior to completion of the request. <b>Action:</b> Either rebuild or disconnect from the structure.
C	xxxx0CA0	<b>Equate Symbol:</b> IXLRSNCODEQUIESCEDSUSPENDFAIL <b>Meaning:</b> Environmental error. The request is failed because the structure is quiesced for a system-managed process and SUSPEND=FAIL is specified on the IXLCONN. <b>Action:</b> None, if this is expected.
C	xxxxFFFF	<b>Equate Symbol:</b> IXLRSNCODENOTAVAILABLE <b>Meaning:</b> Environmental error. XES functions are not available. This could occur because the coupling facility hardware necessary to provide XES functions is not present. <b>Action:</b> Re-IPL the system, or follow your particular failure management protocol.
10	xxxx10xx	<b>Meaning:</b> System error. XES processing failure. The state of the involved structure and the disposition of the request are unpredictable. The answer area (ANSAREA) fields do not contain valid information. <b>Action:</b> Contact the IBM support center.



---

## Chapter 59. IXLLIST REQUEST=DELETE\_MULT

### Description

---

A DELETE\_MULT request allows you to delete multiple entries from the list structure. Once removed, the storage resource associated with the entries can be reused. You can restrict processing to only those entries that satisfy either one or both of the following criteria:

- Reside on a certain list (which you specify using LISTNUM).
- Contain data that satisfies a version number comparison (which you specify using VERSCOMP).

**Important:** For list structures allocated in a CFLEVEL=0 coupling facility, if you do not specify either LISTNUM or VERSCOMP, every entry in the structure is deleted.

With a coupling facility of CFLEVEL=1 or higher, you can restrict processing to only those entries that satisfy one or more of the following additional criteria:

- A successful comparison of the list authority value for the target list with a user-specified value. You specify the authority comparison requirements using AUTHCOMP and AUTHCOMPTYPE. You must also specify the list number using LISTNUM.
- A successful comparison of the entry key value with a user-specified value (KEYCOMP). If the comparison is not successful, the current list entry is not deleted and processing continues with the next entry to be considered.
- A version number comparison that is enhanced to allow for an additional equal-or-less-than-equal comparison operation.

If the request completes prematurely because it exceeds the coupling facility model-dependent time-out criteria, a restart token (RESTOKEN) or an extended restart token (EXTRESTOKEN) is returned to the answer area (ANSAREA). The token can be specified on another DELETE\_MULT request to resume processing with the next entry to be deleted. Resumed requests are processed identically whether using the RESTOKEN or EXTRESTOKEN to specify the starting location.

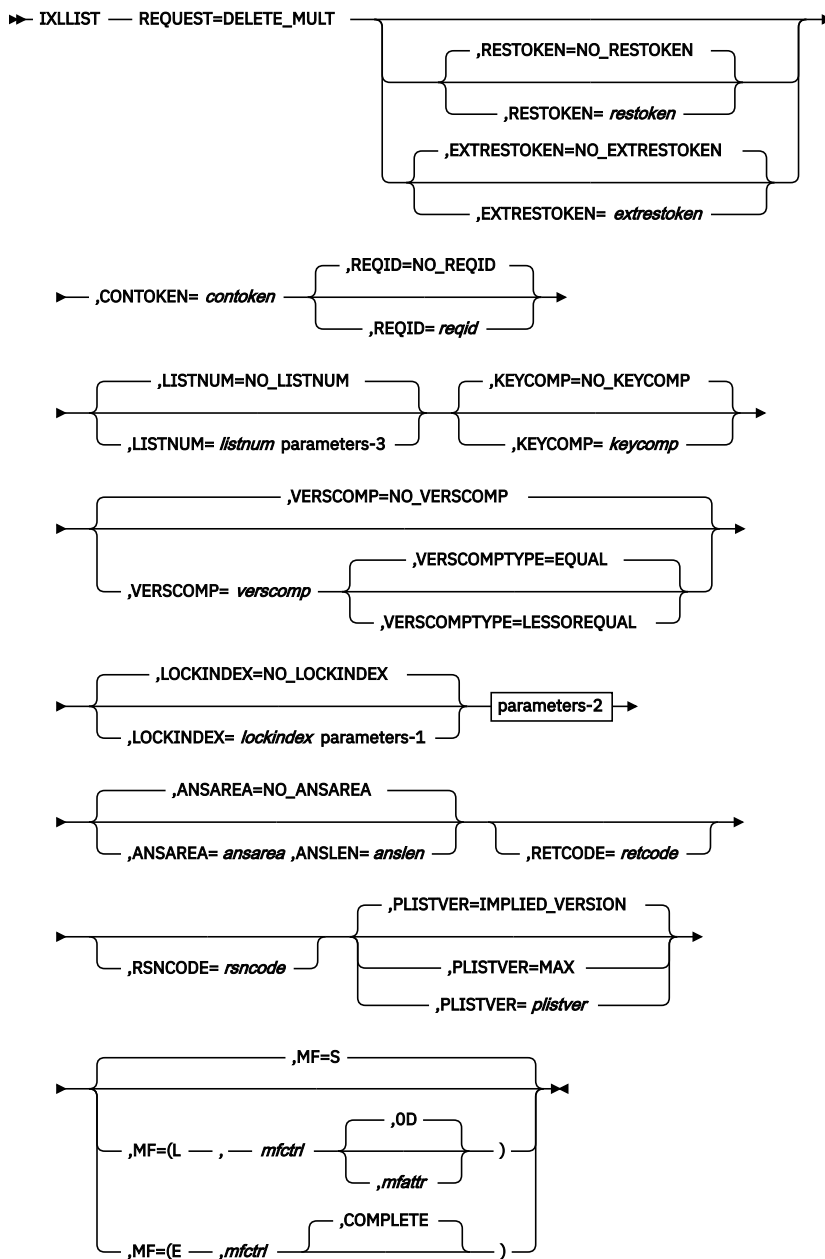
When the DELETE\_MULT request completes, the number of entries that were deleted is returned to the answer area.

### Syntax Diagram

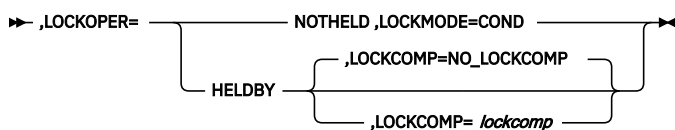
---

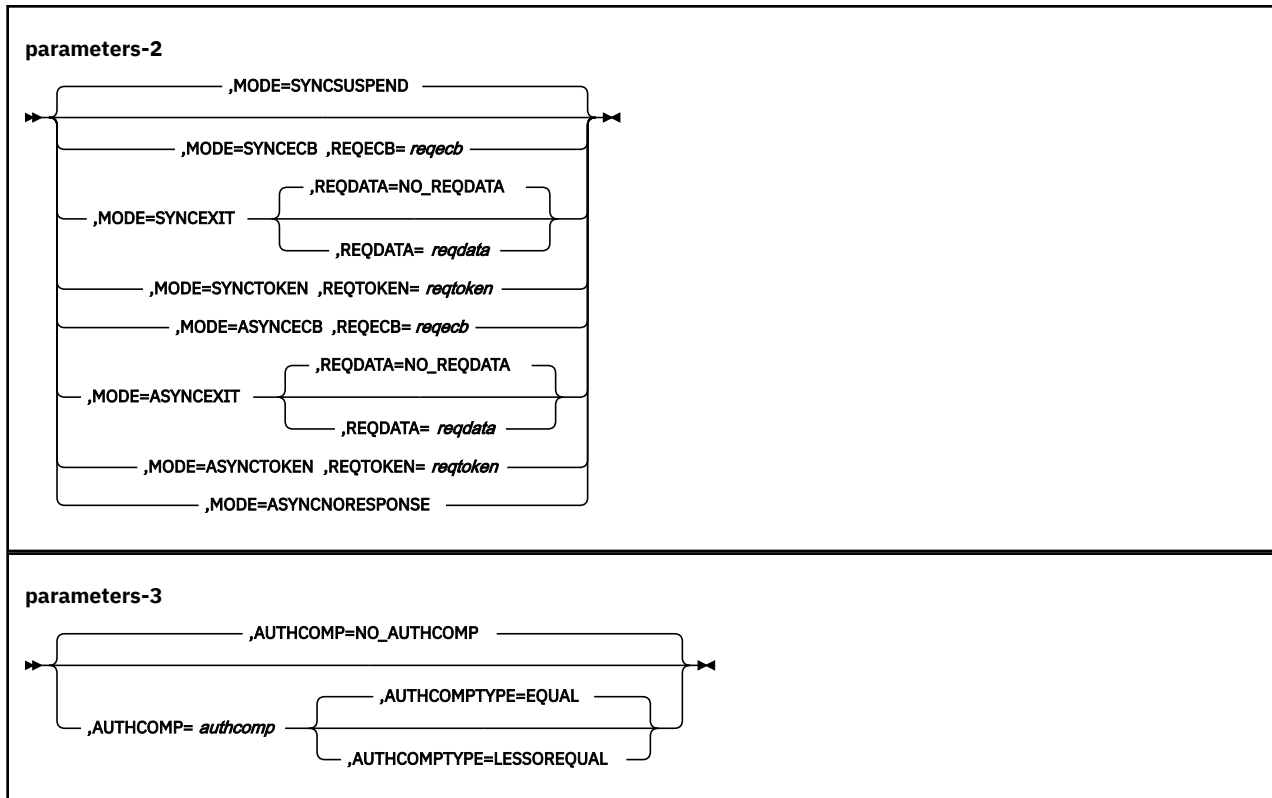
The syntax diagram for IXLLIST REQUEST=DELETE\_MULT is as follows:

## main diagram



## parameters-1



**Note:**

1. If `MODE=SYNCTOKEN` or `MODE=ASYNCTOKEN` is specified, `ANSAREA= ansarea` and `ANSLEN=anslen` are required.
2. If `MODE=ASYNCSUSPEND` is specified, `LOCKINDEX` may not be specified.

## Parameter Descriptions

The parameter descriptions for `REQUEST=DELETE_MULT` are listed in alphabetical order. Default values are underlined:

**`REQUEST=DELETE_MULT`**

Use this input parameter to delete multiple entries from the list structure.

**`,ANSAREA=NO_ANSAREA`****`,ANSAREA=ansarea`**

Use this output parameter to specify an answer area to contain information returned from the request. The format of the answer area is described by mapping macro `IXLYLAA`.

When the request completes, the answer area contains the number of deleted entries (field `LAADELCNT`).

If the request completes prematurely, a restart token (field `LAARESTOKEN`) or extended restart token (field `LAAEXTRESTOKEN`) is returned that may be specified on a subsequent `DELETE_MULT` request. (See the `RESTOKEN` and `EXTRESTOKEN` parameter descriptions.)

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an area (with a length of `ANSLEN`) to contain the information related to the request.

**`,ANSLEN=anslen`**

Use this input parameter to specify the size of the storage area specified by `ANSAREA`.

Check the prologue of the `IXLYLAA` mapping macro for the minimum required size of the answer area, or use the length field (`LAA_LEN`) in the `LAA`.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 2-byte field that contains the size, in bytes, of the answer area.

**,AUTHCOMP=NO\_AUTHCOMP**

**,AUTHCOMP=authcomp**

Use this input parameter to specify a value to be compared to the list authority value of the list specified by LISTNUM. You must supply the LISTNUM parameter.

If the comparison does not meet the condition specified by the AUTHCOMPTYPE parameter (EQUAL or LESSOREQUAL), the request fails.

**Note:** The AUTHCOMP parameter is valid only for list structures allocated in a coupling facility with CFLEVEL=1 or higher.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12 2 to ) of the 16-byte field that contains the list authority value.

**,AUTHCOMPTYPE=EQUAL**

**,AUTHCOMPTYPE=LESSOREQUAL**

Use this input parameter specify how the list audit comparison is to be performed.

**EQUAL**

The list authority for the list specified by LISTNUM must be equal to the value specified for AUTHCOMP.

**LESSOREQUAL**

The list authority for the list specified by LISTNUM must be less than or equal to the value specified for AUTHCOMP.

**Note:** The AUTHCOMPTYPE parameter is valid only for list structures allocated in a coupling facility with CFLEVEL=1 or higher.

**,CONTOKEN=contoken**

Use this input parameter to specify the connect token that was returned by the IXLCONN service. The connect token uniquely identifies your connection to the list structure, and must be specified on each IXLLIST invocation.

The connect token is available in the IXLCONN answer area that is mapped by IXLYCONA.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 16-byte field that contains the connect token.

**,EXTRESTOKEN=NO\_EXTRESTOKEN**

**,EXTRESTOKEN=extrestoken**

Use this input parameter to specify an extended restart token that can be used to resume processing of a DELETE\_MULT request that completed prematurely. The extended restart token is returned in the answer area (field LAAEXTRESTOKEN), and should be specified on the next DELETE\_MULT request to resume processing.

If the request does not complete prematurely, the extended restart token will not be provided.

**Note:**

1. Specifying an extended restart token of all zeros causes list services to treat all of the entries as unprocessed.
2. Do not specify an extended restart token other than the one returned in the answer area or one set to all zeros, because results will be unpredictable.

Requestors that specify IXLCONN ALLOWAUTO=YES must use the extended restart token. Requestors that specify or default to IXLCONN ALLOWAUTO=NO must use the standard restart token (RESTOKEN).

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-byte field that contains the extended restart token.

**,KEYCOMP=NO\_KEYCOMP****,KEYCOMP=keycomp**

Use this input parameter to specify a value to be compared to the entry key of each list entry in the designated list.

If the list entry that is to be processed does not have an entry key value that is equal to the specified KEYCOMP value, the entry is not processed. Processing continues with the next entry.

**Note:** The KEYCOMP parameter is valid only for list structures allocated in a coupling facility with CFLEVEL=1 or higher.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the 16-byte field that contains the value to be compared the entry key of the designated list entry.

**,LISTNUM=NO\_LISTNUM****,LISTNUM=listnum**

Use this input parameter to specify the number of the list on which the entries to be deleted must reside. Specifying a list number restricts delete processing to entries that reside on the designated list.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 4-byte field that contains the number of the list.

**,LOCKCOMP=NO\_LOCKCOMP****,LOCKCOMP=lockcomp**

This parameter has slightly different meanings based on the value that is specified for the LOCKOPER parameter. Generally, this input parameter specifies a connection identifier to be verified as the owner of the lock specified by LOCKINDEX. This verification is a prerequisite to the successful completion of the request.

When LOCKCOMP is specified, the completion of the request is conditional on there being no contention for the lock. If contention exists, the request fails .

The connection identifier is available from the IXLCONN answer area, which is mapped which is by IXLYCONA, in field CONACONID.

The effect of LOCKCOMP is based on the LOCKOPER value specified. See the description under LOCKOPER to see how each request is affected by this parameter.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 1-byte field that contains the connection identifier.

**,LOCKINDEX=NO\_LOCKINDEX****,LOCKINDEX=lockindex**

Use this input parameter to specify the index of the lock to be operated on as specified by LOCKOPER. The lock indexes begin with zero.

**Note:** You cannot code LOCKINDEX with MODE=ASYNCRESPONSE.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 4-byte field that contains the lock index.

**,LOCKMODE=COND**

Use this input parameter specify that the lock operation for the lock specified by LOCKINDEX performed conditionally; that is, only if there is no contention for the lock. If the specified lock is held by another user, the request is terminated with no change to the structure, and return and reason codes describing the termination are returned to the caller.

**,LOCKOPER=NOTHELD****,LOCKOPER=HELD BY**

Use this input parameter to specify the type of operation to be performed on the lock specified by LOCKINDEX.

**LOCKOPER=NOTHELD**

The request is performed only if the lock is not held by any connection. This option also ensures that the lock remains free for the duration of the request. If another connection holds the

lock, your task is suspended until the lock is released, or your request fails depending on the LOCKMODE value you specify. See the LOCKMODE description for how to handle possible lock contention.

### LOCKOPER=HELD BY

- When LOCKCOMP is not specified, the request is performed only if your connection holds the lock.
- When LOCKCOMP is specified, the request is performed only if the lock is held by the connection that is specified by LOCKCOMP.

**,MF=S**

**,MF=(L,mfctrl)**

**,MF=(L,mfctrl,mfattr)**

**,MF=(L,mfctrl,0D)**

**,MF=(M,mfctrl)**

**,MF=(M,mfctrl,COMPLETE)**

**,MF=(M,mfctrl,NOCHECK)**

**,MF=(E,mfctrl)**

**,MF=(E,mfctrl,COMPLETE)**

**,MF=(E,mfctrl,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

**,mfctrl**

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

**,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

**,COMPLETE**

**,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

### COMPLETE

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if SMILE=*var* were an optional parameter and the default is SMILE=NO\_SMILE then it would not be documented. However, if the default was SMILE=-), then it would be documented because a value would be the default.

**NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

**,MODE=SYNCSUSPEND**

**,MODE=SYNCECB**

**,MODE=SYNCEXIT**

**,MODE=SYNCTOKEN**

**,MODE=ASYNCECB**

**,MODE=ASYNCEXIT**

**,MODE=ASYNCTOKEN**

**,MODE=ASYNCSNORESPONSE**

Use this input parameter to specify:

- Whether the request is to be performed synchronously or asynchronously
- How you want to be notified of request completion

**MODE=SYNCSUSPEND**

The request is suspended until it can complete synchronously. To use this option, your program must be enabled for I/O and external interrupts.

**MODE=SYNCECB**

The request processes synchronously if possible. If the request processes asynchronously, the ECB specified by REQECB is posted when the request completes.

**MODE=SYNCEXIT**

The request processes synchronously if possible. If the request processes asynchronously, your complete exit is given control when the request completes.

**MODE=SYNCTOKEN**

The request processes synchronously if possible. If the request processes asynchronously, an asynchronous request token is returned in the area specified by REQTOKEN on invocation of the request. Use the returned request token on the IXLFCOMP macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=SYNCTOKEN is specified.

**MODE=ASYNCECB**

The request processes asynchronously. The ECB specified by REQECB is posted when the request completes.

**MODE=ASYNCEXIT**

The request processes asynchronously. Your complete exit is given control when the request completes.

**MODE=ASYNCTOKEN**

The request processes asynchronously. An asynchronous request token is returned in the area specified by REQTOKEN upon invocation of the request. Use the returned request token on the IXLFCOMP macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=ASYNCTOKEN is specified.

**MODE=ASYNCSNORESPONSE**

The request processes asynchronously. No notification of request completion is provided. The answer area (ANSAREA) fields will not contain valid information when this mode is specified.

**Note:** You cannot code MODE=ASYNCSNORESPONSE with LOCKINDEX, BUFFER, or BUFLIST.

**,PLISTVER=IMPLIED\_VERSION**

**,PLISTVER=MAX**

**,PLISTVER=plistver**

Use this input parameter to specify the version of the macro. See [“Understanding IXLLIST Version Support”](#) on page 965 for a description of the options available with PLISTVER.

**,REQDATA=NO REQDATA****,REQDATA=reqdata**

Use this input parameter with MODE=SYNCEXIT or MODE=ASYNCEXIT to pass any data you choose to the complete exit. The exit will get control only if the request is processed asynchronously.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the data to be passed to the complete exit.

**,REQECB=reqecb**

Use this output parameter with either MODE=SYNCECB or MODE=ASYNCECB to specify the address of an ECB, which is to be posted when the request completes if the request was processed asynchronously.

Before coding REQECB, you must ensure that:

- You initialize the ECB before you issue the request.
- The ECB resides in either common storage or the home address space where IXLCONN was issued.
- Any tasks that wait for the ECB to be posted reside in the home address space where IXLCONN was issued.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that contains the address of the ECB to be posted when the request completes. The ECB must be aligned on a fullword boundary.

**,REQID=NO REQID****,REQID=reqid**

Use this input parameter to specify a user-defined request identifier to be associated with the request. You can specify this request identifier on the IXPURGE macro to cancel a request that has not yet been processed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the user-defined request identifier.

**,REQTOKEN=reqtoken**

Use this output parameter with either MODE=SYNCTOKEN or MODE=ASYNCTOKEN to specify the address of a storage area to receive the request token that is returned when the request will be processed asynchronously. This token, which uniquely identifies the request, must be used as input to the IXLFComp macro, which you use to determine if the request has completed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field where the system will put the request token.

**,RESTOKEN=NO RESTOKEN****,RESTOKEN=restoken**

Use this input parameter to specify a restart token that can be used to resume processing of a DELETE\_MULT request that completed prematurely because it exceeded the coupling facility model-dependent time-out criteria. The restart token is returned in the answer area (field LAARESTOKEN), and should be specified on the next DELETE\_MULT request to resume processing with the next entry to be deleted.

If the request does not exceed the model-dependent time-out criteria, the returned token will not be provided.

**Note:**

1. Specifying an extended restart token of all zeros causes list services to treat all of the entries as unprocessed.
2. Do not specify an extended restart token other than the one returned in the answer area or one set to all zeros, because results will be unpredictable.

Requestors that specify or default to IXLCONN ALLOWAUTO=NO must use the standard restart token. Requestors that specify IXLCONN ALLOWAUTO=YES must use the extended restart token (EXTRESTOKEN).



**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-byte field that contains the restart token.

**,RETCODE=retcode**

Use this output parameter to specify a field to contain the return code. (The return code is also returned in register 15.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the return code when the request has completed.

**,RSNCODE=rsncode**

Use this output parameter to specify a field to contain the reason code returned, if applicable. (The reason code is also returned in register 0.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the reason code (if any) when the request has completed.

**,VERSCOMP=NO\_VERSCOMP**

**,VERSCOMP=verscomp**

Use this input parameter to specify a version number to be compared to the version number of each entry to be deleted. Only those entries with a version that meets the condition specified by the VERSCOMPTYPE parameter will be deleted.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-byte field that contains the version number.

**,VERSCOMPTYPE=EQUAL**

**,VERSCOMPTYPE=LESSOREQUAL**

Use this input parameter to specify how a list entry version comparison as specified by VERSCOMP is to be performed.

**Note:** The VERSCOMPTYPE parameter is valid only for list structures allocated in a coupling facility with CFLEVEL=1 or higher.

**VERSCOMPTYPE=EQUAL**

The version number for the list entry must be equal to the value specified for VERSCOMP.

**VERSCOMPTYPE=LESSOREQUAL**

The version number for the list entry must be less than or equal to the value specified for VERSCOMP.

## ABEND Codes

---

Abend X'026' (See *z/OS MVS System Codes* for more information on this abend.)

## Return and Reason Codes

---

When the system returns control to your program:

- GPR 15 (and RETCODE, if specified) contains a return code.
- If the return code is not zero, GPR 0 (and RSNCODE, if specified) contains a reason code.

**Note:** The return and reason codes will also be put into the answer area (mapped by IXLYLAA), if there is one, when the request completes.

The IXLYCON macro provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

**0**

IXLRETCODEOK

**4**

IXLRETCODEWARNING

8

IXLRETCODEPARMERROR

C

IXLRETCODEENVERROR

10

IXLRETCODECOMPERROR

The following table contains hexadecimal return and reason codes, the equate symbols associated with each reason code, and the meaning and suggested action for each return and reason code.

Table 59. Return and Reason Codes for IXLLIST REQUEST=DELETE_MULT Macro		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0	None.	<p><b>Meaning:</b> If you specified a MODE value of ASYNCECB, ASYNCEXIT, ASYNCTOKEN, or ASYNCSNORESPONSE, the request has been successfully initiated. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed. For any other MODE value, the request has successfully completed.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=ASYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=ASYNCEXIT, the connection's complete exit will be given control when the request has completed.</li> <li>• If you specified MODE=ASYNCTOKEN, use REQTOKEN when calling IXLFCOMP to determine when the request has completed.</li> </ul>
4	xxxx0402	<p><b>Equate Symbol:</b> IXLRSNCODEASYNCH</p> <p><b>Meaning:</b> The request could not be processed synchronously. It will be processed asynchronously. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed.</p> <ul style="list-style-type: none"> <li>• If MODE=SYNCECB was specified, the ECB specified for REQECB will be posted when the request has finished.</li> <li>• If MODE=SYNCEXIT was specified, the connection's complete exit will be given control when the request has finished.</li> <li>• If MODE=SYNCTOKEN was specified, an asynchronous request token was returned in the area specified by REQTOKEN. This request token may be specified on the IXLFCOMP macro to determine when the request has finished.</li> </ul> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=SYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=SYNCEXIT, the connection's complete exit will be given control when the request has completed.</li> <li>• If you specified MODE=SYNCTOKEN, use REQTOKEN when calling IXLFCOMP to determine when the request has completed.</li> </ul>

Table 59. Return and Reason Codes for IXLLIST REQUEST=DELETE\_MULT Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	xxxx0409	<p><b>Equate Symbol:</b> IXLRSNCODETIMEOUT</p> <p><b>Meaning:</b> The request completed prematurely because it exceeded the coupling facility model-dependent time-out criteria. The following information has been returned in the answer area:</p> <ul style="list-style-type: none"> <li>• The number of entries that were deleted (field LAADELCNT)</li> <li>• A token for restarting the request (field LAARESTOKEN or LAAEXTRESTOKEN)</li> </ul> <p><b>Action:</b> Reissue the request using the restart token (RESTOKEN or EXTRESTOKEN). For more information about premature completion, see <a href="#">z/OS MVS Programming: Sysplex Services Guide</a>.</p>
4	xxxx0410	<p><b>Equate Symbol:</b> IXLRSNCODELOCKCOND</p> <p><b>Meaning:</b> For a LOCKOPER=HELDDBY request, or a LOCKMODE=COND request, or a request that specified LOCKCOMP, the request could not be completed successfully because the specified lock is not currently held as required. The connection identifier of the lock owner is returned in the answer area (LAACONID field).</p> <p><b>Action:</b> Retry the request, or obtain the lock as required, and retry the request. If you are unable to get the lock, check the ID in the LAACONID field and determine if some recovery is necessary.</p>
4	xxxx0412	<p><b>Equate Symbol:</b> IXLRSNCODELOCKHELDDBYSYS</p> <p><b>Meaning:</b> The lock is not held by any connection, but instead is held by the system. For a request that specified any of the following, the request could not be completed successfully because the specified lock is not currently held as required:</p> <ul style="list-style-type: none"> <li>• LOCKOPER=NOTHELD with either of: <ul style="list-style-type: none"> <li>– LOCKMODE=COND</li> <li>– LOCKCOMP</li> </ul> </li> <li>• LOCKOPER=HELDDBY</li> </ul> <p><b>Action:</b> Retry the request, or obtain the lock as required, and retry the request.</p>

Table 59. Return and Reason Codes for IXLLIST REQUEST=DELETE\_MULT Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0801	<p><b>Equate Symbol:</b> IXLRSNCODEBADPARMLIST</p> <p><b>Meaning:</b> Program error. The parameter list for this request is not addressable. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Verify that:</p> <ul style="list-style-type: none"> <li>• The parameter list address is uncorrupted.</li> <li>• The parameter list is addressable in the caller's primary address space.</li> <li>• If you are issuing this macro while disabled, the parameter list resides in either page-fixed or DREF storage.</li> <li>• If you are issuing this macro in AR-mode and you specified the parameter list address using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If you are issuing this macro in AR-mode, SYSSTATE ASCENV=AR must be issued before issuing this macro.</li> </ul>
8	xxxx0804	<p><b>Equate Symbol:</b> IXLRSNCODEBADVERSIONNUM (or IXLRSNCODEBADVERSION#)</p> <p><b>Meaning:</b> The version number in the macro parameter list is not compatible with the level of XES currently being used. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that your program did not overlay the parameter list storage.</li> <li>• Verify that your program was assembled with the correct macro library for the release of MVS on which your program is running.</li> </ul>

Table 59. Return and Reason Codes for IXLLIST REQUEST=DELETE\_MULT Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx080A	<p><b>Equate Symbol:</b> IXLRSNCODEBADCONTOKEN</p> <p><b>Meaning:</b> Program error. The specified connect token (CONTOKEN) was not valid for one of the following reasons:</p> <ol style="list-style-type: none"> <li>1. The user with the connection identifier represented by the token has disconnected from the structure.</li> <li>2. The connector's task (the task that issued IXLCONN) ended.</li> <li>3. The specified token is not the token that was returned from IXLCONN.</li> <li>4. The request was issued from an address space other than the address space in which IXLCONN was issued.</li> <li>5. The connect token was invalidated during rebuild.</li> <li>6. The connect token was invalidated by XES.</li> </ol> <p><b>Note:</b> The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Take the action with the corresponding meaning.</p> <ol style="list-style-type: none"> <li>1. Discontinue use of the structure. Perform recovery and cleanup for the structure and check the protocol for the use of the structure to determine why you did not know of the disconnection.</li> <li>2. Discontinue use of the structure. Perform recovery and cleanup for the structure and check the protocol for the use of the structure to determine why you did not know that the task ended.</li> <li>3. The contoken is returned in the CONACONTOKEN field of the answer area specified on the IXLCONN request.</li> <li>4. Issue your request from the same address space the IXLCONN was issued in.</li> <li>5. Wait for the rebuild to complete, and try again.</li> <li>6. Discontinue use of the structure. Perform recovery and cleanup for the structure.</li> </ol>
8	xxxx0824	<p><b>Equate Symbol:</b> IXLRSNCODEWRONGSTRTYPE</p> <p><b>Meaning:</b> Program error. The connection specified by CONTOKEN is not to a list structure.</p> <p><b>Action:</b> Specify the appropriate CONTOKEN value, and rerun your program. The CONTOKEN is returned in the answer area provided on the IXLCONN invocation. The type of structure built is passed on the TYPE parameter of the IXLCONN macro.</p>

Table 59. Return and Reason Codes for IXLLIST REQUEST=DELETE\_MULT Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0838	<p><b>Equate Symbol:</b> IXLRSNCODEBADANSAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by ANSAREA is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The answer area address specified by ANSAREA is valid.</li> <li>• If the caller is running in AR-mode and the ANSAREA parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• The address specified for ANSAREA is addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are calling IXLLIST while disabled, ANSAREA must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLLIST in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLIST macro.</li> </ul>
8	xxxx0839	<p><b>Equate Symbol:</b> IXLRSNCODEBADREQTOKENAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by REQTOKEN is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The request token area specified by REQTOKEN is valid.</li> <li>• If the caller is running in AR-mode and the REQTOKEN parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If you are calling IXLLIST while disabled, REQTOKEN must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLLIST in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLIST macro.</li> </ul>
8	xxxx083D	<p><b>Equate Symbol:</b> IXLRSNCODEBADANSLEN</p> <p><b>Meaning:</b> Program error. The length of the answer area, as specified by ANSLEN, is not sufficient to contain answer area information. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Increase the size of the answer area, and rerun your program. The minimum size of the ANSAREA is in the LAA_LEN field of the IXLYLAA mapping macro.</p>
8	xxxx0846	<p><b>Equate Symbol:</b> IXLRSNCODEBADLOCKINDEX</p> <p><b>Meaning:</b> Program error. The specified LOCKINDEX exceeds the size of the lock table for the structure.</p> <p><b>Action:</b> Correct LOCKINDEX to specify an index that is contained within the lock table. The maximum value for the LOCKINDEX is one less than the value specified by the LOCKENTRIES parameter on the IXLCONN macro.</p>

Table 59. Return and Reason Codes for IXLLIST REQUEST=DELETE\_MULT Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0847	<p><b>Equate Symbol:</b> IXLRSNCODEBADLISTNUMBER</p> <p><b>Meaning:</b> Program error. The specified LISTNUM value exceeds the number of lists for the structure.</p> <p><b>Action:</b> Correct LISTNUM to specify a list number that exists in the structure. The number of lists allocated for a structure is determined on the LISTHEADERS parameter of the IXLCONN macro. The list numbers go from 0 to n-1, where n is the number of lists specified.</p>
8	xxxx0849	<p><b>Equate Symbol:</b> IXLRSNCODEBADRESTOKEN</p> <p><b>Meaning:</b> Program error. The restart token specified by RESTOKEN is not valid.</p> <p><b>Action:</b> Determine why the restart token cannot be used to restart the request. Possible causes are:</p> <ul style="list-style-type: none"> <li>• The specified token does not correspond to the restart token returned in the answer area of the previous request (field LAARESTOKEN).</li> <li>• The user specified RESTOKEN when EXTRESTOKEN was required.</li> <li>• The user specified EXTRESTOKEN when RESTOKEN was required.</li> </ul> <p>For more information about premature request completion, see <a href="#">z/OS MVS Programming: Sysplex Services Guide</a>.</p>
8	xxxx084B	<p><b>Equate Symbol:</b> IXLRSNCODENOLOCKS</p> <p><b>Meaning:</b> Program error. A locking operation was requested, but the structure does not contain a lock table.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• You are connected to the intended structure.</li> <li>• You intended to perform a locking operation.</li> </ul> <p>The use of locks is determined by the LOCKENTRIES keyword on the IXLCONN macro.</p>
8	xxxx0851	<p><b>Equate Symbol:</b> IXLRSNCODENOSUSPENDISABLE</p> <p><b>Meaning:</b> Program error. The request failed because MODE=SYNCSUSPEND was specified, but the caller is disabled and cannot be suspended.</p> <p><b>Action:</b> Either specify another MODE value, or become enabled (release the CPU lock), then reissue the request.</p>

Table 59. Return and Reason Codes for IXLLIST REQUEST=DELETE\_MULT Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0854	<p><b>Equate Symbol:</b> IXLRSNCODEBADLOCKCOMP</p> <p><b>Meaning:</b> Program error. The connection identifier specified for LOCKCOMP is not valid.</p> <p><b>Action:</b> The connection identifier is returned in the answer area (mapped by IXLYCONA) when the IXLCONN macro was issued.</p>
8	xxxx0859	<p><b>Equate Symbol:</b> IXLRSNCODEBADLISTAUTH</p> <p><b>Meaning:</b> The list authority specified for AUTHCOMP failed the comparison specified by AUTHCOMPTYPE for the list authority of the specified list. The current list authority (field LAALISTAUTH) and description (field LAALISTDESC) are returned in the answer area.</p> <p><b>Action:</b> None required; however you might want to take some action depending on your application. The list authority is set to binary zeros when the structure is allocated. When IXLLIST is issued with the NEWAUTH keyword, the list authority is changed if the correct comparison list authority value is specified. Check the answer area to determine the list authority value for the list specified. Verify that the correct list number was specified.</p>
8	xxxx0887	<p><b>Equate Symbol:</b> IXLRSNCODEBADEXTRESTOKEN</p> <p><b>Meaning:</b> Program error. The extended restart token specified by EXTRESTOKEN is not valid. The specified token refers to an older instance of the target structure.</p> <p><b>Action:</b> Reinitiate the request with an EXTRESTOKEN value of zero. A system-managed process occurred between the time a request returned the extended restart token and the time the connector tried to continue the request using that token. Discard the results of the initial request and reissue the request. For more information about restarting requests, see <a href="#">z/OS MVS Programming: Sysplex Services Guide</a>.</p>
C	xxxx0C06	<p><b>Equate Symbol:</b> IXLRSNCODENOCNN</p> <p><b>Meaning:</b> Environmental error. This system does not have connectivity to the coupling facility that contains the list structure. Possible reasons for this are:</p> <ul style="list-style-type: none"> <li>• The operator issued VARY PATH,OFFLINE.</li> <li>• The operator issued CONFIG CHP,OFFLINE.</li> <li>• Hardware errors to the coupling facility.</li> <li>• Facility or path failure to the coupling facility.</li> </ul> <p><b>Action:</b> Begin rebuilding the structure on a different coupling facility, or disconnect from the structure.</p>



Table 59. Return and Reason Codes for IXLLIST REQUEST=DELETE\_MULT Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C13	<p><b>Equate Symbol:</b> IXLRSNCODEREQPURGED</p> <p><b>Meaning:</b> Environmental error. The request was purged prior to completion of the request. Possible reasons include:</p> <ul style="list-style-type: none"> <li>• The connector failed.</li> <li>• The connector disconnected.</li> <li>• The requestor failed.</li> <li>• The request was purged by IXLPURGE.</li> <li>• Requests were purged when the connector provided an IXLEERSP response for the Rebuild Stop or Rebuild Cleanup event.</li> <li>• The secondary address space was no longer valid.</li> </ul> <p><b>Action:</b> None if this is expected. Otherwise, determine why the connector failed.</p>
C	xxxx0C14	<p><b>Equate Symbol:</b> IXLRSNCODESTATUSUNKNOWN</p> <p><b>Meaning:</b> Environmental error. The request has completed, but the final disposition of the request cannot be determined. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the validity of your data by comparing the expected results with what is in the coupling facility.</li> <li>• Consider running rebuild processing to move the structure to attempt to protect it from further errors of this kind.</li> </ul>
C	xxxx0C25	<p><b>Equate Symbol:</b> IXLRSNCODESTRFAILURE</p> <p><b>Meaning:</b> Environmental error. The list structure failed prior to completion of the request.</p> <p><b>Action:</b> Either rebuild or disconnect from the structure.</p>
C	xxxx0CA0	<p><b>Equate Symbol:</b> IXLRSNCODEQUIESCEDSUSPENDFAIL</p> <p><b>Meaning:</b> Environmental error. The request is failed because the structure is quiesced for a system-managed process and SUSPEND=FAIL is specified on the IXLCONN.</p> <p><b>Action:</b> None, if this is expected.</p>
C	xxxxFFFF	<p><b>Equate Symbol:</b> IXLRSNCODENOTAVAILABLE</p> <p><b>Meaning:</b> Environmental error. XES functions are not available. This could occur because the coupling facility hardware necessary to provide XES functions is not present.</p> <p><b>Action:</b> Re-IPL the system, or follow your particular failure management protocol.</p>

Table 59. Return and Reason Codes for IXLLIST REQUEST=DELETE\_MULT Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
10	xxxx10xx	<b>Meaning:</b> System error. XES processing failure. The state of the involved structure and the disposition of the request are unpredictable. The answer area (ANSAREA) fields do not contain valid information. <b>Action:</b> Contact the IBM support center.

## Chapter 60. IXLLIST REQUEST=DEQ\_EVENTQ

### Description

A DEQ\_EVENTQ request allows you to read and dequeue event monitor controls from your event queue. The EMCs are returned in the storage area specified by the BUFFER or BUFLIST parameter. The information for each EMC object is mapped by the IXLYEMC macro

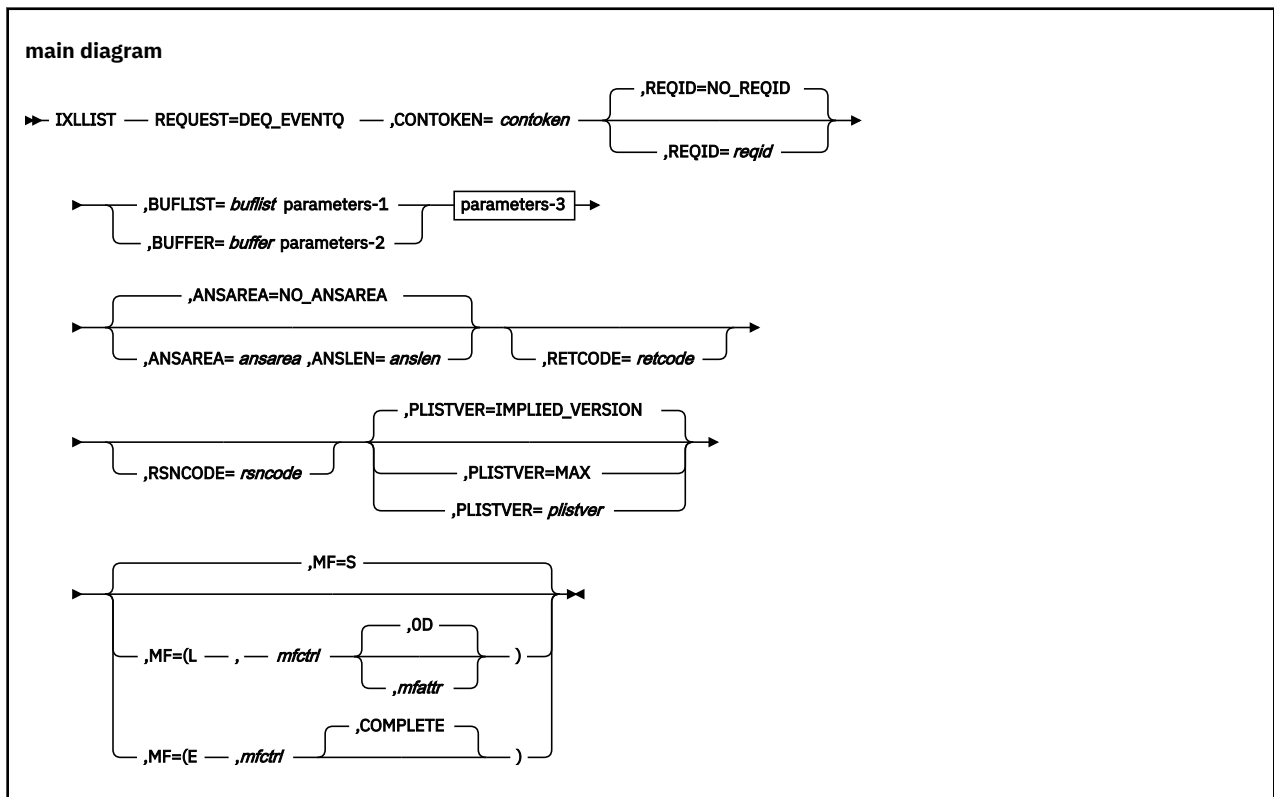
A DEQ\_EVENTQ request can complete before all the events have been read off your event queue. If this occurs, you are notified by a return code indicating that more events remain on the event queue. The count of EMCs that are still queued to the event queue is returned in the answer area.

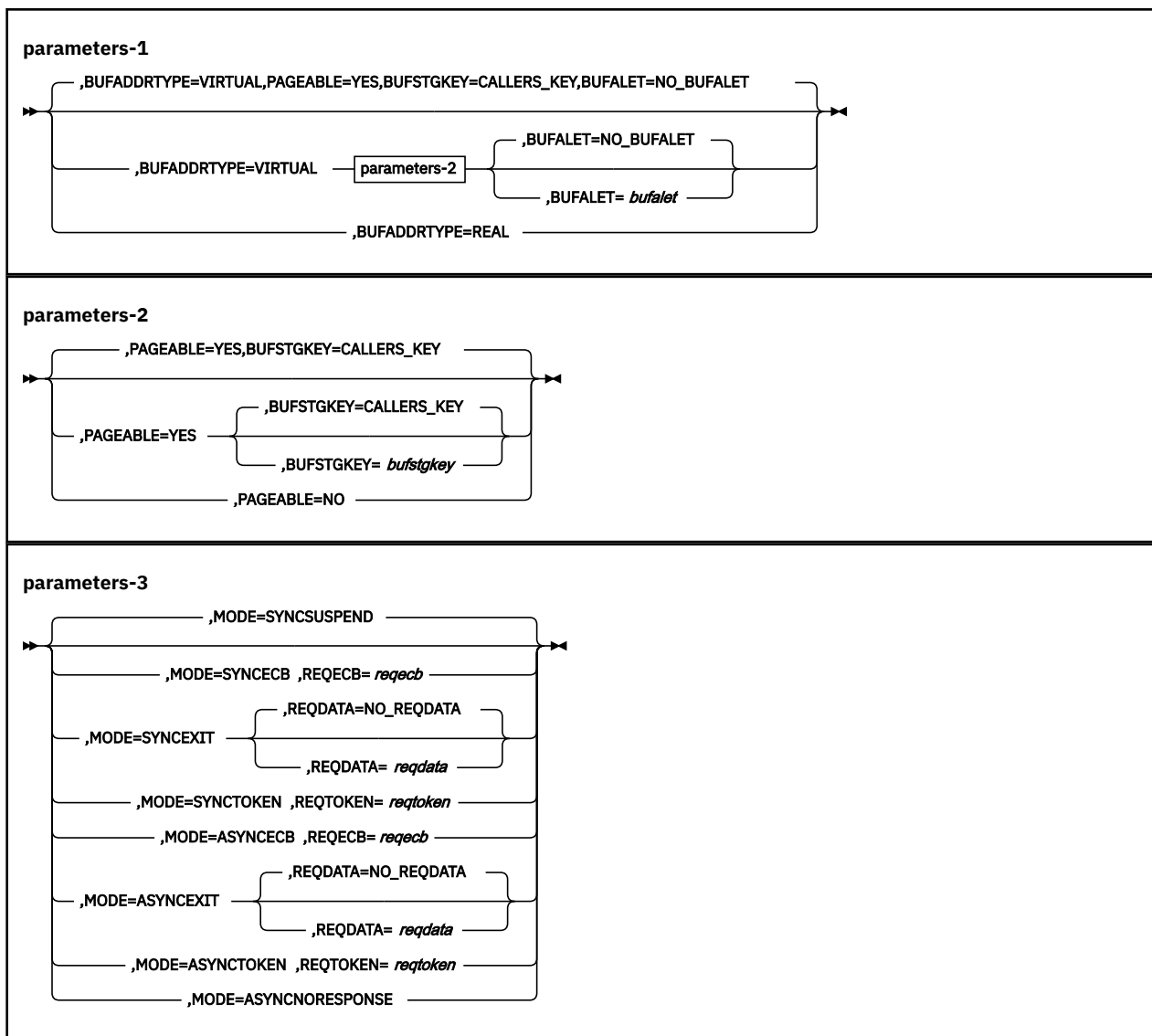
In the case of a premature completion of an IXLLIST REQUEST=DEQ\_EVENTQ request, all prior EMCs were processed successfully. To continue to read and dequeue EMCs from your event queue, reissue the IXLLIST REQUEST=DEQ\_EVENTQ as many times as is required after processing the previous contents of the BUFFER or BUFLIST storage area.

A DEQ\_EVENTQ request can be issued only for keyed list structures allocated in a coupling facility of CFLEVEL=3 or higher. DEQ\_EVENTQ requests issued for a list structure allocated in a coupling facility of CFLEVEL 0, 1, or 2 will fail.

### Syntax Diagram

The syntax diagram for IXLLIST REQUEST=DEQ\_EVENTQ is as follows:





**Note:** If MODE=SYNCTOKEN or MODE=ASYNCTOKEN is specified, ANSAREA= *ansarea* and ANSLN= *anslen* are required.

## Parameter Descriptions

The parameter descriptions for REQUEST=DEQ\_EVENTQ are listed in alphabetical order. Default values are underlined:

### REQUEST=DEQ\_EVENTQ

Use this input parameter to read and dequeue event monitor controls (EMCs) from the user's event queue.

#### ,ANSAREA=NO\_ANSAREA

#### ,ANSAREA=ansarea

Use this output parameter to specify an answer area to contain information returned from the request. The format of the answer area is described by mapping macro IXLYLAA.

When the request completes successfully or completes prematurely, the answer area contains the number of event monitor controls still queued to the event queue (field LAADEQ\_EMQUEUEDCNT) and the number of event monitor controls that were read and dequeued (field LAADEQ\_NUMEMCREAD).

See *z/OS MVS Programming: Sysplex Services Guide* for a description of all the relevant IXLYLAA fields for this request.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an area (with a length of ANSLEN) to contain the information returned by the request.

**,ANSLEN=*anslen***

Use this input parameter to specify the size of the storage area specified by ANSAREA.

Check the prologue of the IXLYLAA mapping macro for the minimum required size of the answer area, or use the length field (LAA\_LEN) in the LAA.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 2-byte field that contains the size, in bytes, of the answer area.

**,BUFADDRTYPE=VIRTUAL**

**,BUFADDRTYPE=REAL**

Use this input parameter to specify whether the buffer addresses specified in the BUFLIST list are virtual storage or real storage addresses.

**VIRTUAL**

The buffer addresses are virtual storage addresses. The virtual storage can be pageable or nonpageable. See the PAGEABLE parameter for information about managing storage binds when specifying virtual storage addresses.

**REAL**

The buffer addresses are real storage addresses.

It is the caller's responsibility to manage the binds between the data buffer virtual storage and the real storage addresses provided. The caller must ensure that the data buffer virtual storage remains bound to the real storage addresses provided until the request completes.

**,BUFALET=NO BUFALET**

**,BUFALET=*bufalet***

Use this input parameter to specify an access list entry token (ALET) to be used in referencing all of the buffers specified by BUFLIST.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 4-byte field that contains the ALET.

**,BUFFER=*buffer***

Use this output parameter to specify a buffer area to hold the event monitor controls (EMC) objects that have been read and dequeued from your event queue. Only 31-bit addressable (below 2GB) virtual storage areas are supported for the BUFFER specification.

You must define the buffer to be 4096 bytes on a 4096-byte boundary. The buffer must not start below storage address 512.

Once the request completes, the buffer contains, starting at offset zero, an array of event monitor controls that were dequeued from the user's event queue. The format of each array element is described by mapping macro IXLYEMC.

**Note:** You cannot code BUFFER with MODE=ASYNCRESPONSE.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 4096-byte area to contain the data returned by the request.

**,BUFLIST=*buflist***

Use this input parameter to specify a buffer to hold the returned event monitor controls (EMC) objects that have been read and dequeued from your event queue.

BUFLIST specifies a 128-byte storage area that contains the address of a single buffer. The address of the buffer should be placed in the second four bytes of the 128-byte area, beginning at offset zero. The first four bytes are reserved and the rest of the area is ignored.

You must define the buffer to be 4096 bytes on a 4096-byte boundary. The buffer must not start below storage address 512.

Once the request completes, the buffer contains, starting at offset zero in the buffer, an array of event monitor controls that were dequeued from the user's event queue. The format of each array element is described by the mapping macro IXLYEMC.

**Note:** You cannot code BUFLIST with MODE=ASYNCHRESPONSE.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 128-byte area that contains the address of the buffer.

**,BUFSTGKEY=CALLERS\_KEY**

**,BUFSTGKEY=bufstgkey**

Use this input parameter to specify a storage key that you define and use when referencing the buffers specified by BUFLIST or the buffer that is specified by BUFFER.

If you do not specify BUFSTGKEY, or if you specify BUFSTGKEY=CALLERS\_KEY, all references to one or more buffers are performed by using the caller's PSW key.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of an 8-bit field that contains the storage key in the format B'kkkkxxxx', where kkkk is the key and xxxx is ignored.

**,CONTOKEN=contoken**

Use this input parameter to specify the connect token that was returned by the IXLCONN service. The connect token uniquely identifies your connection to the list structure, and must be specified on each IXLLIST invocation.

The connect token is available in the IXLCONN answer area that is mapped by IXLYCONA.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 16-byte field that contains the connect token.

**,MF=S**

**,MF=(L,mfctrl)**

**,MF=(L,mfctrl,mfattr)**

**,MF=(L,mfctrl,0D)**

**,MF=(M,mfctrl)**

**,MF=(M,mfctrl,COMPLETE)**

**,MF=(M,mfctrl,NOCHECK)**

**,MF=(E,mfctrl)**

**,MF=(E,mfctrl,COMPLETE)**

**,MF=(E,mfctrl,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

**,mfctrl**

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

**,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to

force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

**,COMPLETE**

**,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

**COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if *SMILE=var* were an optional parameter and the default is *SMILE=NO\_SMILE* then it would not be documented. However, if the default was *SMILE=-:-*, then it would be documented because a value would be the default.

**NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

**,MODE=SYNCSUSPEND**

**,MODE=SYNCECB**

**,MODE=SYNCEXIT**

**,MODE=SYNCTOKEN**

**,MODE=ASYNCECB**

**,MODE=ASYNCEXIT**

**,MODE=ASYNCTOKEN**

**,MODE=ASYNCSUSPEND**

Use this input parameter to specify:

- Whether the request is to be performed synchronously or asynchronously
- How you want to be notified of request completion

**MODE=SYNCSUSPEND**

The request is suspended until it can complete synchronously. To use this option, your program must be enabled for I/O and external interrupts.

**MODE=SYNCECB**

The request processes synchronously if possible. If the request processes asynchronously, the ECB specified by REQECB is posted when the request completes.

**MODE=SYNCEXIT**

The request processes synchronously if possible. If the request processes asynchronously, your complete exit is given control when the request completes.

**MODE=SYNCTOKEN**

The request processes synchronously if possible. If the request processes asynchronously, an asynchronous request token is returned in the area specified by REQTOKEN on invocation of the request. Use the returned request token on the IXLFCOMP macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=SYNCTOKEN is specified.

**MODE=ASYNCECB**

The request processes asynchronously. The ECB specified by REQECB is posted when the request completes.

**MODE=ASYNCEXIT**

The request processes asynchronously. Your complete exit is given control when the request completes.

**MODE=ASYNCTOKEN**

The request processes asynchronously. An asynchronous request token is returned in the area specified by REQTOKEN upon invocation of the request. Use the returned request token on the IXLFCOMP macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=ASYNCTOKEN is specified.

**MODE=ASYNCSNORESPONSE**

The request processes asynchronously. No notification of request completion is provided. The answer area (ANSAREA) fields will not contain valid information when this mode is specified.

**Note:** You cannot code MODE=ASYNCSNORESPONSE with LOCKINDEX, BUFFER, or BUFLIST.

**,PAGEABLE=YES****,PAGEABLE=NO**

Use this input parameter to identify whether the storage areas specified by BUFFER or BUFLIST are in pageable or potentially pageable storage.

**YES**

Specify this option to indicate that the BUFFER or BUFLIST buffers reside in pageable virtual storage. XES performs the required page fixing to fix the buffers in real storage while the cache or list request transfers data to or from the coupling facility.

This includes storage obtained from pageable subpools, disabled reference (DREF) storage, and may include storage that has the potential to become pageable during the processing of a request. (An example is address space storage owned by any swappable address space, for which a PGSER FIX has been successfully processed, but for which the owning address space gets swapped during processing of a cache or list request.) This does not include implicitly non-pageable storage (for example, storage obtained from non-pageable subpools).

The system takes responsibility for managing binds to central storage for the duration of the cache or list request, regardless of what address space owns the storage or whether the storage-owning address space is swappable or nonswappable. The storage can be owned by any address space.

High shared virtual storage areas (above 2GB) may not be used.

**NO**

Specify this option to indicate that the BUFFER or BUFLIST buffers reside in non-pageable virtual storage. XES does not page fix the buffers in real storage.

This includes implicitly non-pageable storage areas (for example, storage obtained from non-pageable subpools), and may include storage that has the potential to become pageable during the processing of a request (An example is address space storage owned by any swappable address space, for which a PGSER FIX has been successfully processed, but for which the owning address space gets swapped-out during processing of a cache or list request.)

The system takes responsibility for managing binds to central storage for the duration of the cache or list request, if and only if the non-pageable storage is owned by either the requester's address space or the connector's address space. If the storage is owned by any other address space, then the invoker is responsible for ensuring that the virtual storage remains non-pageable for the duration of the request (including the case in which the storage is owned by a swappable address space that is swapped during processing of an IXLCACHE or IXLLIST request). Subject to this consideration, the storage can be owned by any address space. See [z/OS MVS Programming: Sysplex Services Guide](#).

**,PLISTVER=IMPLIED\_VERSION****,PLISTVER=MAX****,PLISTVER=plistver**

Use this input parameter to specify the version of the macro. See [“Understanding IXLLIST Version Support” on page 965](#) for a description of the options available with PLISTVER.



**,REQDATA=NO\_REQDATA****,REQDATA=reqdata**

Use this input parameter with MODE=SYNCEXIT or MODE=ASYNCEXIT to pass any data you choose to the complete exit. The exit will get control only if the request is processed asynchronously.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the data to be passed to the complete exit.

**,REQECB=reqecb**

Use this output parameter with either MODE=SYNCECB or MODE=ASYNCECB to specify the address of an ECB, which is to be posted when the request completes if the request was processed asynchronously.

Before coding REQECB, you must ensure that:

- You initialize the ECB before you issue the request.
- The ECB resides in either common storage or the home address space where IXLCONN was issued.
- Any tasks that wait for the ECB to be posted reside in the home address space where IXLCONN was issued.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that contains the address of the ECB to be posted when the request completes. The ECB must be aligned on a fullword boundary.

**,REQID=NO\_REQID****,REQID=reqid**

Use this input parameter to specify a user-defined request identifier to be associated with the request. You can specify this request identifier on the IXPURGE macro to cancel a request that has not yet been processed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the user-defined request identifier.

**,REQTOKEN=reqtoken**

Use this output parameter with either MODE=SYNCTOKEN or MODE=ASYNCTOKEN to specify the address of a storage area to receive the request token that is returned when the request will be processed asynchronously. This token, which uniquely identifies the request, must be used as input to the IXLFCOMP macro, which you use to determine if the request has completed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field where the system will put the request token.

**,RETCODE=retcode**

Use this output parameter to specify a field to contain the return code. (The return code is also returned in register 15.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the return code when the request has completed.

**,RSNCODE=rsncode**

Use this output parameter to specify a field to contain the reason code returned, if applicable. (The reason code is also returned in register 0.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the reason code (if any) when the request has completed.

## ABEND Codes

---

Abend X'026' (See *z/OS MVS System Codes* for more information on this abend.)

## Return and Reason Codes

---

When the system returns control to your program:

- GPR 15 (and RETCODE, if specified) contains a return code.
- If the return code is not zero, GPR 0 (and RSNCODE, if specified) contains a reason code.

**Note:** The return and reason codes will also be put into the answer area (mapped by IXLYLAA), if there is one, when the request completes.

The IXLYCON macro provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

<b>0</b>	IXLRETCODEOK
<b>4</b>	IXLRETCODEWARNING
<b>8</b>	IXLRETCODEPARMERROR
<b>C</b>	IXLRETCODEENVERROR
<b>10</b>	IXLRETCODECOMPERROR

The following table contains hexadecimal return and reason codes, the equate symbols associated with each reason code, and the meaning and suggested action for each return and reason code.

Table 60. Return and Reason Codes for IXLLIST REQUEST=DEQ_EVENTQ Macro		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0	None.	<p><b>Meaning:</b> If you specified a MODE value of ASYNCECB, ASYNCEXIT, ASYNCTOKEN, or ASYNCSNORESPONSE, the request has been successfully initiated. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed. For any other MODE value, the request has successfully completed.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=ASYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=ASYNCEXIT, the connection's complete exit will be given control when the request has completed.</li> <li>• If you specified MODE=ASYNCTOKEN, use REQTOKEN when calling IXLFCOMP to determine when the request has completed.</li> </ul>

Table 60. Return and Reason Codes for IXLLIST REQUEST=DEQ\_EVENTQ Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	xxxx0402	<p><b>Equate Symbol:</b> IXLRNCODEASYNCH</p> <p><b>Meaning:</b> The request could not be processed synchronously. It will be processed asynchronously. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed.</p> <ul style="list-style-type: none"> <li>• If MODE=SYNCECB was specified, the ECB specified for REQECB will be posted when the request has finished.</li> <li>• If MODE=SYNCEXIT was specified, the connection's complete exit will be given control when the request has finished.</li> <li>• If MODE=SYNCTOKEN was specified, an asynchronous request token was returned in the area specified by REQTOKEN. This request token may be specified on the IXLFCOMP macro to determine when the request has finished.</li> </ul> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=SYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=SYNCEXIT, the connection's complete exit will be given control when the request has completed.</li> <li>• If you specified MODE=SYNCTOKEN, use REQTOKEN when calling IXLFCOMP to determine when the request has completed.</li> </ul>
4	xxxx0409	<p><b>Equate Symbol:</b> IXLRNCODETIMEOUT</p> <p><b>Meaning:</b> The request completed prematurely because it exceeded the coupling facility model-dependent time-out criteria. The following information has been returned in the answer area:</p> <ul style="list-style-type: none"> <li>• The number of event monitor controls that were read and dequeued from the event queue (field LAADEQ_NUMEMCREAD).</li> <li>• The number of event monitor controls that remain on the event queue (field LAADEQ_EMCQUEUEDCNT).</li> </ul> <p><b>Action:</b> Reissue the request to continue.</p> <p>Be sure to process the information returned from this request before reissuing the request. The data returned from this request will be overwritten if you specify the same buffer address. Continue to reissue the request until the return code indicates that all processing has completed.</p> <p>For more information about premature completion of a DEQ_EVENTQ request, see <a href="#">z/OS MVS Programming: Sysplex Services Guide</a>.</p>

Table 60. Return and Reason Codes for IXLLIST REQUEST=DEQ\_EVENTQ Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0801	<p><b>Equate Symbol:</b> IXLRSNCODEBADPARMLIST</p> <p><b>Meaning:</b> Program error. The parameter list for this request is not addressable. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Verify that:</p> <ul style="list-style-type: none"> <li>• The parameter list address is uncorrupted.</li> <li>• The parameter list is addressable in the caller's primary address space.</li> <li>• If you are issuing this macro while disabled, the parameter list resides in either page-fixed or DREF storage.</li> <li>• If you are issuing this macro in AR-mode and you specified the parameter list address using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If you are issuing this macro in AR-mode, SYSSTATE ASCENV=AR must be issued before issuing this macro.</li> </ul>
8	xxxx0804	<p><b>Equate Symbol:</b> IXLRSNCODEBADVERSIONNUM (or IXLRSNCODEBADVERSION#)</p> <p><b>Meaning:</b> The version number in the macro parameter list is not compatible with the level of XES currently being used. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that your program did not overlay the parameter list storage.</li> <li>• Verify that your program was assembled with the correct macro library for the release of MVS on which your program is running.</li> </ul>

Table 60. Return and Reason Codes for IXLLIST REQUEST=DEQ\_EVENTQ Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx080A	<p><b>Equate Symbol:</b> IXLRNCODEBADCONTOKEN</p> <p><b>Meaning:</b> Program error. The specified connect token (CONTOKEN) was not valid for one of the following reasons:</p> <ol style="list-style-type: none"> <li>1. The user with the connection identifier represented by the token has disconnected from the structure.</li> <li>2. The connector's task (the task that issued IXLCONN) ended.</li> <li>3. The specified token is not the token that was returned from IXLCONN.</li> <li>4. The request was issued from an address space other than the address space in which IXLCONN was issued.</li> <li>5. The connect token was invalidated during rebuild.</li> <li>6. The connect token was invalidated by XES.</li> </ol> <p><b>Note:</b> The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Take the action with the corresponding meaning.</p> <ol style="list-style-type: none"> <li>1. Discontinue use of the structure. Perform recovery and cleanup for the structure and check the protocol for the use of the structure to determine why you did not know of the disconnection.</li> <li>2. Discontinue use of the structure. Perform recovery and cleanup for the structure and check the protocol for the use of the structure to determine why you did not know that the task ended.</li> <li>3. The contoken is returned in the CONACONTOKEN field of the answer area specified on the IXLCONN request.</li> <li>4. Issue your request from the same address space the IXLCONN was issued in.</li> <li>5. Wait for the rebuild to complete, and try again.</li> <li>6. Discontinue use of the structure. Perform recovery and cleanup for the structure.</li> </ol>
8	xxxx0824	<p><b>Equate Symbol:</b> IXLRNCODEWRONGSTRTYPE</p> <p><b>Meaning:</b> Program error. The connection specified by CONTOKEN is not to a list structure.</p> <p><b>Action:</b> Specify the appropriate CONTOKEN value, and rerun your program. The CONTOKEN is returned in the answer area provided on the IXLCONN invocation. The type of structure built is passed on the TYPE parameter of the IXLCONN macro.</p>

Table 60. Return and Reason Codes for IXLLIST REQUEST=DEQ\_EVENTQ Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0833	<p><b>Equate Symbol:</b> IXLRSNCODEBADPGBLATTR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list is specified as being pageable (PAGEABLE=YES), but is not.</p> <p><b>Action:</b> Change the buffer area(s) to pageable storage, or specify PAGEABLE=NO. See the PAGEABLE parameter description for specification instructions.</p>
8	xxxx0834	<p><b>Equate Symbol:</b> IXLRSNCODEBADNONPGBLATTR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list, is specified as being nonpageable (PAGEABLE=NO), but is either pageable or not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The type of storage in the buffer area corresponds to the value you specified on the PAGEABLE parameter. See the PAGEABLE parameter description for specification instructions.</li> <li>• The correct buffer address was used.</li> <li>• The buffer area was not previously freed.</li> <li>• If you are calling IXLLIST while disabled, the buffers must reside in either page-fixed or DREF storage.</li> <li>• The buffer areas were allocated in a storage key that matches the key specified by the BUFSTGKEY parameter, or, if BUFSTGKEY is omitted, the caller's PSW key.</li> <li>• If BUFLIST was specified and your program is running in AR-mode: <ul style="list-style-type: none"> <li>– If the BUFFER or BUFLIST parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>– The BUFALET specification is correct.</li> <li>– You specified SYSSTATE ASCENV=AR before issuing the IXLLIST macro.</li> </ul> </li> </ul>

Table 60. Return and Reason Codes for IXLLIST REQUEST=DEQ\_EVENTQ Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0835	<p><b>Equate Symbol:</b> IXLRNCODEBADDATAADDR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list, is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The correct buffer address was used.</li> <li>• The buffer area was not previously freed.</li> <li>• The buffer area was allocated in a storage key that matches the key specified using the BUFSTGKEY parameter, or, if BUFSTGKEY is omitted, the caller's PSW key.</li> <li>• If BUFLIST was specified and your program is running in AR mode: <ul style="list-style-type: none"> <li>– The BUFALET specification is correct.</li> <li>– You specified SYSSTATE ASCENV=AR before issuing the macro.</li> </ul> </li> </ul>
8	xxxx0836	<p><b>Equate Symbol:</b> IXLRNCODEBADREALADDR</p> <p><b>Meaning:</b> Program error. Real storage addresses were provided in the BUFLIST list, but one of the buffers is not addressable in central storage.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that BUFADDRTYPE was specified as you intended.</li> <li>• Ensure that the buffer addresses specified by BUFLIST are valid.</li> </ul>
8	xxxx0838	<p><b>Equate Symbol:</b> IXLRNCODEBADANSAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by ANSAREA is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The answer area address specified by ANSAREA is valid.</li> <li>• If the caller is running in AR-mode and the ANSAREA parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• The address specified for ANSAREA is addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are calling IXLLIST while disabled, ANSAREA must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLLIST in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLIST macro.</li> </ul>

Table 60. Return and Reason Codes for IXLLIST REQUEST=DEQ\_EVENTQ Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0839	<p><b>Equate Symbol:</b> IXLRSNCODEBADREQTOKENAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by REQTOKEN is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The request token area specified by REQTOKEN is valid.</li> <li>• If the caller is running in AR-mode and the REQTOKEN parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If you are calling IXLLIST while disabled, REQTOKEN must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLLIST in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLIST macro.</li> </ul>
8	xxxx083D	<p><b>Equate Symbol:</b> IXLRSNCODEBADANSLEN</p> <p><b>Meaning:</b> Program error. The length of the answer area, as specified by ANSLEN, is not sufficient to contain answer area information. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Increase the size of the answer area, and rerun your program. The minimum size of the ANSAREA is in the LAA_LEN field of the IXLYLAA mapping macro.</p>
8	xxxx084A	<p><b>Equate Symbol:</b> IXLRSNCODENOKEYS</p> <p><b>Meaning:</b> Program error. The structure does not support the use of entry keys. The IXLLIST request type either required the structure to support entry keys or designated a list entry or list position by list number and entry key.</p> <p><b>Action:</b> Ensure that you are connected to the intended structure. The use of keys for a structure is determined by the REFOPTION keyword on the IXLCONN macro.</p>
8	xxxx0851	<p><b>Equate Symbol:</b> IXLRSNCODENOSUSPENDISABLE</p> <p><b>Meaning:</b> Program error. The request failed because MODE=SYNCSUSPEND was specified, but the caller is disabled and cannot be suspended.</p> <p><b>Action:</b> Either specify another MODE value, or become enabled (release the CPU lock), then reissue the request.</p>
8	xxxx0864	<p><b>Equate Symbol:</b> IXLRSNCODEBADBUFSIZE</p> <p><b>Meaning:</b> Program error. The size of the BUFFER area or the buffer areas specified by BUFLIST is not large enough to contain the data being read. No data is returned.</p> <p><b>Action:</b> If more space is available, specify a larger buffer size, and reissue the request.</p>



Table 60. Return and Reason Codes for IXLLIST REQUEST=DEQ\_EVENTQ Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0865	<b>Equate Symbol:</b> IXLRSNCODEBADBUFSPEC <b>Meaning:</b> Program error. There is an error in the buffer specification. <b>Action:</b> Check the following: <ul style="list-style-type: none"> <li>• The requirements for BUFLIST or BUFFER.</li> <li>• Buffer pointer(s) in the BUFLIST.</li> <li>• Buffer boundaries.</li> </ul>
8	xxxx0866	<b>Equate Symbol:</b> IXLRSNCODEBADBUFKEY <b>Meaning:</b> Program error. The buffer storage key specified by BUFSTGKEY is incorrect or BUFSTGKEY was not specified and the PSW key does not match the key of the buffers. The data cannot be stored in the specified buffer area. <b>Action:</b> Check the following: <ul style="list-style-type: none"> <li>• Determine if the key of the storage being used for the buffers is different from the PSW key.</li> <li>• If BUFSTGKEY was specified, verify that the key was put in the correct bits (see the explanation of this parameter for more information).</li> <li>• If you are calling IXLLIST in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLIST macro.</li> </ul>
8	xxxx0867	<b>Equate Symbol:</b> IXLRSNCODEBADBUFLIST <b>Meaning:</b> Program error. The 128-byte storage area specified by BUFLIST is not addressable. <b>Action:</b> Ensure that the address specified by BUFLIST is valid.
8	xxxx08AD	<b>Equate Symbol:</b> IXLRSNCODEBADHIGHSHAREDVIRT <b>Meaning:</b> Program error. The request specified a high shared virtual storage area (above 2GB). <b>Action:</b> None required.
C	xxxx0C06	<b>Equate Symbol:</b> IXLRSNCODENOCNN <b>Meaning:</b> Environmental error. This system does not have connectivity to the coupling facility that contains the list structure. Possible reasons for this are: <ul style="list-style-type: none"> <li>• The operator issued VARY PATH,OFFLINE.</li> <li>• The operator issued CONFIG CHP,OFFLINE.</li> <li>• Hardware errors to the coupling facility.</li> <li>• Facility or path failure to the coupling facility.</li> </ul> <b>Action:</b> Begin rebuilding the structure on a different coupling facility, or disconnect from the structure.

Table 60. Return and Reason Codes for IXLLIST REQUEST=DEQ\_EVENTQ Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C13	<p><b>Equate Symbol:</b> IXLRNCDEREQPURGED</p> <p><b>Meaning:</b> Environmental error. The request was purged prior to completion of the request. Possible reasons include:</p> <ul style="list-style-type: none"> <li>• The connector failed.</li> <li>• The connector disconnected.</li> <li>• The requestor failed.</li> <li>• The request was purged by IXLPURGE.</li> <li>• Requests were purged when the connector provided an IXLEERSP response for the Rebuild Stop or Rebuild Cleanup event.</li> <li>• The secondary address space was no longer valid.</li> </ul> <p><b>Action:</b> None if this is expected. Otherwise, determine why the connector failed.</p>
C	xxxx0C14	<p><b>Equate Symbol:</b> IXLRNCDERSTATUSUNKNOWN</p> <p><b>Meaning:</b> Environmental error. The request has completed, but the final disposition of the request cannot be determined. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the validity of your data by comparing the expected results with what is in the coupling facility.</li> <li>• Consider running rebuild processing to move the structure to attempt to protect it from further errors of this kind.</li> </ul>
C	xxxx0C25	<p><b>Equate Symbol:</b> IXLRNCDERSTRFAILURE</p> <p><b>Meaning:</b> Environmental error. The list structure failed prior to completion of the request.</p> <p><b>Action:</b> Either rebuild or disconnect from the structure.</p>
C	xxxx0C68	<p><b>Equate Symbol:</b> IXLRNCDERBADREQCFLEVEL</p> <p><b>Meaning:</b> Environmental error. The request type is not permitted for the level of coupling facility in which the target structure is allocated.</p> <p><b>Action:</b> Either disconnect from the structure (using IXLDISC) or initiate a rebuild of the structure, if allowed (using IXLREBLD) in a coupling facility of the correct CFLEVEL.</p>
C	xxxx0CA0	<p><b>Equate Symbol:</b> IXLRNCDERQUIESCEDSUSPENDFAIL</p> <p><b>Meaning:</b> Environmental error. The request is failed because the structure is quiesced for a system-managed process and SUSPEND=FAIL is specified on the IXLCONN.</p> <p><b>Action:</b> None, if this is expected.</p>

Table 60. Return and Reason Codes for IXLLIST REQUEST=DEQ\_EVENTQ Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxxFFFF	<b>Equate Symbol:</b> IXLRSNCODENOTAVAILABLE <b>Meaning:</b> Environmental error. XES functions are not available. This could occur because the coupling facility hardware necessary to provide XES functions is not present. <b>Action:</b> Re-IPL the system, or follow your particular failure management protocol.
10	xxxx10xx	<b>Meaning:</b> System error. XES processing failure. The state of the involved structure and the disposition of the request are unpredictable. The answer area (ANSAREA) fields do not contain valid information. <b>Action:</b> Contact the IBM support center.



---

## Chapter 61. IXLLIST REQUEST=LOCK

### Description

---

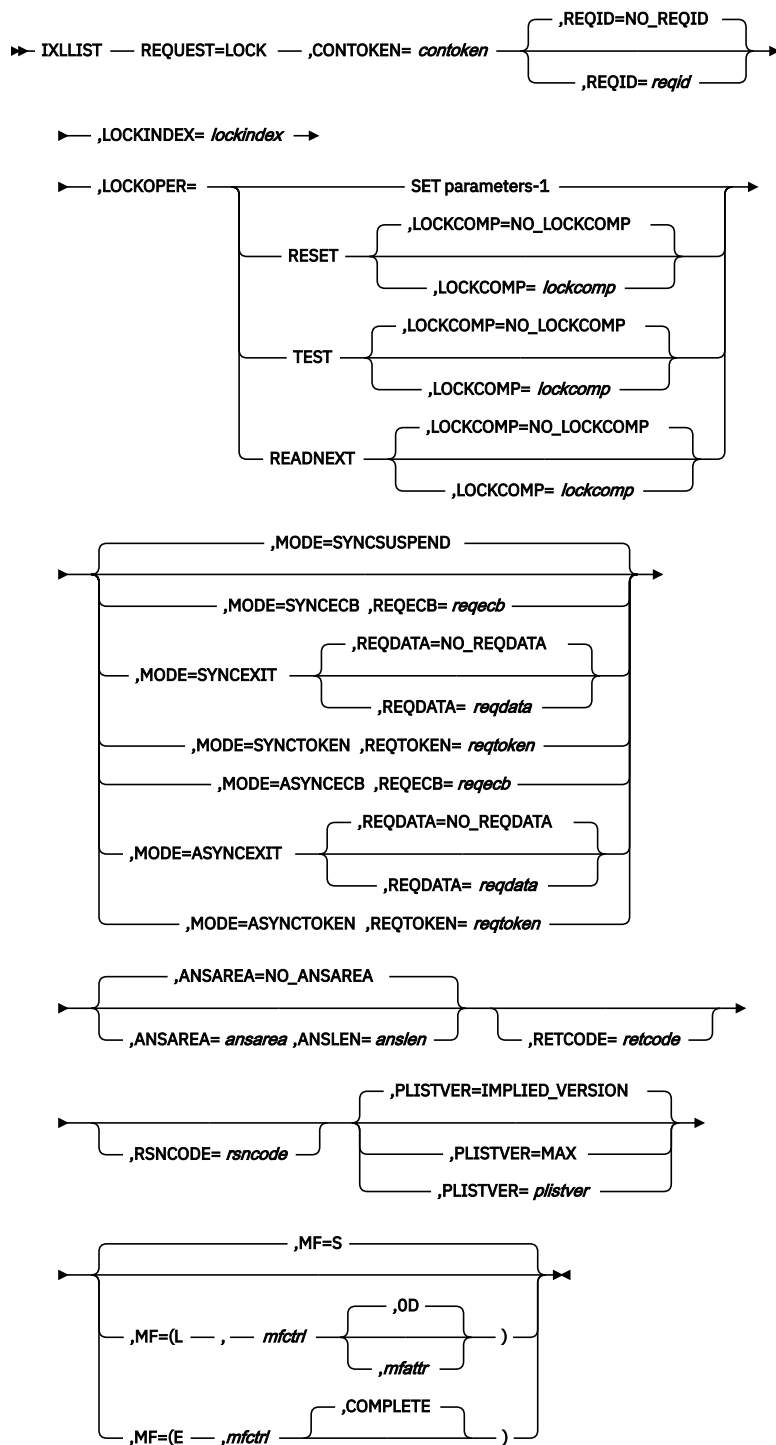
A LOCK request allows you to perform locking functions. The lock designated by LOCKINDEX is operated on as specified by the LOCKOPER parameter. You can use REQUEST=LOCK only for structures that contain a lock table (the LOCKENTRIES parameter on the IXLCONN macro causes a list structure to have a lock table.)

### Syntax Diagram

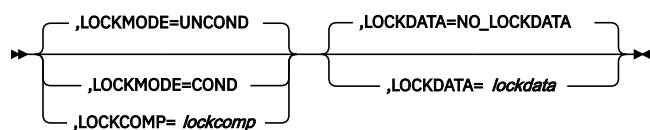
---

The syntax diagram for IXLLIST REQUEST=LOCK is as follows:

## main diagram



## parameters-1



**Note:** If MODE=SYNCTOKEN or MODE=ASYNCTOKEN is specified, ANSAREA=*ansarea* and ANSLen=*anslen* are required.

## Parameter Descriptions

---

The parameter descriptions for REQUEST=LOCK are listed in alphabetical order. Default values are underlined:

### **REQUEST=LOCK**

Use this input parameter to specify that the LOCKINDEX lock be operated on as specified by the LOCKOPER parameter.

### **,ANSAREA=NO ANSAREA**

### **,ANSAREA=*ansarea***

Use this output parameter to specify an answer area to contain information returned from the request. The format of the answer area is described by mapping macro IXLYLAA. Upon successful completion of a request for LOCKOPER=READNEXT request, the answer area contains the connection ID of the connection holding the lock (field LAACONID) and the index of the lock found for a request (field LAALOCKINDEX). For a request with LOCKOPER=TEST, the answer area contains the connection ID of the connection holding the lock (field LAACONID). IXLYLAA fields for this request.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an area (with a length of ANSLen) to contain the information.

### **,ANSLen=*anslen***

Use this input parameter to specify the size of the storage area specified by ANSAREA.

Check the prologue of the IXLYLAA mapping macro for the minimum required size of the answer area, or use the length field (LAA\_LEN) in the LAA.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 2-byte field that contains the size, in bytes, of the answer area.

### **,CONTOKEN=*contoken***

Use this input parameter to specify the connect token that was returned by the IXLCONN service. The connect token uniquely identifies your connection to the list structure, and must be specified on each IXLLIST invocation.

The connect token is available in the IXLCONN answer area that is mapped by IXLYCONA.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 16-byte field that contains the connect token.

### **,LOCKCOMP=NO LOCKCOMP**

### **,LOCKCOMP=*lockcomp***

This parameter has slightly different meanings based on the value that is specified for the LOCKOPER parameter. Generally, this input parameter specifies a connection identifier to be verified as the owner of the lock specified by LOCKINDEX. This verification is a prerequisite to the successful completion of the request.

When LOCKCOMP is specified, the completion of the request is conditional on there being no contention for the lock. If contention exists, the request fails .

The connection identifier is available from the IXLCONN answer area, which is mapped which is by IXLYCONA, in field CONACONID.

The effect of LOCKCOMP is based on the LOCKOPER value specified. See the description under LOCKOPER to see how each request is affected by this parameter.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 1-byte field that contains the connection identifier.

**,LOCKDATA=NO LOCKDATA****,LOCKDATA=lockdata**

Use this input parameter to specify information that is to be passed to your notify exit when another user requests the LOCKINDEX lock after you have obtained the lock by using LOCKOPER=SET. This user-defined information will be passed to your notify exit when the other user issues a request specifying either one of the following:

- LOCKOPER=SET
- LOCKOPER=NOTHELD with LOCKMODE=UNCOND

**To Code:** Specify the RS-type name or address (by using a register 2 - 12 2 to ) of an 8-byte field that contains the user-defined information.

**,LOCKINDEX=lockindex**

Use this input parameter to specify the index of the lock to be operated on as specified by LOCKOPER. Note that lock indexes begin with zero.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 4-byte field that contains the lock index.

**,LOCKMODE=UNCOND****,LOCKMODE=COND**

Use this input parameter to specify how contention for the lock that is specified by LOCKINDEX should be handled if your request causes lock contention.

**UNCOND**

If the specified lock is held by another user, your request is either:

- Suspended until the lock becomes available (if MODE=SYNCSUSPEND is specified).
- Performed asynchronously with notification to you when the request completes (if any MODE value other than SYNCSUSPEND is specified).

**COND**

The lock operation is performed conditionally; that is, only if there is no contention for the lock. If the specified lock is held by another user, the request is terminated with no change to the structure, and return and reason codes describing the termination are returned to the caller.

**,LOCKOPER=SET****,LOCKOPER=RESET****,LOCKOPER=TEST****,LOCKOPER=READNEXT**

Use this input parameter to specify the type of operation to be performed on the lock specified by LOCKINDEX, or for READNEXT, beginning with the lock specified by LOCKINDEX.

**LOCKOPER=SET**

- When LOCKCOMP is not specified, your connection gets the lock, providing no other connection holds the lock. If another connection holds the lock, the request either continues once the lock is released, or fails depending on the LOCKMODE value you specify.
- When LOCKCOMP is specified, if the connection specified by LOCKCOMP holds the lock, the lock will be transferred to your connection.

**LOCKOPER=RESET**

- When LOCKCOMP is not specified, the lock will be released if it is held by your connection.
- When LOCKCOMP is specified, the lock will be released if it is held by the connection specified by LOCKCOMP.

**LOCKOPER=TEST**

- When LOCKCOMP is not specified, the lock is tested to check if it is held by your connection.
- When LOCKCOMP is also specified, the lock is tested to check if it is held by the connection specified by LOCKCOMP.



If the system returns a return code of zero, the lock was held by the specified user. If the system returns a return code of X'4' and a reason code of IXLRNCODELOCKNOTHELD, IXLRNCODELOCKCOND, or IXLRNCODELOCKHELDDBY SYS, the lock is either not held or is held by a user other than the specified user.

### **LOCKOPER=READNEXT**

This option scans the lock table, beginning with the lock index specified by LOCKINDEX, and returns to the answer area information as follows:

- When LOCKCOMP is not specified, the lock index of the next held lock, and the connection identifier of the owner of that lock, is returned.
- When LOCKCOMP is specified, the lock index of the next lock held by the LOCKCOMP connection is returned.

The request might end prematurely because it exceeded the coupling facility model-dependent time-out criteria. If this happens, the index of the next lock to be processed by the request is returned in the answer area (LAALOCKINDEX). This returned lock index can be specified for LOCKINDEX on another IXLLIST REQUEST=LOCK request to resume processing of the lock table. Except for LOCKINDEX, the same parameters and values should be specified on the resumed request as on the previous request.

**,MF=S**

**,MF=(L,mfctrl)**

**,MF=(L,mfctrl,mfattr)**

**,MF=(L,mfctrl,0D)**

**,MF=(M,mfctrl)**

**,MF=(M,mfctrl,COMPLETE)**

**,MF=(M,mfctrl,NOCHECK)**

**,MF=(E,mfctrl)**

**,MF=(E,mfctrl,COMPLETE)**

**,MF=(E,mfctrl,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

**,mfctrl**

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

**,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

**,COMPLETE**

**,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

**COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if SMILE=*var* were an optional parameter and the default is SMILE=NO\_SMILE then it would not be documented. However, if the default was SMILE=-), then it would be documented because a value would be the default.

**NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

**,MODE=SYNCSUSPEND**

**,MODE=SYNCECB**

**,MODE=SYNCEXIT**

**,MODE=SYNCTOKEN**

**,MODE=ASYNCECB**

**,MODE=ASYNCEXIT**

**,MODE=ASYNCTOKEN**

Use this input parameter to specify whether the request is to be performed synchronously or asynchronously. MODE can also be used to specify how your program can be notified of request completion.

**MODE=SYNCSUSPEND**

The request is performed synchronously. If necessary, the caller is suspended. Control is returned to the caller when the request completes. The caller must be in an enabled state to use this option.

**MODE=SYNCECB**

The request attempts synchronous processing. If the request cannot be completed synchronously, control is returned to the caller before the request completes, and the ECB specified by REQECB is posted when the request completes.

**MODE=SYNCEXIT**

The request attempts synchronous processing. If the request cannot be completed synchronously, control is returned to the caller before the request completes, and the connection's complete exit is called when the request completes.

**MODE=SYNCTOKEN**

The request attempts synchronous processing. If the request cannot be completed synchronously, control is returned to the caller before the request completes, and a token that uniquely identifies the request is returned in the area specified by REQTOKEN. The token is required to force completion of the request. See the REQTOKEN parameter description for a complete explanation of the request token.

**Note:** ANSAREA is a required parameter when MODE=SYNCTOKEN is specified.

**MODE=ASYNCECB**

Control is returned to the caller before the request completes. The ECB specified by REQECB is posted when the request completes.

**MODE=ASYNCEXIT**

Control is returned to the caller before the request completes. The connection's complete exit is called when the request completes.

**MODE=ASYNCTOKEN**

Control is returned to the caller before the request completes. A token that uniquely identifies the request is returned to the area specified by REQTOKEN. The token is required to force completion of the request. See the REQTOKEN parameter description for a complete explanation of the request token.

**Note:** ANSAREA is a required parameter when MODE=ASYNCTOKEN is specified.

**,PLISTVER=IMPLIED\_VERSION****,PLISTVER=MAX****,PLISTVER=*plistver***

Use this input parameter to specify the version of the macro. See [“Understanding IXLLIST Version Support”](#) on page 965 for a description of the options available with PLISTVER.

**,REQDATA=NO\_REQDATA****,REQDATA=*reqdata***

Use this input parameter with MODE=SYNCEXIT or MODE=ASYNCEXIT to pass any data you choose to the complete exit. The exit will get control only if the request is processed asynchronously.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the data to be passed to the complete exit.

**,REQECB=*reqecb***

Use this output parameter with either MODE=SYNCECB or MODE=ASYNCECB to specify the address of an ECB, which is to be posted when the request completes if the request was processed asynchronously.

Before coding REQECB, you must ensure that:

- You initialize the ECB before you issue the request.
- The ECB resides in either common storage or the home address space where IXLCONN was issued.
- Any tasks that wait for the ECB to be posted reside in the home address space where IXLCONN was issued.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that contains the address of the ECB to be posted when the request completes. The ECB must be aligned on a fullword boundary.

**,REQID=NO\_REQID****,REQID=*reqid***

Use this input parameter to specify a user-defined request identifier to be associated with the request. You can specify this request identifier on the IXLPURGE macro to cancel a request that has not yet been processed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the user-defined request identifier.

**,REQTOKEN=*reqtoken***

Use this output parameter with either MODE=SYNCTOKEN or MODE=ASYNCTOKEN to specify the address of a storage area to receive the request token that is returned when the request will be processed asynchronously. This token, which uniquely identifies the request, must be used as input to the IXLFComp macro, which you use to determine if the request has completed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field where the system will put the request token.

**,RETCODE=*retcode***

Use this output parameter to specify a field to contain the return code. (The return code is also returned in register 15.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the return code when the request has completed.

**,RSNCODE=*rsncode***

Use this output parameter to specify a field to contain the reason code returned, if applicable. (The reason code is also returned in register 0.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the reason code (if any) when the request has completed.

## ABEND Codes

Abend X'026' (See *z/OS MVS System Codes* for more information on this abend.)

## Return and Reason Codes

When the system returns control to your program:

- GPR 15 (and RETCODE, if specified) contains a return code.
- If the return code is not zero, GPR 0 (and RSNCODE, if specified) contains a reason code.

**Note:** The return and reason codes will also be put into the answer area (mapped by IXLYLAA), if there is one, when the request completes.

The IXLYCON macro provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

<b>0</b>	IXLRETCODEOK
<b>4</b>	IXLRETCODEWARNING
<b>8</b>	IXLRETCODEPARMERROR
<b>C</b>	IXLRETCODEENVEERROR
<b>10</b>	IXLRETCODECOMPERROR

The following table contains hexadecimal return and reason codes, the equate symbols associated with each reason code, and the meaning and suggested action for each return and reason code.

Table 61. Return and Reason Codes for IXLLIST REQUEST=LOCK Macro		
Hexadecimal Return Code	Hexadecimal Reason Cod	Equate Symbol Meaning and Action
0	None.	<p><b>Meaning:</b></p> <ul style="list-style-type: none"> <li>• If you specified a MODE value of ASYNCECB, ASYNCEXIT, ASYNCTOKEN, or ASYNCNORESPONSE, the request has been successfully initiated. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed. For any other MODE value, the request has successfully completed.</li> <li>• If LOCKOPER=TEST was specified, this return code indicates the lock is held by the connection specified by CONTOKEN.</li> </ul> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=ASYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=ASYNCEXIT, the connection's complete exit will be given control when the request has completed.</li> <li>• If you specified MODE=ASYNCTOKEN, use REQTOKEN when calling IXLFCOMP to determine when the request has completed.</li> <li>• If you specified MODE=ASYNCNORESPONSE, no action is required. You will not be notified when the request completes.</li> </ul>

Table 61. Return and Reason Codes for IXLLIST REQUEST=LOCK Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Cod	Equate Symbol Meaning and Action
4	xxxx0402	<p><b>Equate Symbol:</b> IXLRNCODEASYNCH</p> <p><b>Meaning:</b> The request could not be processed synchronously. It will be processed asynchronously. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed.</p> <ul style="list-style-type: none"> <li>If MODE=SYNCECB was specified, the ECB specified for REQECB will be posted when the request has finished.</li> <li>If MODE=SYNCEXIT was specified, the connection's complete exit will be given control when the request has finished.</li> <li>If MODE=SYNCTOKEN was specified, an asynchronous request token was returned in the area specified by REQTOKEN. This request token may be specified on the IXLFCOMP macro to determine when the request has finished.</li> </ul> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>If you specified MODE=SYNCECB, wait on the ECB you specified in REQECB.</li> <li>If you specified MODE=SYNCEXIT, the connection's complete exit will be given control when the request has completed.</li> <li>If you specified MODE=SYNCTOKEN, use REQTOKEN when calling IXLFCOMP to determine when the request has completed.</li> </ul>
4	xxxx0409	<p><b>Equate Symbol:</b> IXLRNCODETIMEOUT</p> <p><b>Meaning:</b> The request specifying LOCKOPER=READNEXT completed prematurely because it exceeded the coupling facility model-dependent time-out criteria. The index of the next lock to be processed has been returned in the answer area (field LAALOCKINDEX).</p> <p><b>Action:</b> Reissue the request specifying for LOCKINDEX the index of the next lock to be processed. For more information about premature completion, see <a href="#">z/OS MVS Programming: Sysplex Services Guide</a>.</p>
4	xxxx040E	<p><b>Equate Symbol:</b> IXLRNCODELOCKNOTHELD</p> <p><b>Meaning:</b> A LOCKOPER=TEST request determined that the specified lock was not held for the specified connection. The connection ID of the lock owner is returned in the answer area (field LAACONID).</p> <p><b>Action:</b> None necessary. If this reason code is not expected, determine who holds the lock and see if any clean-up is necessary. The lock may need to be released, or you can wait and try again.</p>
4	xxxx0410	<p><b>Equate Symbol:</b> IXLRNCODELOCKCOND</p> <p><b>Meaning:</b> For a LOCKMODE=COND request, or a request that specified LOCKCOMP, the request could not be completed successfully because the specified lock is not currently held as required. The connection identifier of the lock owner is returned in the answer area (LAACONID field).</p> <p><b>Action:</b> Retry the request, or obtain the lock as required, and retry the request. If you are unable to get the lock, check the ID in the LAACONID field and determine if some recovery is necessary.</p>
4	xxxx0412	<p><b>Equate Symbol:</b> IXLRNCODELOCKHELDDBYSYS</p> <p><b>Meaning:</b> The lock is not held by any connection, but instead is held by the system. For a request that specified LOCKMODE=COND or LOCKCOMP, the request could not be completed successfully because the specified lock is not currently held as required. For a request that specified LOCKOPER=READNEXT, the request found a lock that is not generally available (see field LAALOCKINDEX in the answer area).</p> <p><b>Action:</b> Retry the request, or obtain the lock as required, and retry the request.</p>
4	xxxx041F	<p><b>Equate Symbol:</b> IXLRNCODENOLOCKSHELD</p> <p><b>Meaning:</b> A request specifying LOCKOPER=READNEXT found no locks held from the LOCKINDEX lock to the end of the lock table.</p> <p><b>Action:</b> None necessary.</p>

Table 61. Return and Reason Codes for IXLLIST REQUEST=LOCK Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Cod	Equate Symbol Meaning and Action
8	xxxx0801	<p><b>Equate Symbol:</b> IXLRNCODEBADPARMLIST</p> <p><b>Meaning:</b> Program error. The parameter list for this request is not addressable. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Verify that:</p> <ul style="list-style-type: none"> <li>• The parameter list address is uncorrupted.</li> <li>• The parameter list is addressable in the caller's primary address space.</li> <li>• If you are issuing this macro while disabled, the parameter list resides in either page-fixed or DREF storage.</li> <li>• If you are issuing this macro in AR-mode and you specified the parameter list address using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If you are issuing this macro in AR-mode, SYSSTATE ASCENV=AR must be issued before issuing this macro.</li> </ul>
8	xxxx0804	<p><b>Equate Symbol:</b> IXLRNCODEBADVERSIONNUM (or IXLRNCODEBADVERSION#)</p> <p><b>Meaning:</b> The version number in the macro parameter list is not compatible with the level of XES currently being used. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that your program did not overlay the parameter list storage.</li> <li>• Verify that your program was assembled with the correct macro library for the release of MVS on which your program is running.</li> </ul>
8	xxxx080A	<p><b>Equate Symbol:</b> IXLRNCODEBADCONTOKEN</p> <p><b>Meaning:</b> Program error. The specified connect token (CONTOKEN) was not valid for one of the following reasons:</p> <ol style="list-style-type: none"> <li>1. The user with the connection identifier represented by the token has disconnected from the structure.</li> <li>2. The connector's task (the task that issued IXLCONN) ended.</li> <li>3. The specified token is not the token that was returned from IXLCONN.</li> <li>4. The request was issued from an address space other than the address space in which IXLCONN was issued.</li> <li>5. The connect token was invalidated during rebuild.</li> <li>6. The connect token was invalidated by XES.</li> </ol> <p><b>Note:</b> The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Take the action with the corresponding meaning.</p> <ol style="list-style-type: none"> <li>1. Discontinue use of the structure. Perform recovery and cleanup for the structure and check the protocol for the use of the structure to determine why you did not know of the disconnection.</li> <li>2. Discontinue use of the structure. Perform recovery and cleanup for the structure and check the protocol for the use of the structure to determine why you did not know that the task ended.</li> <li>3. The contoken is returned in the CONACONTOKEN field of the answer area specified on the IXLCONN request.</li> <li>4. Issue your request from the same address space the IXLCONN was issued in.</li> <li>5. Wait for the rebuild to complete, and try again.</li> <li>6. Discontinue use of the structure. Perform recovery and cleanup for the structure.</li> </ol>

Table 61. Return and Reason Codes for IXLLIST REQUEST=LOCK Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Cod	Equate Symbol Meaning and Action
8	xxxx0824	<p><b>Equate Symbol:</b> IXLRNCODEWRONGSTRTYPE</p> <p><b>Meaning:</b> Program error. The connection specified by CONTOKEN is not to a list structure.</p> <p><b>Action:</b> Specify the appropriate CONTOKEN value, and rerun your program. The CONTOKEN is returned in the answer area provided on the IXLCONN invocation. The type of structure built is passed on the TYPE parameter of the IXLCONN macro.</p>
8	xxxx0838	<p><b>Equate Symbol:</b> IXLRNCODEBADANSAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by ANSAREA is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The answer area address specified by ANSAREA is valid.</li> <li>• If the caller is running in AR-mode and the ANSAREA parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• The address specified for ANSAREA is addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are calling IXLLIST while disabled, ANSAREA must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLLIST in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLIST macro.</li> </ul>
8	xxxx0839	<p><b>Equate Symbol:</b> IXLRNCODEBADREQTOKENAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by REQTOKEN is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The request token area specified by REQTOKEN is valid.</li> <li>• If the caller is running in AR-mode and the REQTOKEN parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If you are calling IXLLIST while disabled, REQTOKEN must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLLIST in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLIST macro.</li> </ul>
8	xxxx083D	<p><b>Equate Symbol:</b> IXLRNCODEBADANSLEN</p> <p><b>Meaning:</b> Program error. The length of the answer area, as specified by ANSLEN, is not sufficient to contain answer area information. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Increase the size of the answer area, and rerun your program. The minimum size of the ANSAREA is in the LAA_LEN field of the IXLYLAA mapping macro.</p>
8	xxxx0842	<p><b>Equate Symbol:</b> IXLRNCODEPERSISTENTLOCK</p> <p><b>Meaning:</b> Program error. The request specifying LOCKOPER=SET with LOCKMODE=UNCOND failed because the lock is held by a connection that is in the failed-persistent state. The connection identifier of the lock owner is returned in the answer area (field LAACONID).</p> <p><b>Action:</b> Either perform recovery for the connection, or wait until recovery for the connection is performed.</p>

Table 61. Return and Reason Codes for IXLLIST REQUEST=LOCK Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Cod	Equate Symbol Meaning and Action
8	xxxx0846	<p><b>Equate Symbol:</b> IXLRNCODEBADLOCKINDEX</p> <p><b>Meaning:</b> Program error. The specified LOCKINDEX exceeds the size of the lock table for the structure.</p> <p><b>Action:</b> Correct LOCKINDEX to specify an index that is contained within the lock table. The maximum value for the LOCKINDEX is one less than the value specified by the LOCKENTRIES parameter on the IXLCONN macro.</p>
8	xxxx0848	<p><b>Equate Symbol:</b> IXLRNCODEBADRESET</p> <p><b>Meaning:</b> Program error. LOCKOPER=RESET was specified for a lock not currently held by the invoker. The value of the connection ID holding the lock is returned in the answer area (field LAACONID).</p> <p><b>Action:</b> Check your code to ensure that your connection did not already reset the lock.</p>
8	xxxx084B	<p><b>Equate Symbol:</b> IXLRNCODENOLOCKS</p> <p><b>Meaning:</b> Program error. A locking operation was requested, but the structure does not contain a lock table.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• You are connected to the intended structure.</li> <li>• You intended to perform a locking operation.</li> </ul> <p>The use of locks is determined by the LOCKENTRIES keyword on the IXLCONN macro.</p>
8	xxxx0851	<p><b>Equate Symbol:</b> IXLRNCODENOSUSPENDISABLE</p> <p><b>Meaning:</b> Program error. The request failed because MODE=SYNCSUSPEND was specified, but the caller is disabled and cannot be suspended.</p> <p><b>Action:</b> Either specify another MODE value, or become enabled (release the CPU lock), then reissue the request.</p>
8	xxxx0854	<p><b>Equate Symbol:</b> IXLRNCODEBADLOCKCOMP</p> <p><b>Meaning:</b> Program error. The connection identifier specified for LOCKCOMP is not valid.</p> <p><b>Action:</b> The connection identifier is returned in the answer area (mapped by IXLYCONA) when the IXLCONN macro was issued.</p>
C	xxxx0C06	<p><b>Equate Symbol:</b> IXLRNCODENOCONN</p> <p><b>Meaning:</b> Environmental error. This system does not have connectivity to the coupling facility that contains the list structure. Possible reasons for this are:</p> <ul style="list-style-type: none"> <li>• The operator issued VARY PATH,OFFLINE.</li> <li>• The operator issued CONFIG CHP,OFFLINE.</li> <li>• Hardware errors to the coupling facility.</li> <li>• Facility or path failure to the coupling facility.</li> </ul> <p><b>Action:</b> Begin rebuilding the structure on a different coupling facility, or disconnect from the structure.</p>



Table 61. Return and Reason Codes for IXLLIST REQUEST=LOCK Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Cod	Equate Symbol Meaning and Action
C	xxxx0C13	<p><b>Equate Symbol:</b> IXLRNCCODEREQPURGED</p> <p><b>Meaning:</b> Environmental error. The request was purged prior to completion of the request. Possible reasons include:</p> <ul style="list-style-type: none"> <li>• The connector failed.</li> <li>• The connector disconnected.</li> <li>• The requestor failed.</li> <li>• The request was purged by IXLPURGE.</li> <li>• Requests were purged when the connector provided an IXLEERSP response for the Rebuild Stop or Rebuild Cleanup event.</li> <li>• The secondary address space was no longer valid.</li> </ul> <p><b>Action:</b> None if this is expected. Otherwise, determine why the connector failed.</p>
C	xxxx0C14	<p><b>Equate Symbol:</b> IXLRNCCODESTATUSUNKNOWN</p> <p><b>Meaning:</b> Environmental error. The request has completed, but the final disposition of the request cannot be determined. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the validity of your data by comparing the expected results with what is in the coupling facility.</li> <li>• Consider running rebuild processing to move the structure to attempt to protect it from further errors of this kind.</li> </ul>
C	xxxx0C25	<p><b>Equate Symbol:</b> IXLRNCCODESTRFAILURE</p> <p><b>Meaning:</b> Environmental error. The list structure failed prior to completion of the request.</p> <p><b>Action:</b> Either rebuild or disconnect from the structure.</p>
C	xxxx0CA0	<p><b>Equate Symbol:</b> IXLRNCCODEQUIESCEDSUSPENDFAIL</p> <p><b>Meaning:</b> Environmental error. The request is failed because the structure is quiesced for a system-managed process and SUSPEND=FAIL is specified on the IXLCONN.</p> <p><b>Action:</b> None, if this is expected.</p>
C	xxxxFFFF	<p><b>Equate Symbol:</b> IXLRNCCODENOTAVAILABLE</p> <p><b>Meaning:</b> Environmental error. XES functions are not available. This could occur because the coupling facility hardware necessary to provide XES functions is not present.</p> <p><b>Action:</b> Re-IPL the system, or follow your particular failure management protocol.</p>
10	xxxx10xx	<p><b>Meaning:</b> System error. XES processing failure. The state of the involved structure and the disposition of the request are unpredictable. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Contact the IBM support center.</p>



---

## Chapter 62. IXLLIST REQUEST=MONITOR\_EVENTQ

---

### Description

A MONITOR\_EVENTQ request allows you to start (ACTION=START) or stop (ACTION=STOP) monitoring your event queue for the presence of event monitor controls (EMCs). The system queues or withdraws event monitor controls on your event queue when sublist monitoring is in effect. The EMCs indicate the empty or nonempty state of the sublist(s) that you are monitoring.

When starting event queue monitoring, you must specify the list notification vector index (VECTORINDEX) to be associated with the event queue. List services uses the specified vector index to indicate whether the event queue is in the **empty** or **non-empty** state.

If you want to be notified when the event queue for which monitoring has been activated has changed state from empty to non-empty, specify DRIVEEXIT=YES. This option causes your list transition exit to be driven, informing you that your monitored event queue now has at least one entry on it. You must identify your list transition exit on the LISTTRANEXIT parameter of the IXLCONN macro. You can use the IXLVECTR macro to test the vector entry that you have associated with your monitored event queue.

A MONITOR\_EVENTQ request can be issued only for keyed list structures allocated in a coupling facility of CFLEVEL=3 or higher. MONITOR\_EVENTQ requests issued for a list structure allocated in a coupling facility of CFLEVEL 0, 1, or 2 will fail.

If you have started event queue monitoring with a certain vector index specified and you want to monitor the event queue with a different vector index, do not stop monitoring the event queue with the old vector index. Simply start monitoring the event queue with the new index; the new index will automatically replace the old.

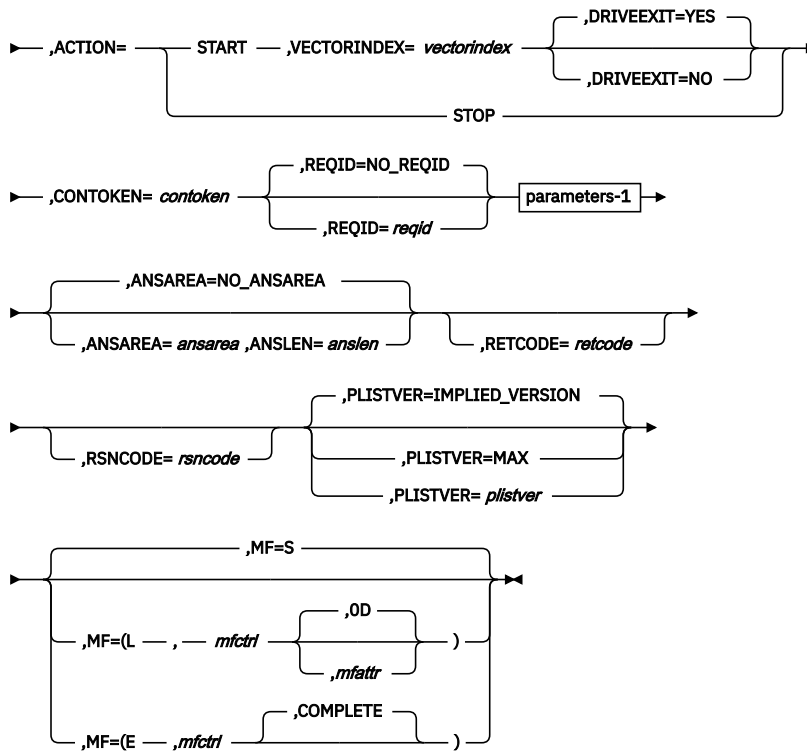
---

### Syntax Diagram

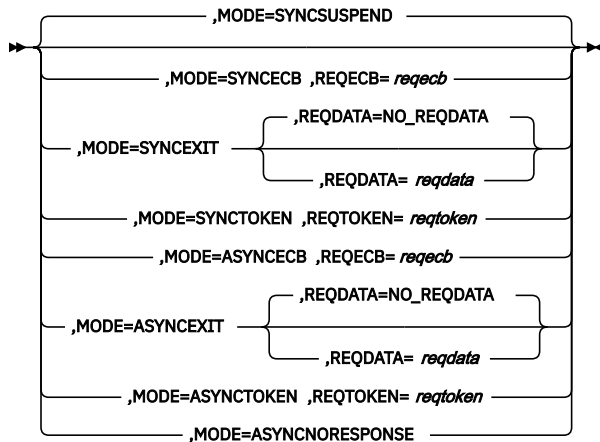
The syntax diagram for IXLLIST REQUEST=MONITOR\_EVENTQ is as follows:

## main diagram

➔ IXLLIST — REQUEST=MONITOR\_EVENTQ ➔



## parameters-1



**Note:** If MODE=SYNCTOKEN or MODE=ASYNCTOKEN is specified, ANSAREA= *ansarea* and ANSLEN= *anslen* are required.

## Parameter Descriptions

The parameter descriptions for REQUEST=MONITOR\_EVENTQ are listed in alphabetical order. Default values are underlined:

### REQUEST=MONITOR\_EVENTQ

Use this input parameter to specify that monitoring of the event queue associated with this user be started or stopped.

**,ACTION=START****,ACTION=STOP**

Use this input parameter to specify whether event queue monitoring is to be started or stopped.

**START**

Start monitoring the event queue or restart monitoring with different parameters.

**STOP**

Stop monitoring the event queue.

**,ANSAREA=NO\_ANSAREA****,ANSAREA=ansarea**

Use this output parameter to specify an answer area to contain information returned from the request. The format of the answer area is described by mapping macro IXLYLAA. Upon successful completion of a request with ACTION=START, the answer area contains the number of event monitor controls (EMCs) queued to your event queue when monitoring was established (field LAAMNEQ\_EVENTCNT) and the state (empty or nonempty) of the event queue (field LAAMNEQ\_EVENTQUEUED).

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an area (with a length of ANSLEN) to contain the information.

**,ANSLEN=anslen**

Use this input parameter to specify the size of the storage area specified by ANSAREA.

Check the prologue of the IXLYLAA mapping macro for the minimum required size of the answer area, or use the length field (LAA\_LEN) in the LAA.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 2-byte field that contains the size, in bytes, of the answer area.

**,CONTOKEN=contoken**

Use this input parameter to specify the connect token that was returned by the IXLCONN service. The connect token uniquely identifies your connection to the list structure, and must be specified on each IXLLIST invocation.

The connect token is available in the IXLCONN answer area that is mapped by IXLYCONA.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 16-byte field that contains the connect token.

**,DRIVEEXIT=YES****,DRIVEEXIT=NO**

Use this input parameter to specify whether list services should drive the connection's list transition exit when the state of the event queue changes from empty to nonempty. The list transition exit informs the connection that the event queue being monitored has changed state. Use the IXLVECTR macro to test the vector entry that you have associated with your monitored event queue.

**YES**

The connection's list transition exit will be called when its monitored event queue changes state from empty to non-empty.

**NO**

The connection's list transition exit will not be called when its monitored event queue changes state. However, the vector index associated with the event queue will be updated to reflect the current empty/non-empty state information.

**,MF=S**  
**,MF=(L,mfctrl)**  
**,MF=(L,mfctrl,mfattr)**  
**,MF=(L,mfctrl,0D)**  
**,MF=(M,mfctrl)**  
**,MF=(M,mfctrl,COMPLETE)**  
**,MF=(M,mfctrl,NOCHECK)**  
**,MF=(E,mfctrl)**  
**,MF=(E,mfctrl,COMPLETE)**  
**,MF=(E,mfctrl,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

#### **,mfctrl**

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

#### **,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

#### **,COMPLETE**

#### **,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

#### **COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if SMILE=*var* were an optional parameter and the default is SMILE=NO\_SMILE then it would not be documented. However, if the default was SMILE=:), then it would be documented because a value would be the default.

#### **NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

**,MODE=SYNCSUSPEND**  
**,MODE=SYNCECB**  
**,MODE=SYNCEXIT**  
**,MODE=SYNCTOKEN**  
**,MODE=ASYNCECB**  
**,MODE=ASYNCEXIT**  
**,MODE=ASYNCTOKEN**  
**,MODE=ASYNCSNORESPONSE**

Use this input parameter to specify:

- Whether the request is to be performed synchronously or asynchronously
- How you want to be notified of request completion

**MODE=SYNCSUSPEND**

The request is suspended until it can complete synchronously. To use this option, your program must be enabled for I/O and external interrupts.

**MODE=SYNCECB**

The request processes synchronously if possible. If the request processes asynchronously, the ECB specified by REQECB is posted when the request completes.

**MODE=SYNCEXIT**

The request processes synchronously if possible. If the request processes asynchronously, your complete exit is given control when the request completes.

**MODE=SYNCTOKEN**

The request processes synchronously if possible. If the request processes asynchronously, an asynchronous request token is returned in the area specified by REQTOKEN on invocation of the request. Use the returned request token on the IXLFComp macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=SYNCTOKEN is specified.

**MODE=ASYNCECB**

The request processes asynchronously. The ECB specified by REQECB is posted when the request completes.

**MODE=ASYNCEXIT**

The request processes asynchronously. Your complete exit is given control when the request completes.

**MODE=ASYNCTOKEN**

The request processes asynchronously. An asynchronous request token is returned in the area specified by REQTOKEN upon invocation of the request. Use the returned request token on the IXLFComp macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=ASYNCTOKEN is specified.

**MODE=ASYNCSNORESPONSE**

The request processes asynchronously. No notification of request completion is provided. The answer area (ANSAREA) fields will not contain valid information when this mode is specified.

**Note:** You cannot code MODE=ASYNCSNORESPONSE with LOCKINDEX, BUFFER, or BUFLIST.

**,PLISTVER=IMPLIED\_VERSION**  
**,PLISTVER=MAX**  
**,PLISTVER=plistver**

Use this input parameter to specify the version of the macro. See [“Understanding IXLLIST Version Support”](#) on page 965 for a description of the options available with PLISTVER.

**,REQDATA=NO\_REQDATA**  
**,REQDATA=reqdata**

Use this input parameter with MODE=SYNCEXIT or MODE=ASYNCEXIT to pass any data you choose to the complete exit. The exit will get control only if the request is processed asynchronously.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the data to be passed to the complete exit.

**,REQECB=reqecb**

Use this output parameter with either MODE=SYNCECB or MODE=ASYNCECB to specify the address of an ECB, which is to be posted when the request completes if the request was processed asynchronously.

Before coding REQECB, you must ensure that:

- You initialize the ECB before you issue the request.
- The ECB resides in either common storage or the home address space where IXLCONN was issued.
- Any tasks that wait for the ECB to be posted reside in the home address space where IXLCONN was issued.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that contains the address of the ECB to be posted when the request completes. The ECB must be aligned on a fullword boundary.

**,REQID=NO\_REQID**

**,REQID=reqid**

Use this input parameter to specify a user-defined request identifier to be associated with the request. You can specify this request identifier on the IXPURGE macro to cancel a request that has not yet been processed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the user-defined request identifier.

**,REQTOKEN=reqtoken**

Use this output parameter with either MODE=SYNCTOKEN or MODE=ASYNCTOKEN to specify the address of a storage area to receive the request token that is returned when the request will be processed asynchronously. This token, which uniquely identifies the request, must be used as input to the IXLFCOMP macro, which you use to determine if the request has completed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field where the system will put the request token.

**,RETCODE=retcode**

Use this output parameter to specify a field to contain the return code. (The return code is also returned in register 15.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the return code when the request has completed.

**,RSNCODE=rsncode**

Use this output parameter to specify a field to contain the reason code returned, if applicable. (The reason code is also returned in register 0.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the reason code (if any) when the request has completed.

**,VECTORINDEX=vectorindex**

Use this input parameter to specify the list notification vector index to be associated with the monitored event queue. If a start request completes successfully, the index identified by this vector index will reflect the event queue's state—either empty or non-empty. The timing as to when the index reflects the event queue's state is not always immediate; however, the index will eventually reflect the correct state of the event queue.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword field that contains the list notification vector index.

## ABEND Codes

Abend X'026' (See [z/OS MVS System Codes](#) for more information on this abend.)



## Return and Reason Codes

When the system returns control to your program:

- GPR 15 (and RETCODE, if specified) contains a return code.
- GPR 0 (and RSNCODE, if specified) contains a reason code, if applicable.

**Note:** The return and reason codes will also be put into the answer area (mapped by IXLYLAA), if there is one, when the request completes.

The IXLYCON macro provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

<b>0</b>	IXLRETCODEOK
<b>4</b>	IXLRETCODEWARNING
<b>8</b>	IXLRETCODEPARMERROR
<b>C</b>	IXLRETCODEENVERROR
<b>10</b>	IXLRETCODECOMPERROR

The following table contains hexadecimal return and reason codes, the equate symbols associated with each reason code, and the meaning and suggested action for each return and reason code.

Table 62. Return and Reason Codes for IXLLIST REQUEST=MONITOR_EVENTQ Macro		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0	None.	<p><b>Meaning:</b> If you specified a MODE value of ASYNCECB, ASYNCEXIT, ASYNCTOKEN, or ASYNCNORESPONSE, the request has been successfully initiated. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed. For any other MODE value, the request has successfully completed.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=ASYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=ASYNCEXIT, the connection's complete exit will be given control when the request has completed.</li> <li>• If you specified MODE=ASYNCTOKEN, use REQTOKEN when calling IXLFComp to determine when the request has completed.</li> </ul>

Table 62. Return and Reason Codes for IXLLIST REQUEST=MONITOR\_EVENTQ Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	xxxx0402	<p><b>Equate Symbol:</b> IXLRNCODEASYNCH</p> <p><b>Meaning:</b> The request could not be processed synchronously. It will be processed asynchronously. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed.</p> <ul style="list-style-type: none"> <li>• If MODE=SYNCECB was specified, the ECB specified for REQECB will be posted when the request has finished.</li> <li>• If MODE=SYNCEXIT was specified, the connection's complete exit will be given control when the request has finished.</li> <li>• If MODE=SYNCTOKEN was specified, an asynchronous request token was returned in the area specified by REQTOKEN. This request token may be specified on the IXLFCOMP macro to determine when the request has finished.</li> </ul> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=SYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=SYNCEXIT, the connection's complete exit will be given control when the request has completed.</li> <li>• If you specified MODE=SYNCTOKEN, use REQTOKEN when calling IXLFCOMP to determine when the request has completed.</li> </ul>
8	xxxx0801	<p><b>Equate Symbol:</b> IXLRNCODEBADPARMLIST</p> <p><b>Meaning:</b> Program error. The parameter list for this request is not addressable. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Verify that:</p> <ul style="list-style-type: none"> <li>• The parameter list address is uncorrupted.</li> <li>• The parameter list is addressable in the caller's primary address space.</li> <li>• If you are issuing this macro while disabled, the parameter list resides in either page-fixed or DREF storage.</li> <li>• If you are issuing this macro in AR-mode and you specified the parameter list address using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If you are issuing this macro in AR-mode, SYSSTATE ASCENV=AR must be issued before issuing this macro.</li> </ul>

Table 62. Return and Reason Codes for IXLLIST REQUEST=MONITOR\_EVENTQ Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0804	<p><b>Equate Symbol:</b> IXLRNSCODEBADVERSIONNUM (or IXLRNSCODEBADVERSION#)</p> <p><b>Meaning:</b> The version number in the macro parameter list is not compatible with the level of XES currently being used. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that your program did not overlay the parameter list storage.</li> <li>• Verify that your program was assembled with the correct macro library for the release of MVS on which your program is running.</li> </ul>
8	xxxx080A	<p><b>Equate Symbol:</b> IXLRNSCODEBADCONTOKEN</p> <p><b>Meaning:</b> Program error. The specified connect token (CONTOKEN) was not valid for one of the following reasons:</p> <ol style="list-style-type: none"> <li>1. The user with the connection identifier represented by the token has disconnected from the structure.</li> <li>2. The connector's task (the task that issued IXLCONN) ended.</li> <li>3. The specified token is not the token that was returned from IXLCONN.</li> <li>4. The request was issued from an address space other than the address space in which IXLCONN was issued.</li> <li>5. The connect token was invalidated during rebuild.</li> <li>6. The connect token was invalidated by XES.</li> </ol> <p><b>Note:</b> The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Take the action with the corresponding meaning.</p> <ol style="list-style-type: none"> <li>1. Discontinue use of the structure. Perform recovery and cleanup for the structure and check the protocol for the use of the structure to determine why you did not know of the disconnection.</li> <li>2. Discontinue use of the structure. Perform recovery and cleanup for the structure and check the protocol for the use of the structure to determine why you did not know that the task ended.</li> <li>3. The contoken is returned in the CONACONTOKEN field of the answer area specified on the IXLCONN request.</li> <li>4. Issue your request from the same address space the IXLCONN was issued in.</li> <li>5. Wait for the rebuild to complete, and try again.</li> <li>6. Discontinue use of the structure. Perform recovery and cleanup for the structure.</li> </ol>

Table 62. Return and Reason Codes for IXLLIST REQUEST=MONITOR\_EVENTQ Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0824	<p><b>Equate Symbol:</b> IXLRSNCODEWRONGSTRTYPE</p> <p><b>Meaning:</b> Program error. The connection specified by CONTOKEN is not to a list structure.</p> <p><b>Action:</b> Specify the appropriate CONTOKEN value, and rerun your program. The CONTOKEN is returned in the answer area provided on the IXLCONN invocation. The type of structure built is passed on the TYPE parameter of the IXLCONN macro.</p>
8	xxxx0838	<p><b>Equate Symbol:</b> IXLRSNCODEBADANSAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by ANSAREA is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The answer area address specified by ANSAREA is valid.</li> <li>• If the caller is running in AR-mode and the ANSAREA parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• The address specified for ANSAREA is addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are calling IXLLIST while disabled, ANSAREA must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLLIST in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLIST macro.</li> </ul>
8	xxxx0839	<p><b>Equate Symbol:</b> IXLRSNCODEBADREQTOKENAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by REQTOKEN is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The request token area specified by REQTOKEN is valid.</li> <li>• If the caller is running in AR-mode and the REQTOKEN parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If you are calling IXLLIST while disabled, REQTOKEN must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLLIST in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLIST macro.</li> </ul>
8	xxxx083D	<p><b>Equate Symbol:</b> IXLRSNCODEBADANSLEN</p> <p><b>Meaning:</b> Program error. The length of the answer area, as specified by ANSLEN, is not sufficient to contain answer area information. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Increase the size of the answer area, and rerun your program. The minimum size of the ANSAREA is in the LAA_LEN field of the IXLYLAA mapping macro.</p>

Table 62. Return and Reason Codes for IXLLIST REQUEST=MONITOR\_EVENTQ Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx084A	<p><b>Equate Symbol:</b> IXLRSNCODENOKEYS</p> <p><b>Meaning:</b> Program error. The structure does not support the use of entry keys. The IXLLIST request type either required the structure to support entry keys or designated a list entry or list position by list number and entry key.</p> <p><b>Action:</b> Ensure that you are connected to the intended structure. The use of keys for a structure is determined by the REFOPTION keyword on the IXLCONN macro.</p>
8	xxxx0851	<p><b>Equate Symbol:</b> IXLRSNCODENOSUSPENDISABLE</p> <p><b>Meaning:</b> Program error. The request failed because MODE=SYNCSUSPEND was specified, but the caller is disabled and cannot be suspended.</p> <p><b>Action:</b> Either specify another MODE value, or become enabled (release the CPU lock), then reissue the request.</p>
8	xxxx0852	<p><b>Equate Symbol:</b> IXLRSNCODENOLISTVECTOR</p> <p><b>Meaning:</b> Program error. The request failed because no local vector for monitoring lists or the user's event queue exists for this connection. Either a list notification vector for monitoring lists or the user's event queue was not requested for this connection or the list notification vector has been deleted.</p> <p><b>Action:</b> Either you are not connected to this structure or the VECTORLEN parameter was not specified when the IXLCONN was done for this structure.</p>
8	xxxx0853	<p><b>Equate Symbol:</b> IXLRSNCODEINVLISTVINDEX</p> <p><b>Meaning:</b> Program error. The list notification vector index (VECTORINDEX) specified with ACTION=START was not valid. This might be because the vector index you specified is greater than the number of vector entries in the list notification vector.</p> <p><b>Action:</b> Correct the VECTORINDEX that you are using.</p>
C	xxxx0C06	<p><b>Equate Symbol:</b> IXLRSNCODENOCOONN</p> <p><b>Meaning:</b> Environmental error. This system does not have connectivity to the coupling facility that contains the list structure. Possible reasons for this are:</p> <ul style="list-style-type: none"> <li>• The operator issued VARY PATH,OFFLINE.</li> <li>• The operator issued CONFIG CHP,OFFLINE.</li> <li>• Hardware errors to the coupling facility.</li> <li>• Facility or path failure to the coupling facility.</li> </ul> <p><b>Action:</b> Begin rebuilding the structure on a different coupling facility, or disconnect from the structure.</p>

Table 62. Return and Reason Codes for IXLLIST REQUEST=MONITOR\_EVENTQ Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C13	<p><b>Equate Symbol:</b> IXLRSNCODEREQPURGED</p> <p><b>Meaning:</b> Environmental error. The request was purged prior to completion of the request. Possible reasons include:</p> <ul style="list-style-type: none"> <li>• The connector failed.</li> <li>• The connector disconnected.</li> <li>• The requestor failed.</li> <li>• The request was purged by IXLPURGE.</li> <li>• Requests were purged when the connector provided an IXLEERSP response for the Rebuild Stop or Rebuild Cleanup event.</li> <li>• The secondary address space was no longer valid.</li> </ul> <p><b>Action:</b> None if this is expected. Otherwise, determine why the connector failed.</p>
C	xxxx0C14	<p><b>Equate Symbol:</b> IXLRSNCODESTATUSUNKNOWN</p> <p><b>Meaning:</b> Environmental error. The request has completed, but the final disposition of the request cannot be determined. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the validity of your data by comparing the expected results with what is in the coupling facility.</li> <li>• Consider running rebuild processing to move the structure to attempt to protect it from further errors of this kind.</li> </ul>
C	xxxx0C25	<p><b>Equate Symbol:</b> IXLRSNCODESTRFAILURE</p> <p><b>Meaning:</b> Environmental error. The list structure failed prior to completion of the request.</p> <p><b>Action:</b> Either rebuild or disconnect from the structure.</p>
C	xxxx0C68	<p><b>Equate Symbol:</b> IXLRSNCODEBADREQCFLEVEL</p> <p><b>Meaning:</b> Environmental error. The request type is not permitted for the level of coupling facility in which the target structure is allocated.</p> <p><b>Action:</b> Either disconnect from the structure (using IXLDISC) or initiate a rebuild of the structure, if allowed (using IXLREBLD) in a coupling facility of the correct CFLEVEL.</p>
C	xxxx0CA0	<p><b>Equate Symbol:</b> IXLRSNCODEQUIESCEDSUSPENDFAIL</p> <p><b>Meaning:</b> Environmental error. The request is failed because the structure is quiesced for a system-managed process and SUSPEND=FAIL is specified on the IXLCONN.</p> <p><b>Action:</b> None, if this is expected.</p>

Table 62. Return and Reason Codes for IXLLIST REQUEST=MONITOR\_EVENTQ Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxxFFFF	<b>Equate Symbol:</b> IXLRSNCODENOTAVAILABLE <b>Meaning:</b> Environmental error. XES functions are not available. This could occur because the coupling facility hardware necessary to provide XES functions is not present. <b>Action:</b> Re-IPL the system, or follow your particular failure management protocol.
10	xxxx10xx	<b>Meaning:</b> System error. XES processing failure. The state of the involved structure and the disposition of the request are unpredictable. The answer area (ANSAREA) fields do not contain valid information. <b>Action:</b> Contact the IBM support center.





---

## Chapter 63. IXLLIST REQUEST=MONITOR\_LIST

---

### Description

A MONITOR\_LIST request allows you to start (ACTION=START) or stop (ACTION=STOP) monitoring a specified list (LISTNUM) for the presence of entries. When starting list monitoring, you must specify the list notification vector index (VECTORINDEX) to be associated with the LISTNUM list. List services uses the specified vector index to indicate whether the list is in the **empty** or **non-empty** state.

If you want to be notified when one of the lists for which list monitoring has been activated has changed state from empty to non-empty, you can specify DRIVEEXIT=YES. This option causes your list transition exit to be driven, informing you that a list that you monitor now has at least one entry on it. Your list transition exit must be identified on the LISTTRANEXIT parameter of the IXLCONN macro. To determine which of the monitored lists has changed state, you must use the IXLVECTR macro.

There are some considerations involving when you must stop list monitoring. If you have started list monitoring for a certain list with a certain vector index specified and you want to:

**Monitor a different list with the same vector index**

You must stop list monitoring of the old list with the index before you start list monitoring of the new list with that index.

**Monitor the same list with a different vector index**

Do not stop monitoring the list with the old vector index. Simply start monitoring the list with the new index; the new index will automatically replace the old.

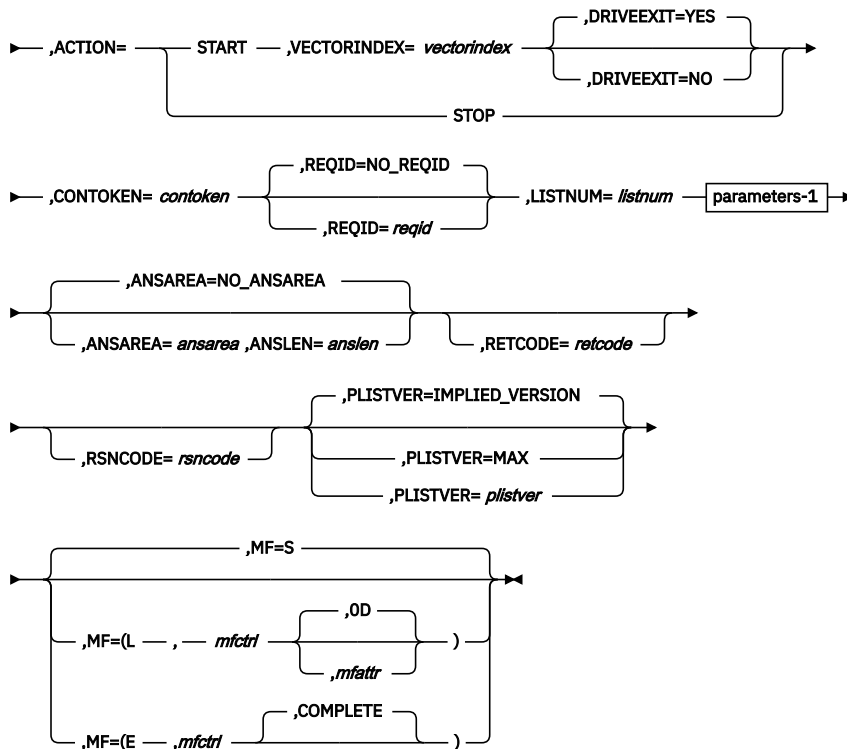
---

### Syntax Diagram

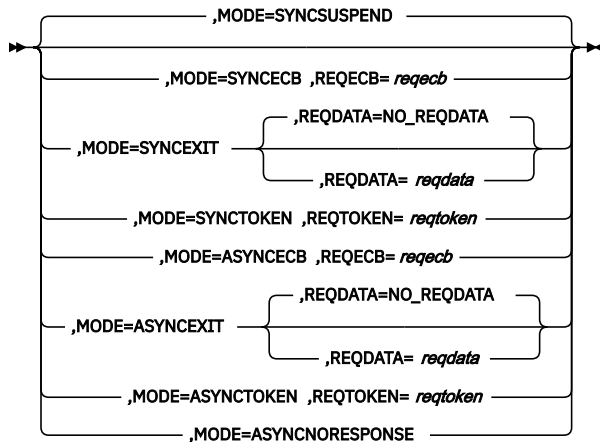
The syntax diagram for IXLLIST REQUEST=MONITOR\_LIST is as follows:

## main diagram

IXLLIST — REQUEST=MONITOR\_LIST →



## parameters-1



## Note:

1. If MODE=SYNCTOKEN or MODE=ASYNCTOKEN is specified, ANSAREA=*ansarea* and ANSLEN=*anslen* are required.
2. If MODE=ASYNCNORESPONSE is specified, you may not specify ACTION=START.

## Parameter Descriptions

The parameter descriptions for REQUEST=MONITOR\_LIST are listed in alphabetical order. Default values are underlined:

**REQUEST=MONITOR\_LIST**

Use this input parameter to specify that monitoring of the list specified by LISTNUM be started or stopped.

**,ACTION=START****,ACTION=STOP**

Use this input parameter to specify whether list monitoring is to be started or stopped.

**START**

Monitoring of the LISTNUM list is started.

**STOP**

Monitoring of the LISTNUM list is stopped.

**Note:** You cannot code ACTION=START with MODE=ASYNCHRESPONSE.

**,ANSAREA=NO\_ANSAREA****,ANSAREA=ansarea**

Use this output parameter to specify an answer area to contain information returned from the request. The format of the answer area is described by mapping macro IXLYLAA. Upon successful completion of a request with ACTION=START, the answer area contains the number of allocated entries or elements on the processed list (field LAALISTCNT).

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an area (with a length of ANSLEN) to contain the information.

**,ANSLEN=anslen**

Use this input parameter to specify the size of the storage area specified by ANSAREA.

Check the prologue of the IXLYLAA mapping macro for the minimum required size of the answer area, or use the length field (LAA\_LEN) in the LAA.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 2-byte field that contains the size, in bytes, of the answer area.

**,CONTOKEN=contoken**

Use this input parameter to specify the connect token that was returned by the IXLCONN service. The connect token uniquely identifies your connection to the list structure, and must be specified on each IXLLIST invocation.

The connect token is available in the IXLCONN answer area that is mapped by IXLYCONA.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 16-byte field that contains the connect token.

**,DRIVEEXIT=YES****,DRIVEEXIT=NO**

Use this input parameter to specify whether list services should drive the CONTOKEN connection's list transition exit when the state of the LISTNUM list changes state from empty to non-empty. The list transition exit informs the connection that one of the lists being monitored has changed state. To determine which of the monitored lists has changed state, use the IXLVECTR macro.

**YES**

The connection's list transition exit will be called when a monitored list changes state from empty to non-empty.

**NO**

The connection's list transition exit will not be called when a monitored list changes state. However, the vector index associated with the list will be updated to reflect the current empty/non-empty state information.

**,LISTNUM=listnum**

Use this input parameter to specify the number of the list you wish to monitor.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 4-byte field that contains the number of the list.

**,MF=S**  
**,MF=(L,mfctrl)**  
**,MF=(L,mfctrl,mfattr)**  
**,MF=(L,mfctrl,0D)**  
**,MF=(M,mfctrl)**  
**,MF=(M,mfctrl,COMPLETE)**  
**,MF=(M,mfctrl,NOCHECK)**  
**,MF=(E,mfctrl)**  
**,MF=(E,mfctrl,COMPLETE)**  
**,MF=(E,mfctrl,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

#### **,mfctrl**

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

#### **,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

#### **,COMPLETE**

#### **,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

#### **COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if SMILE=*var* were an optional parameter and the default is SMILE=NO\_SMILE then it would not be documented. However, if the default was SMILE=:~), then it would be documented because a value would be the default.

#### **NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

**,MODE=SYNCSUSPEND**  
**,MODE=SYNCECB**  
**,MODE=SYNCEXIT**  
**,MODE=SYNCTOKEN**  
**,MODE=ASYNCECB**  
**,MODE=ASYNCEXIT**  
**,MODE=ASYNCTOKEN**  
**,MODE=ASYNCSNORESPONSE**

Use this input parameter to specify:

- Whether the request is to be performed synchronously or asynchronously
- How you want to be notified of request completion

**MODE=SYNCSUSPEND**

The request is suspended until it can complete synchronously. To use this option, your program must be enabled for I/O and external interrupts.

**MODE=SYNCECB**

The request processes synchronously if possible. If the request processes asynchronously, the ECB specified by REQECB is posted when the request completes.

**MODE=SYNCEXIT**

The request processes synchronously if possible. If the request processes asynchronously, your complete exit is given control when the request completes.

**MODE=SYNCTOKEN**

The request processes synchronously if possible. If the request processes asynchronously, an asynchronous request token is returned in the area specified by REQTOKEN on invocation of the request. Use the returned request token on the IXLFComp macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=SYNCTOKEN is specified.

**MODE=ASYNCECB**

The request processes asynchronously. The ECB specified by REQECB is posted when the request completes.

**MODE=ASYNCEXIT**

The request processes asynchronously. Your complete exit is given control when the request completes.

**MODE=ASYNCTOKEN**

The request processes asynchronously. An asynchronous request token is returned in the area specified by REQTOKEN upon invocation of the request. Use the returned request token on the IXLFComp macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=ASYNCTOKEN is specified.

**MODE=ASYNCSNORESPONSE**

The request processes asynchronously. No notification of request completion is provided. The answer area (ANSAREA) fields will not contain valid information when this mode is specified.

**Note:** You cannot code MODE=ASYNCSNORESPONSE with LOCKINDEX, BUFFER, or BUFLIST.

**,PLISTVER=IMPLIED\_VERSION**  
**,PLISTVER=MAX**  
**,PLISTVER=plistver**

Use this input parameter to specify the version of the macro. See [“Understanding IXLLIST Version Support”](#) on page 965 for a descriptions of the options available with PLISTVER.

**,REQDATA=NO\_REQDATA**  
**,REQDATA=reqdata**

Use this input parameter with MODE=SYNCEXIT or MODE=ASYNCEXIT to pass any data you choose to the complete exit. The exit will get control only if the request is processed asynchronously.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the data to be passed to the complete exit.

**,REQECB=reqecb**

Use this output parameter with either MODE=SYNCECB or MODE=ASYNCECB to specify the address of an ECB, which is to be posted when the request completes if the request was processed asynchronously.

Before coding REQECB, you must ensure that:

- You initialize the ECB before you issue the request.
- The ECB resides in either common storage or the home address space where IXLCONN was issued.
- Any tasks that wait for the ECB to be posted reside in the home address space where IXLCONN was issued.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that contains the address of the ECB to be posted when the request completes. The ECB must be aligned on a fullword boundary.

**,REQID=NO\_REQID**

**,REQID=reqid**

Use this input parameter to specify a user-defined request identifier to be associated with the request. You can specify this request identifier on the IXPURGE macro to cancel a request that has not yet been processed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the user-defined request identifier.

**,REQTOKEN=reqtoken**

Use this output parameter with either MODE=SYNCTOKEN or MODE=ASYNCTOKEN to specify the address of a storage area to receive the request token that is returned when the request will be processed asynchronously. This token, which uniquely identifies the request, must be used as input to the IXLFComp macro, which you use to determine if the request has completed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field where the system will put the request token.

**,RETCODE=retcode**

Use this output parameter to specify a field to contain the return code. (The return code is also returned in register 15.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the return code when the request has completed.

**,RSNCODE=rsncode**

Use this output parameter to specify a field to contain the reason code returned, if applicable. (The reason code is also returned in register 0.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the reason code (if any) when the request has completed.

**,VECTORINDEX=vectorindex**

Use this input parameter to specify the list notification vector index to be associated with the list specified by LISTNUM. If a start request completes successfully, the index identified by this vector index reflects the list's state—either empty or non-empty.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 4-byte field that contains the list notification vector index.

## ABEND Codes

---

Abend X'026' (See [z/OS MVS System Codes](#) for more information on this abend.)

## Return and Reason Codes

When the system returns control to your program:

- GPR 15 (and RETCODE, if specified) contains a return code.
- GPR 0 (and RSNCODE, if specified) contains a reason code, if applicable.

**Note:** The return and reason codes will also be put into the answer area (mapped by IXLYLAA), if there is one, when the request completes.

The IXLYCON macro provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

<b>0</b>	IXLRETCODEOK
<b>4</b>	IXLRETCODEWARNING
<b>8</b>	IXLRETCODEPARMERROR
<b>C</b>	IXLRETCODEENVERROR
<b>10</b>	IXLRETCODECOMPERROR

The following table contains hexadecimal return and reason codes, the equate symbols associated with each reason code, and the meaning and suggested action for each return and reason code.

Table 63. Return and Reason Codes for IXLLIST REQUEST=MONITOR_LIST Macro		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0	None.	<p><b>Meaning:</b> If you specified a MODE value of ASYNCECB, ASYNCEXIT, ASYNCTOKEN, or ASYNCNORESPONSE, the request has been successfully initiated. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed. For any other MODE value, the request has successfully completed.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=ASYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=ASYNCEXIT, the connection's complete exit will be given control when the request has completed.</li> <li>• If you specified MODE=ASYNCTOKEN, use REQTOKEN when calling IXLFCOMP to determine when the request has completed.</li> </ul>

Table 63. Return and Reason Codes for IXLLIST REQUEST=MONITOR\_LIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	xxxx0402	<p><b>Equate Symbol:</b> IXLRSNCODEASYNCH</p> <p><b>Meaning:</b> The request could not be processed synchronously. It will be processed asynchronously. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed.</p> <ul style="list-style-type: none"> <li>• If MODE=SYNCECB was specified, the ECB specified for REQECB will be posted when the request has finished.</li> <li>• If MODE=SYNCEXIT was specified, the connection's complete exit will be given control when the request has finished.</li> <li>• If MODE=SYNCTOKEN was specified, an asynchronous request token was returned in the area specified by REQTOKEN. This request token may be specified on the IXLFCOMP macro to determine when the request has finished.</li> </ul> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=SYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=SYNCEXIT, the connection's complete exit will be given control when the request has completed.</li> <li>• If you specified MODE=SYNCTOKEN, use REQTOKEN when calling IXLFCOMP to determine when the request has completed.</li> </ul>
8	xxxx0801	<p><b>Equate Symbol:</b> IXLRSNCODEBADPARMLIST</p> <p><b>Meaning:</b> Program error. The parameter list for this request is not addressable. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Verify that:</p> <ul style="list-style-type: none"> <li>• The parameter list address is uncorrupted.</li> <li>• The parameter list is addressable in the caller's primary address space.</li> <li>• If you are issuing this macro while disabled, the parameter list resides in either page-fixed or DREF storage.</li> <li>• If you are issuing this macro in AR-mode and you specified the parameter list address using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If you are issuing this macro in AR-mode, SYSSTATE ASCENV=AR must be issued before issuing this macro.</li> </ul>



Table 63. Return and Reason Codes for IXLLIST REQUEST=MONITOR\_LIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0804	<p><b>Equate Symbol:</b> IXLRNCODEBADVERSIONNUM (or IXLRNCODEBADVERSION#)</p> <p><b>Meaning:</b> The version number in the macro parameter list is not compatible with the level of XES currently being used. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that your program did not overlay the parameter list storage.</li> <li>• Verify that your program was assembled with the correct macro library for the release of MVS on which your program is running.</li> </ul>
8	xxxx080A	<p><b>Equate Symbol:</b> IXLRNCODEBADCONTOKEN</p> <p><b>Meaning:</b> Program error. The specified connect token (CONTOKEN) was not valid for one of the following reasons:</p> <ol style="list-style-type: none"> <li>1. The user with the connection identifier represented by the token has disconnected from the structure.</li> <li>2. The connector's task (the task that issued IXLCONN) ended.</li> <li>3. The specified token is not the token that was returned from IXLCONN.</li> <li>4. The request was issued from an address space other than the address space in which IXLCONN was issued.</li> <li>5. The connect token was invalidated during rebuild.</li> <li>6. The connect token was invalidated by XES.</li> </ol> <p><b>Note:</b> The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Take the action with the corresponding meaning.</p> <ol style="list-style-type: none"> <li>1. Discontinue use of the structure. Perform recovery and cleanup for the structure and check the protocol for the use of the structure to determine why you did not know of the disconnection.</li> <li>2. Discontinue use of the structure. Perform recovery and cleanup for the structure and check the protocol for the use of the structure to determine why you did not know that the task ended.</li> <li>3. The contoken is returned in the CONACONTOKEN field of the answer area specified on the IXLCONN request.</li> <li>4. Issue your request from the same address space the IXLCONN was issued in.</li> <li>5. Wait for the rebuild to complete, and try again.</li> <li>6. Discontinue use of the structure. Perform recovery and cleanup for the structure.</li> </ol>

Table 63. Return and Reason Codes for IXLLIST REQUEST=MONITOR\_LIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0824	<p><b>Equate Symbol:</b> IXLRSNCODEWRONGSTRTYPE</p> <p><b>Meaning:</b> Program error. The connection specified by CONTOKEN is not to a list structure.</p> <p><b>Action:</b> Specify the appropriate CONTOKEN value, and rerun your program. The CONTOKEN is returned in the answer area provided on the IXLCONN invocation. The type of structure built is passed on the TYPE parameter of the IXLCONN macro.</p>
8	xxxx0838	<p><b>Equate Symbol:</b> IXLRSNCODEBADANSAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by ANSAREA is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The answer area address specified by ANSAREA is valid.</li> <li>• If the caller is running in AR-mode and the ANSAREA parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• The address specified for ANSAREA is addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are calling IXLLIST while disabled, ANSAREA must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLLIST in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLIST macro.</li> </ul>
8	xxxx0839	<p><b>Equate Symbol:</b> IXLRSNCODEBADREQTOKENAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by REQTOKEN is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The request token area specified by REQTOKEN is valid.</li> <li>• If the caller is running in AR-mode and the REQTOKEN parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If you are calling IXLLIST while disabled, REQTOKEN must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLLIST in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLIST macro.</li> </ul>
8	xxxx083D	<p><b>Equate Symbol:</b> IXLRSNCODEBADANSLEN</p> <p><b>Meaning:</b> Program error. The length of the answer area, as specified by ANSLEN, is not sufficient to contain answer area information. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Increase the size of the answer area, and rerun your program. The minimum size of the ANSAREA is in the LAA_LEN field of the IXLYLAA mapping macro.</p>

Table 63. Return and Reason Codes for IXLLIST REQUEST=MONITOR\_LIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0847	<p><b>Equate Symbol:</b> IXLRNCODEBADLISTNUMBER</p> <p><b>Meaning:</b> Program error. The specified LISTNUM value exceeds the number of lists for the structure.</p> <p><b>Action:</b> Correct LISTNUM to specify a list number that exists in the structure. The number of lists allocated for a structure is determined on the LISTHEADERS parameter of the IXLCONN macro. The list numbers go from 0 to n-1, where n is the number of lists specified.</p>
8	xxxx0851	<p><b>Equate Symbol:</b> IXLRNCODENOSUSPENDISABLE</p> <p><b>Meaning:</b> Program error. The request failed because MODE=SYNCSUSPEND was specified, but the caller is disabled and cannot be suspended.</p> <p><b>Action:</b> Either specify another MODE value, or become enabled (release the CPU lock), then reissue the request.</p>
8	xxxx0852	<p><b>Equate Symbol:</b> IXLRNCODENOLISTVECTOR</p> <p><b>Meaning:</b> Program error. The request failed because no local vector for monitoring lists or the user's event queue exists for this connection. Either a list notification vector for monitoring lists or the user's event queue was not requested for this connection or the list notification vector has been deleted.</p> <p><b>Action:</b> Either you are not connected to this structure or the VECTORLEN parameter was not specified when the IXLCONN was done for this structure.</p>
8	xxxx0853	<p><b>Equate Symbol:</b> IXLRNCODEINVLISTVINDEX</p> <p><b>Meaning:</b> Program error. The specified list notification vector index (VECTORINDEX) was not valid. This might be because the vector index you specified is greater than the number of vector entries in the list notification vector.</p> <p><b>Action:</b> Verify the list being accessed. Change the vector index to a smaller value, or use the IXLVECTR macro to increase the size of the list notification vector. The number of vector indexes is specified on the VECTORLEN parameter for the IXLCONN macro. This value may also be changed by the IXLVECTR macro. The vector entries are numbered from 0 to n-1 where n is the number of entries.</p>

Table 63. Return and Reason Codes for IXLLIST REQUEST=MONITOR\_LIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C06	<p><b>Equate Symbol:</b> IXLRSNCODENOCNN</p> <p><b>Meaning:</b> Environmental error. This system does not have connectivity to the coupling facility that contains the list structure. Possible reasons for this are:</p> <ul style="list-style-type: none"> <li>• The operator issued VARY PATH,OFFLINE.</li> <li>• The operator issued CONFIG CHP,OFFLINE.</li> <li>• Hardware errors to the coupling facility.</li> <li>• Facility or path failure to the coupling facility.</li> </ul> <p><b>Action:</b> Begin rebuilding the structure on a different coupling facility, or disconnect from the structure.</p>
C	xxxx0C13	<p><b>Equate Symbol:</b> IXLRSNCODEREQPURGED</p> <p><b>Meaning:</b> Environmental error. The request was purged prior to completion of the request. Possible reasons include:</p> <ul style="list-style-type: none"> <li>• The connector failed.</li> <li>• The connector disconnected.</li> <li>• The requestor failed.</li> <li>• The request was purged by IXLPURGE.</li> <li>• Requests were purged when the connector provided an IXLEERSP response for the Rebuild Stop or Rebuild Cleanup event.</li> <li>• The secondary address space was no longer valid.</li> </ul> <p><b>Action:</b> None if this is expected. Otherwise, determine why the connector failed.</p>
C	xxxx0C14	<p><b>Equate Symbol:</b> IXLRSNCODESTATUSUNKNOWN</p> <p><b>Meaning:</b> Environmental error. The request has completed, but the final disposition of the request cannot be determined. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the validity of your data by comparing the expected results with what is in the coupling facility.</li> <li>• Consider running rebuild processing to move the structure to attempt to protect it from further errors of this kind.</li> </ul>
C	xxxx0C25	<p><b>Equate Symbol:</b> IXLRSNCODESTRFAILURE</p> <p><b>Meaning:</b> Environmental error. The list structure failed prior to completion of the request.</p> <p><b>Action:</b> Either rebuild or disconnect from the structure.</p>

Table 63. Return and Reason Codes for IXLLIST REQUEST=MONITOR\_LIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0CA0	<b>Equate Symbol:</b> IXLRSNCODEQUIESCEDSUSPENDFAIL <b>Meaning:</b> Environmental error. The request is failed because the structure is quiesced for a system-managed process and SUSPEND=FAIL is specified on the IXLCONN. <b>Action:</b> None, if this is expected.
C	xxxxFFFF	<b>Equate Symbol:</b> IXLRSNCODENOTAVAILABLE <b>Meaning:</b> Environmental error. XES functions are not available. This could occur because the coupling facility hardware necessary to provide XES functions is not present. <b>Action:</b> Re-IPL the system, or follow your particular failure management protocol.
10	xxxx10xx	<b>Meaning:</b> System error. XES processing failure. The state of the involved structure and the disposition of the request are unpredictable. The answer area (ANSAREA) fields do not contain valid information. <b>Action:</b> Contact the IBM support center.



## Chapter 64. IXLLIST REQUEST=MONITOR\_SUBLIST

### Description

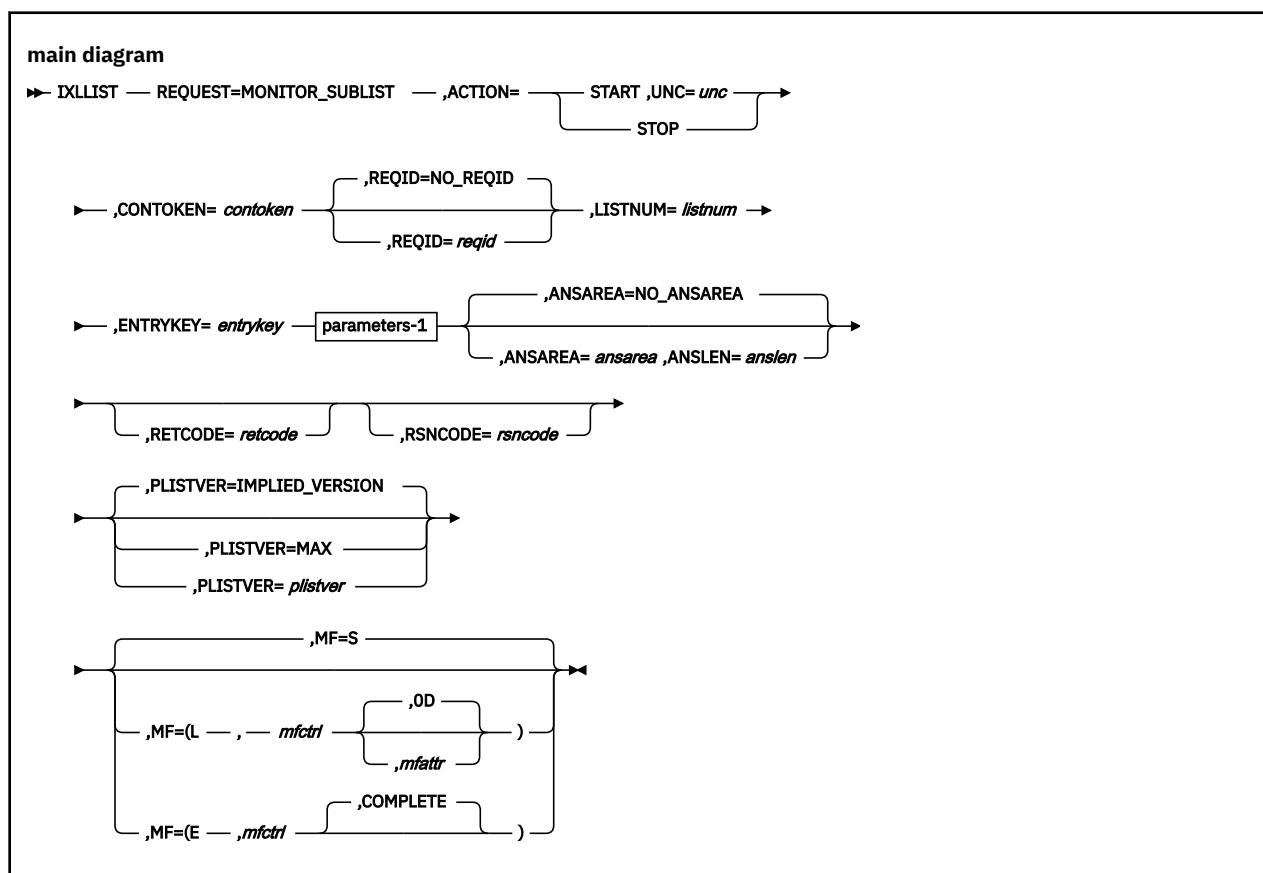
A MONITOR\_SUBLIST request allows you to start (ACTION=START) or stop (ACTION=STOP) monitoring a designated sublist within the list structure. The sublist is identified by list number (LISTNUM) and entry key (ENTRYKEY). To accomplish sublist monitoring, you must specify when you connect to the list structure that you have a local vector associated with your connection to the structure (VECTORLEN keyword) and that the structure supports keyed entries (REFOPTION=KEY).

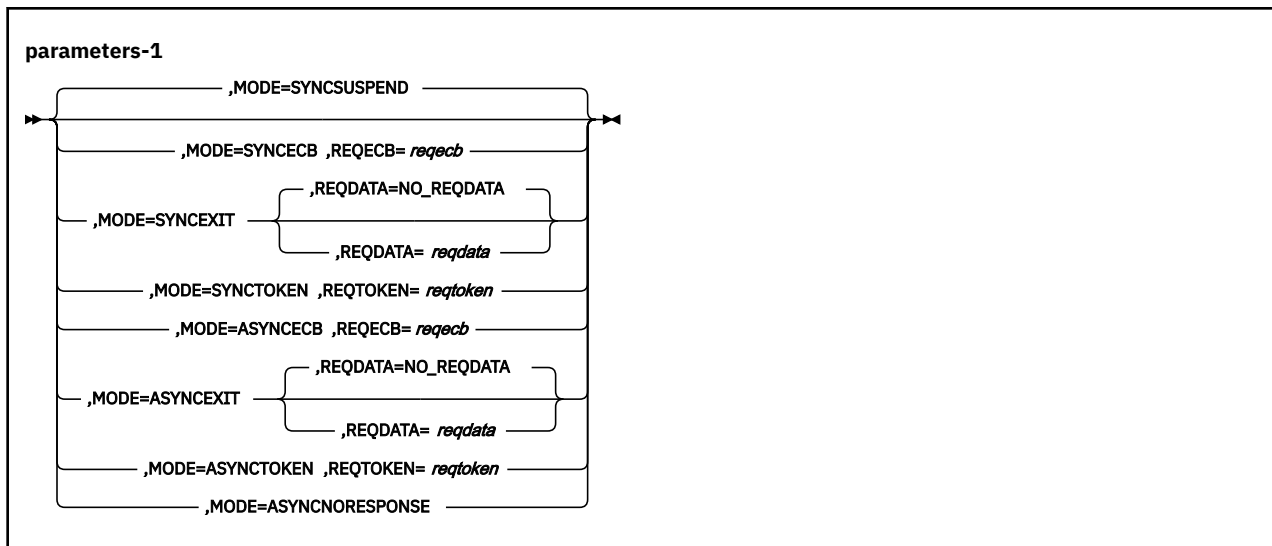
While sublist monitoring is in effect, the system will queue or withdraw event monitor controls (EMCs) on your event queue to indicate the empty or nonempty state of the sublist. Use sublist monitoring in conjunction with event queue monitoring to process state transitions for monitored sublists.

A MONITOR\_SUBLIST request can be issued only for keyed list structures allocated in a coupling facility of CFLEVEL=3 or higher. MONITOR\_SUBLIST requests issued for a list structure allocated in a coupling facility of CFLEVEL=0, 1, or 2 will fail.

### Syntax Diagram

The syntax diagram for IXLLIST REQUEST=MONITOR\_SUBLIST is as follows:





**Note:** If `MODE=SYNCTOKEN` or `MODE=ASYNCTOKEN` is specified, `ANSAREA=ansarea` and `ANSLEN=anslen` are required.

## Parameter Descriptions

The parameter descriptions for `REQUEST=MONITOR_SUBLIST` are listed in alphabetical order. Default values are underlined:

### **REQUEST=MONITOR\_SUBLIST**

Use this input parameter to specify that monitoring of the sublist specified by `LISTNUM` and `ENTRYKEY` be started or stopped.

### **,ACTION=START**

### **,ACTION=STOP**

Use this input parameter to specify whether sublist monitoring is to be started or stopped.

### **START**

Start monitoring the sublist identified by `LISTNUM` and `ENTRYKEY` or restart monitoring with new parameters.

### **STOP**

Stop monitoring the sublist identified by `LISTNUM` and `ENTRYKEY`.

### **,ANSAREA=NO\_ANSAREA**

### **,ANSAREA=ansarea**

Use this output parameter to specify an answer area to contain information returned from the request. The format of the answer area is described by mapping macro `IXLYLAA`. Upon successful completion of a request with `ACTION=START`, the answer area contains the number of event monitor controls in use (field `LAAMNSL_EMCCNT`), the maximum number of event monitor controls for the structure (field `LAAMNSL_MAXEMCCNT`), and the state (empty or non-empty) of the sublist (field `LAAMNSL_ENTRYQUEUED`).

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an area (with a length of `ANSLEN`) to contain the information.

### **,ANSLEN=anslen**

Use this input parameter to specify the size of the storage area specified by `ANSAREA`.

Check the prologue of the `IXLYLAA` mapping macro for the minimum required size of the answer area, or use the length field (`LAA_LEN`) in the LAA.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 2-byte field that contains the size, in bytes, of the answer area.



**,CONTOKEN=contoken**

Use this input parameter to specify the connect token that was returned by the IXLCONN service. The connect token uniquely identifies your connection to the list structure, and must be specified on each IXLLIST invocation.

The connect token is available in the IXLCONN answer area that is mapped by IXLYCONA.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 16-byte field that contains the connect token.

**,ENTRYKEY=entrykey**

Use this input parameter to specify the list entry key to be used in conjunction with LISTNUM to designate the sublist for which monitoring is to be started or stopped.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-byte field that contains the entry key of the sublist.

**,LISTNUM=listnum**

Use this input parameter to specify the number of the list containing the sublist you wish to monitor.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 4-byte field that contains the number of the list.

**,MF=S****,MF=(L,mfctrl)****,MF=(L,mfctrl,mfattr)****,MF=(L,mfctrl,0D)****,MF=(M,mfctrl)****,MF=(M,mfctrl,COMPLETE)****,MF=(M,mfctrl,NOCHECK)****,MF=(E,mfctrl)****,MF=(E,mfctrl,COMPLETE)****,MF=(E,mfctrl,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

**,mfctrl**

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

**,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

**,COMPLETE****,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

**COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if SMILE=*var* were an optional parameter and the default is SMILE=NO\_SMILE then it would not be documented. However, if the default was SMILE=:-), then it would be documented because a value would be the default.

**NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

**,MODE=SYNCSUSPEND**

**,MODE=SYNCECB**

**,MODE=SYNCEXIT**

**,MODE=SYNCTOKEN**

**,MODE=ASYNCECB**

**,MODE=ASYNCEXIT**

**,MODE=ASYNCTOKEN**

**,MODE=ASYNCSNORESPONSE**

Use this input parameter to specify:

- Whether the request is to be performed synchronously or asynchronously
- How you want to be notified of request completion

**MODE=SYNCSUSPEND**

The request is suspended until it can complete synchronously. To use this option, your program must be enabled for I/O and external interrupts.

**MODE=SYNCECB**

The request processes synchronously if possible. If the request processes asynchronously, the ECB specified by REQECB is posted when the request completes.

**MODE=SYNCEXIT**

The request processes synchronously if possible. If the request processes asynchronously, your complete exit is given control when the request completes.

**MODE=SYNCTOKEN**

The request processes synchronously if possible. If the request processes asynchronously, an asynchronous request token is returned in the area specified by REQTOKEN on invocation of the request. Use the returned request token on the IXLFCOMP macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=SYNCTOKEN is specified.

**MODE=ASYNCECB**

The request processes asynchronously. The ECB specified by REQECB is posted when the request completes.

**MODE=ASYNCEXIT**

The request processes asynchronously. Your complete exit is given control when the request completes.

**MODE=ASYNCTOKEN**

The request processes asynchronously. An asynchronous request token is returned in the area specified by REQTOKEN upon invocation of the request. Use the returned request token on the IXLFCOMP macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=ASYNCTOKEN is specified.

**MODE=ASYNCSNORESPONSE**

The request processes asynchronously. No notification of request completion is provided. The answer area (ANSAREA) fields will not contain valid information when this mode is specified.

**Note:** You cannot code MODE=ASYNCRESPONSE with LOCKINDEX, BUFFER, or BUFLIST.

**,PLISTVER=IMPLIED\_VERSION**

**,PLISTVER=MAX**

**,PLISTVER=plistver**

Use this input parameter to specify the version of the macro. See “Understanding IXLLIST Version Support” on page 965 for a description of the options available with PLISTVER.

**,REQDATA=NO\_REQDATA**

**,REQDATA=reqdata**

Use this input parameter with MODE=SYNCEXIT or MODE=ASYNCEXIT to pass any data you choose to the complete exit. The exit will get control only if the request is processed asynchronously.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the data to be passed to the complete exit.

**,REQECB=reqecb**

Use this output parameter with either MODE=SYNCECB or MODE=ASYNCECB to specify the address of an ECB, which is to be posted when the request completes if the request was processed asynchronously.

Before coding REQECB, you must ensure that:

- You initialize the ECB before you issue the request.
- The ECB resides in either common storage or the home address space where IXLCONN was issued.
- Any tasks that wait for the ECB to be posted reside in the home address space where IXLCONN was issued.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that contains the address of the ECB to be posted when the request completes. The ECB must be aligned on a fullword boundary.

**,REQID=NO\_REQID**

**,REQID=reqid**

Use this input parameter to specify a user-defined request identifier to be associated with the request. You can specify this request identifier on the IXPURGE macro to cancel a request that has not yet been processed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the user-defined request identifier.

**,REQTOKEN=reqtoken**

Use this output parameter with either MODE=SYNCTOKEN or MODE=ASYNCTOKEN to specify the address of a storage area to receive the request token that is returned when the request will be processed asynchronously. This token, which uniquely identifies the request, must be used as input to the IXLFComp macro, which you use to determine if the request has completed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field where the system will put the request token.

**,RETCODE=retcode**

Use this output parameter to specify a field to contain the return code. (The return code is also returned in register 15.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the return code when the request has completed.

**,RSNCODE=rsncode**

Use this output parameter to specify a field to contain the reason code returned, if applicable. (The reason code is also returned in register 0.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the reason code (if any) when the request has completed.

**,UNC=unc**

Use this input parameter to specify the user notification control information that represents the user's monitoring interest in the designated sublist. The user notification controls are contained in the event monitor controls (EMC) which is queued to the user's event queue when the monitored sublist becomes nonempty.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-byte field that contains the user notification controls.

## ABEND Codes

---

Abend X'026' (See [z/OS MVS System Codes](#) for more information on this abend.)

## Return and Reason Codes

---

When the system returns control to your program:

- GPR 15 (and RETCODE, if specified) contains a return code.
- GPR 0 (and RSNCODE, if specified) contains a reason code, if applicable.

**Note:** The return and reason codes will also be put into the answer area (mapped by IXLYLAA), if there is one, when the request completes.

The IXLYCON macro provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

**0**

IXLRETCODEOK

**4**

IXLRETCODEWARNING

**8**

IXLRETCODEPARMERROR

**C**

IXLRETCODEENVERROR

**10**

IXLRETCODECOMPERROR

The following table contains hexadecimal return and reason codes, the equate symbols associated with each reason code, and the meaning and suggested action for each return and reason code.

Table 64. Return and Reason Codes for IXLLIST REQUEST=MONITOR\_SUBLIST Macro

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0	None.	<p><b>Meaning:</b> If you specified a MODE value of ASYNCECB, ASYNCEXIT, ASYNCTOKEN, or ASYNCSNORESPONSE, the request has been successfully initiated. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed. For any other MODE value, the request has successfully completed.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=ASYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=ASYNCEXIT, the connection's complete exit will be given control when the request has completed.</li> <li>• If you specified MODE=ASYNCTOKEN, use REQTOKEN when calling IXLFCOMP to determine when the request has completed.</li> </ul>
4	xxxx0402	<p><b>Equate Symbol:</b> IXLRNSCODEASYNCH</p> <p><b>Meaning:</b> The request could not be processed synchronously. It will be processed asynchronously. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed.</p> <ul style="list-style-type: none"> <li>• If MODE=SYNCECB was specified, the ECB specified for REQECB will be posted when the request has finished.</li> <li>• If MODE=SYNCEXIT was specified, the connection's complete exit will be given control when the request has finished.</li> <li>• If MODE=SYNCTOKEN was specified, an asynchronous request token was returned in the area specified by REQTOKEN. This request token may be specified on the IXLFCOMP macro to determine when the request has finished.</li> </ul> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=SYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=SYNCEXIT, the connection's complete exit will be given control when the request has completed.</li> <li>• If you specified MODE=SYNCTOKEN, use REQTOKEN when calling IXLFCOMP to determine when the request has completed.</li> </ul>

Table 64. Return and Reason Codes for IXLLIST REQUEST=MONITOR_SUBLIST Macro (continued)		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0801	<p><b>Equate Symbol:</b> IXLRSNCODEBADPARMLIST</p> <p><b>Meaning:</b> Program error. The parameter list for this request is not addressable. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Verify that:</p> <ul style="list-style-type: none"> <li>• The parameter list address is uncorrupted.</li> <li>• The parameter list is addressable in the caller's primary address space.</li> <li>• If you are issuing this macro while disabled, the parameter list resides in either page-fixed or DREF storage.</li> <li>• If you are issuing this macro in AR-mode and you specified the parameter list address using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If you are issuing this macro in AR-mode, SYSSTATE ASCENV=AR must be issued before issuing this macro.</li> </ul>
8	xxxx0804	<p><b>Equate Symbol:</b> IXLRSNCODEBADVERSIONNUM (or IXLRSNCODEBADVERSION#)</p> <p><b>Meaning:</b> The version number in the macro parameter list is not compatible with the level of XES currently being used. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that your program did not overlay the parameter list storage.</li> <li>• Verify that your program was assembled with the correct macro library for the release of MVS on which your program is running.</li> </ul>

Table 64. Return and Reason Codes for IXLLIST REQUEST=MONITOR\_SUBLIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx080A	<p><b>Equate Symbol:</b> IXLRSNCODEBADCONTOKEN</p> <p><b>Meaning:</b> Program error. The specified connect token (CONTOKEN) was not valid for one of the following reasons:</p> <ol style="list-style-type: none"> <li>1. The user with the connection identifier represented by the token has disconnected from the structure.</li> <li>2. The connector's task (the task that issued IXLCONN) ended.</li> <li>3. The specified token is not the token that was returned from IXLCONN.</li> <li>4. The request was issued from an address space other than the address space in which IXLCONN was issued.</li> <li>5. The connect token was invalidated during rebuild.</li> <li>6. The connect token was invalidated by XES.</li> </ol> <p><b>Note:</b> The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Take the action with the corresponding meaning.</p> <ol style="list-style-type: none"> <li>1. Discontinue use of the structure. Perform recovery and cleanup for the structure and check the protocol for the use of the structure to determine why you did not know of the disconnection.</li> <li>2. Discontinue use of the structure. Perform recovery and cleanup for the structure and check the protocol for the use of the structure to determine why you did not know that the task ended.</li> <li>3. The contoken is returned in the CONACONTOKEN field of the answer area specified on the IXLCONN request.</li> <li>4. Issue your request from the same address space the IXLCONN was issued in.</li> <li>5. Wait for the rebuild to complete, and try again.</li> <li>6. Discontinue use of the structure. Perform recovery and cleanup for the structure.</li> </ol>
8	xxxx0824	<p><b>Equate Symbol:</b> IXLRSNCODEWRONGSTRTYPE</p> <p><b>Meaning:</b> Program error. The connection specified by CONTOKEN is not to a list structure.</p> <p><b>Action:</b> Specify the appropriate CONTOKEN value, and rerun your program. The CONTOKEN is returned in the answer area provided on the IXLCONN invocation. The type of structure built is passed on the TYPE parameter of the IXLCONN macro.</p>

Table 64. Return and Reason Codes for IXLLIST REQUEST=MONITOR\_SUBLIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0838	<p><b>Equate Symbol:</b> IXLRSNCODEBADANSAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by ANSAREA is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The answer area address specified by ANSAREA is valid.</li> <li>• If the caller is running in AR-mode and the ANSAREA parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• The address specified for ANSAREA is addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are calling IXLLIST while disabled, ANSAREA must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLLIST in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLIST macro.</li> </ul>
8	xxxx0839	<p><b>Equate Symbol:</b> IXLRSNCODEBADREQTOKENAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by REQTOKEN is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The request token area specified by REQTOKEN is valid.</li> <li>• If the caller is running in AR-mode and the REQTOKEN parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If you are calling IXLLIST while disabled, REQTOKEN must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLLIST in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLIST macro.</li> </ul>
8	xxxx083D	<p><b>Equate Symbol:</b> IXLRSNCODEBADANSLEN</p> <p><b>Meaning:</b> Program error. The length of the answer area, as specified by ANSLEN, is not sufficient to contain answer area information. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Increase the size of the answer area, and rerun your program. The minimum size of the ANSAREA is in the LAA_LEN field of the IXLYLAA mapping macro.</p>



Table 64. Return and Reason Codes for IXLLIST REQUEST=MONITOR\_SUBLIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0847	<p><b>Equate Symbol:</b> IXLRSNCODEBADLISTNUMBER</p> <p><b>Meaning:</b> Program error. The specified LISTNUM value exceeds the number of lists for the structure.</p> <p><b>Action:</b> Correct LISTNUM to specify a list number that exists in the structure. The number of lists allocated for a structure is determined on the LISTHEADERS parameter of the IXLCONN macro. The list numbers go from 0 to n-1, where n is the number of lists specified.</p>
8	xxxx084A	<p><b>Equate Symbol:</b> IXLRSNCODENOKEYS</p> <p><b>Meaning:</b> Program error. The structure does not support the use of entry keys. The IXLLIST request type either required the structure to support entry keys or designated a list entry or list position by list number and entry key.</p> <p><b>Action:</b> Ensure that you are connected to the intended structure. The use of keys for a structure is determined by the REFOPTION keyword on the IXLCONN macro.</p>
8	xxxx0851	<p><b>Equate Symbol:</b> IXLRSNCODENOSUSPENDISABLE</p> <p><b>Meaning:</b> Program error. The request failed because MODE=SYNCSUSPEND was specified, but the caller is disabled and cannot be suspended.</p> <p><b>Action:</b> Either specify another MODE value, or become enabled (release the CPU lock), then reissue the request.</p>
8	xxxx0852	<p><b>Equate Symbol:</b> IXLRSNCODENOLISTVECTOR</p> <p><b>Meaning:</b> Program error. The request failed because no local vector for monitoring lists or the user's event queue exists for this connection. Either a list notification vector for monitoring lists or the user's event queue was not requested for this connection or the list notification vector has been deleted.</p> <p><b>Action:</b> Either you are not connected to this structure or the VECTORLEN parameter was not specified when the IXLCONN was done for this structure.</p>
C	xxxx0C06	<p><b>Equate Symbol:</b> IXLRSNCODENOCOONN</p> <p><b>Meaning:</b> Environmental error. This system does not have connectivity to the coupling facility that contains the list structure. Possible reasons for this are:</p> <ul style="list-style-type: none"> <li>• The operator issued VARY PATH,OFFLINE.</li> <li>• The operator issued CONFIG CHP,OFFLINE.</li> <li>• Hardware errors to the coupling facility.</li> <li>• Facility or path failure to the coupling facility.</li> </ul> <p><b>Action:</b> Begin rebuilding the structure on a different coupling facility, or disconnect from the structure.</p>

Table 64. Return and Reason Codes for IXLLIST REQUEST=MONITOR_SUBLIST Macro (continued)		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C13	<p><b>Equate Symbol:</b> IXLRNCDEREQPURGED</p> <p><b>Meaning:</b> Environmental error. The request was purged prior to completion of the request. Possible reasons include:</p> <ul style="list-style-type: none"> <li>• The connector failed.</li> <li>• The connector disconnected.</li> <li>• The requestor failed.</li> <li>• The request was purged by IXLPURGE.</li> <li>• Requests were purged when the connector provided an IXLEERSP response for the Rebuild Stop or Rebuild Cleanup event.</li> <li>• The secondary address space was no longer valid.</li> </ul> <p><b>Action:</b> None if this is expected. Otherwise, determine why the connector failed.</p>
C	xxxx0C14	<p><b>Equate Symbol:</b> IXLRNCDERSTATUSUNKNOWN</p> <p><b>Meaning:</b> Environmental error. The request has completed, but the final disposition of the request cannot be determined. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the validity of your data by comparing the expected results with what is in the coupling facility.</li> <li>• Consider running rebuild processing to move the structure to attempt to protect it from further errors of this kind.</li> </ul>
C	xxxx0C17	<p><b>Equate Symbol:</b> IXLRNCDERSTRFULL</p> <p><b>Meaning:</b> Environmental error. The request attempted to create a new event monitor controls (EMC) object, but the structure is full and cannot accommodate any more EMCs.</p> <p><b>Action:</b> Determine why the structure is full. You should be monitoring the use of EMC objects (LAAMNSL_EMCCNT is the count of EMCs in use when sublist monitoring was established and LAAMNSL_MAXEMCCNT is the maximum number of EMCs for the structure.) Evaluate this data periodically to ensure structure resources are being used efficiently. You might be able to delete existing EMCs to free up space, or, if rebuild is allowed (IXLCONN macro, ALLOWREBLD parameter), rebuild the structure either to make it larger or with a changed EMCSTGPCT to allow more EMCs to be created.</p>
C	xxxx0C25	<p><b>Equate Symbol:</b> IXLRNCDERSTRFAILURE</p> <p><b>Meaning:</b> Environmental error. The list structure failed prior to completion of the request.</p> <p><b>Action:</b> Either rebuild or disconnect from the structure.</p>

Table 64. Return and Reason Codes for IXLLIST REQUEST=MONITOR\_SUBLIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C68	<b>Equate Symbol:</b> IXLRSNCODEBADREQCFLEVEL <b>Meaning:</b> Environmental error. The request type is not permitted for the level of coupling facility in which the target structure is allocated. <b>Action:</b> Either disconnect from the structure (using IXLDISC) or initiate a rebuild of the structure, if allowed (using IXLREBLD) in a coupling facility of the correct CFLEVEL.
C	xxxx0CA0	<b>Equate Symbol:</b> IXLRSNCODEQUIESCEDSUSPENDFAIL <b>Meaning:</b> Environmental error. The request is failed because the structure is quiesced for a system-managed process and SUSPEND=FAIL is specified on the IXLCONN. <b>Action:</b> None, if this is expected.
C	xxxxFFFF	<b>Equate Symbol:</b> IXLRSNCODENOTAVAILABLE <b>Meaning:</b> Environmental error. XES functions are not available. This could occur because the coupling facility hardware necessary to provide XES functions is not present. <b>Action:</b> Re-IPL the system, or follow your particular failure management protocol.
10	xxxx10xx	<b>Meaning:</b> System error. XES processing failure. The state of the involved structure and the disposition of the request are unpredictable. The answer area (ANSAREA) fields do not contain valid information. <b>Action:</b> Contact the IBM support center.



## Chapter 65. IXLLIST REQUEST=MONITOR\_SUBLISTS

### Description

A MONITOR\_SUBLISTS request allows you to start monitoring a set of sublists in a list structure. Each sublist is designated by a list number and an entry key. You cannot stop monitoring the sublists with the MONITOR\_SUBLISTS request. (Use the MONITOR\_SUBLIST request to stop the monitoring of each individual sublist.)

While sublist monitoring is in effect, the system will queue or withdraw event monitor controls (EMCs) on your event queue to indicate the empty or nonempty state of the sublists. Use sublist monitoring in conjunction with event queue monitoring to process state transitions for monitored sublists.

Information about the sublists to be monitored is contained in the storage area specified by BUFFER or BUFLIST. The information for each sublist entry is mapped by the IXLYMSRI macro. The entries are indexed with the first entry starting at offset zero in the BUFFER or BUFLIST area. To designate the entries to be monitored, specify the starting and ending index numbers of the IXLYMSRI entries with the STARTINDEX and ENDINDEX keywords.

A MONITOR\_SUBLISTS request can complete prematurely for the following reasons:

- The request has exceeded the model-dependent time-out criteria. The index of the next IXLYMSRI entry to be processed is returned in the answer area (ANSAREA).
- There is insufficient event monitor control space in the list structure to register as a sublist monitor. The index of the first unprocessed IXLYMSRI entry (the entry that experienced the failure) is returned in the answer area.
- The user has specified a list number that is not valid in an IXLYMSRI entry. The index of the failing IXLYMSRI entry is returned in the answer area.

Before restarting a MONITOR\_SUBLISTS request that has completed prematurely, you should process the entries that were successfully returned. Determine how to restart based on the type of premature completion that occurred.

- In the case of model-dependent time-out you can update STARTINDEX to field LAAMNSLS\_FAILINDEX in the answer area on a subsequent MONITOR\_SUBLISTS request to finish processing the entries.
- In the case of insufficient EMC space you must take actions to relieve the space constraints before updating STARTINDEX on a subsequent MONITOR\_SUBLISTS request.
- In the case of a list number that is not valid, you must correct the erroneous specification before updating STARTINDEX on a subsequent MONITOR\_SUBLISTS request.

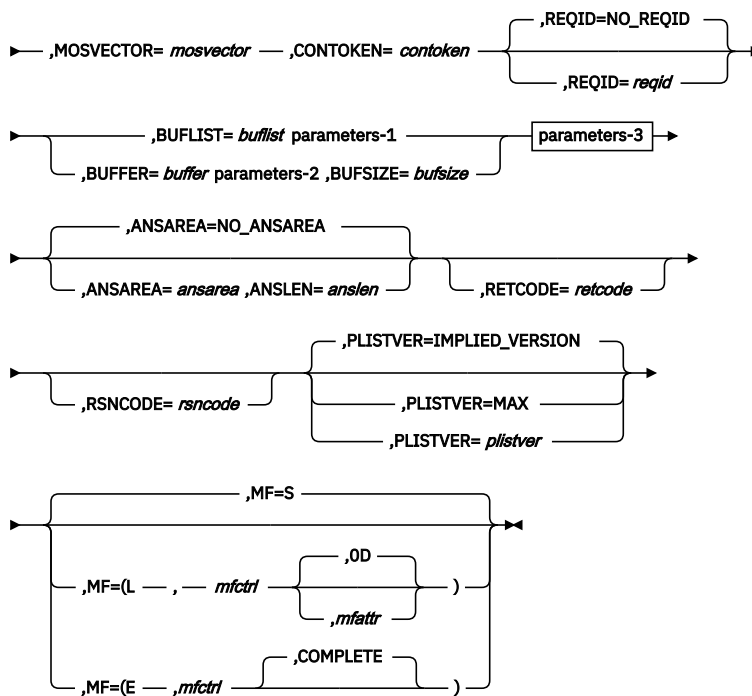
A MONITOR\_SUBLISTS request can be issued only for keyed list structures allocated in a coupling facility of CFLEVEL=3 or higher. MONITOR\_SUBLISTS requests issued for a list structure allocated in a coupling facility of CFLEVEL 0, 1, or 2 will fail.

### Syntax Diagram

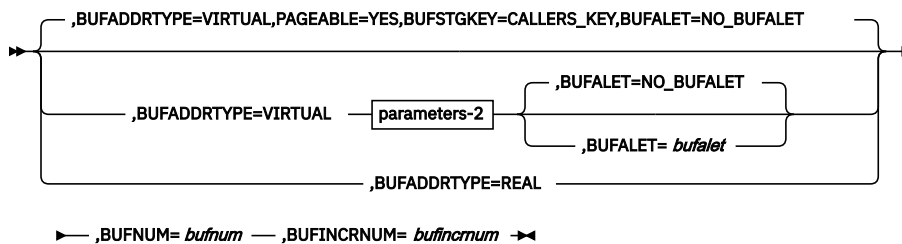
The syntax diagram for IXLLIST REQUEST=MONITOR\_SUBLISTS is as follows:

## main diagram

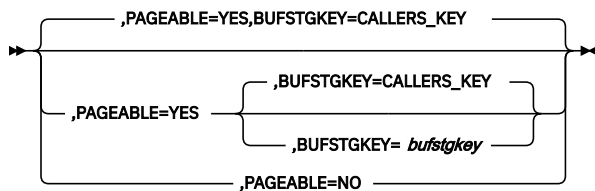
➤ IXLLIST — REQUEST=MONITOR\_SUBLISTS — ,STARTINDEX= *startindex* — ,ENDINDEX= *endindex* ➤

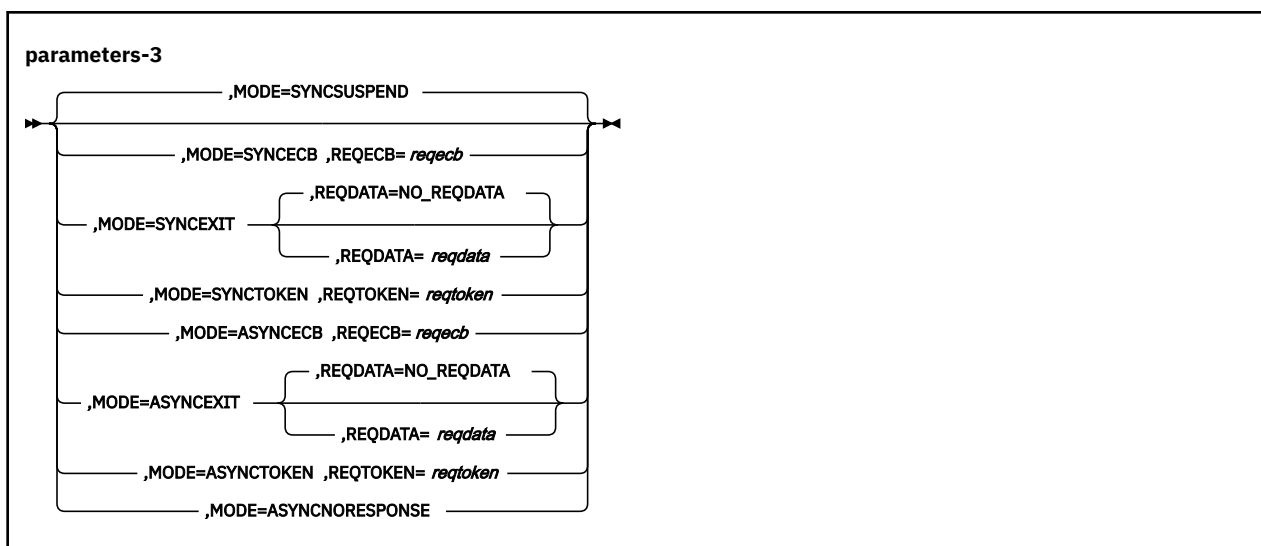


## parameters-1



## parameters-2





**Note:** If MODE=SYNCTOKEN or MODE=ASYNCTOKEN is specified, ANSAREA=*ansarea* and ANSLN=*anslen* are required.

## Parameter Descriptions

The parameter descriptions for REQUEST=MONITOR\_SUBLISTS are listed in alphabetical order. Default values are underlined:

### REQUEST=MONITOR\_SUBLISTS

Use this input parameter to monitor a set of sublists that are identified in an input array of entries.

#### **,ANSAREA=NO\_ANSAREA**

#### **,ANSAREA=*ansarea***

Use this output parameter to specify an answer area to contain information returned from the request. The format of the answer area is described by mapping macro IXLYLAA.

When the request completes successfully or completes prematurely, the answer area contains the number of event monitor controls in use (field LAAMNSLS\_EMCCNT) and the maximum number of event monitor controls for the structure (field LAAMNSLS\_MAXEMCCNT). For a premature completion, the answer area also contains the index of the first unprocessed IXLYMSRI entry (field LAAMNSLS\_FAILINDEX). This index can be specified (using the STARTINDEX parameter) on the next MONITOR\_SUBLISTS request to finish processing the entries in the input array.

See *z/OS MVS Programming: Sysplex Services Guide* for a description of all the relevant IXLYLAA fields for this request.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an area (with a length of ANSLN) to contain the information returned by the request.

#### **,ANSLN=*anslen***

Use this input parameter to specify the size of the storage area specified by ANSAREA.

Check the prologue of the IXLYLAA mapping macro for the minimum required size of the answer area, or use the length field (LAA\_LEN) in the LAA.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 2-byte field that contains the size, in bytes, of the answer area.

#### **,BUFADDRTYPE=VIRTUAL**

#### **,BUFADDRTYPE=REAL**

Use this input parameter to specify whether the buffer addresses specified in the BUFLIST list are virtual storage or real storage addresses.

**VIRTUAL**

The buffer addresses are virtual storage addresses. The virtual storage can be pageable or nonpageable. See the PAGEABLE parameter for information about managing storage binds when specifying virtual storage addresses.

**REAL**

The buffer addresses are real storage addresses.

It is the caller's responsibility to manage the binds between the data buffer virtual storage and the real storage addresses provided. The caller must ensure that the data buffer virtual storage remains bound to the real storage addresses provided until the request completes.

**,BUFALET=NO\_BUFALET****,BUFALET=*bufalet***

Use this input parameter to specify an access list entry token (ALET) to be used in referencing all of the buffers specified by BUFLIST.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 4-byte field that contains the ALET.

**,BUFFER=*buffer***

Use this input parameter to specify a buffer area to hold an array of entries that identify the sublists to be monitored. Each entry is mapped by the IXLYMSRI macro.

You can define the buffer size to be a total of up to 65536 bytes. Depending on the size you select, the following restrictions apply:

- If you specify a buffer size of less than or equal to 4096 bytes, you must ensure that the buffer:
  - Is 256, 512, 1024, or 4096 bytes.
  - Starts on a 256-byte boundary.
  - Does not cross a 4096-byte boundary.
  - Does not start below storage address 512.
- If you specify a buffer size of greater than 4096 bytes, you must ensure that the buffer:
  - Is a multiple of 4096 bytes.
  - Is less than or equal to 65536 bytes.
  - Starts on a 4096-byte boundary.
  - Does not start below storage address 512.

See the BUFSIZE parameter description for defining the size of the buffer.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an area (with a size of BUFSIZE) that contains the array of sublist identifiers.

**,BUFINCRNUM**

Use this input parameter to specify the number of 256-byte segments comprising each buffer in the BUFLIST list.

Valid BUFINCRNUM values are 1,2,4,8, or 16, which correspond to BUFLIST sizes of 256, 512, 1024, 2048, and 4096 bytes respectively.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 1-byte field that contains 1,2,4,8, or 16.

**,BUFLIST=*buflist***

Use this input parameter to specify a list of buffers to hold the array of entries that identify the sublists to be monitored.

BUFLIST specifies a 128-byte storage area that consists of a list of 0 to 16 buffer addresses.

**The 128-byte storage area must:**

- Consist of 0 to 16 elements.



- Each element must consist of an 8-byte field in which:
  - The left (high-order) four bytes are reserved and
  - The right (low-order) four bytes contain the address of a buffer.

**The BUFLIST buffers must:**

- Reside in the same address space or data space.
- Be the same size: either 256, 512, 1024, 2048, or 4096 bytes.
- Start on a 256-byte boundary and not cross a 4096-byte boundary.
- Not start below storage address 512.

**Note:** The buffers do not have to be contiguous in storage. List services treats BUFLIST buffers as a single buffer even if the buffers are not contiguous.

See the BUFNUM and BUFINCRNUM parameter descriptions for specifying the number and size of buffers.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 128-byte area that contains a list of buffer addresses.

**,BUFNUM=*bufnum***

Use this input parameter to specify the number of buffers in the BUFLIST list. Valid BUFNUM values are from 0 to 16. A value of zero indicates that no sublists are to be monitored.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 1-byte field that contains the number of buffers in the buffer list.

**,BUFSIZE=*bufsize***

Use this input parameter to specify the size of the BUFFER area. See the BUFFER parameter description for valid buffer sizes.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a fullword field that contains the size of the buffer (BUFFER) in bytes.

**,BUFSTGKEY=CALLERS\_KEY**

**,BUFSTGKEY=*bufstgkey***

Use this input parameter to specify a storage key that you define and use when referencing the buffers specified by BUFLIST or the buffer that is specified by BUFFER.

If you do not specify BUFSTGKEY, or if you specify BUFSTGKEY=CALLERS\_KEY, all references to one or more buffers are performed by using the caller's PSW key.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of an 8-bit field that contains the storage key in the format B'kkkkxxxx', where kkkk is the key and xxxx is ignored.

**,CONTOKEN=*contoken***

Use this input parameter to specify the connect token that was returned by the IXLCONN service. The connect token uniquely identifies your connection to the list structure, and must be specified on each IXLLIST invocation.

The connect token is available in the IXLCONN answer area that is mapped by IXLYCONA.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 16-byte field that contains the connect token.

**,ENDINDEX=*endindex***

Use this input parameter to specify the index number of the last IXLYMSRI entry to be processed. The valid range is from the STARTINDEX value to 1024.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a halfword field that contains the index of the last entry to be processed.

**,MF=S**  
**,MF=(L,mfctrl)**  
**,MF=(L,mfctrl,mfattr)**  
**,MF=(L,mfctrl,0D)**  
**,MF=(M,mfctrl)**  
**,MF=(M,mfctrl,COMPLETE)**  
**,MF=(M,mfctrl,NOCHECK)**  
**,MF=(E,mfctrl)**  
**,MF=(E,mfctrl,COMPLETE)**  
**,MF=(E,mfctrl,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

#### **,mfctrl**

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

#### **,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

#### **,COMPLETE**

#### **,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

#### **COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if SMILE=*var* were an optional parameter and the default is SMILE=NO\_SMILE then it would not be documented. However, if the default was SMILE=:), then it would be documented because a value would be the default.

#### **NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

**,MODE=SYNCSUSPEND**  
**,MODE=SYNCECB**  
**,MODE=SYNCEXIT**  
**,MODE=SYNCTOKEN**  
**,MODE=ASYNCECB**  
**,MODE=ASYNCEXIT**  
**,MODE=ASYNCTOKEN**  
**,MODE=ASYNCSNORESPONSE**

Use this input parameter to specify:

- Whether the request is to be performed synchronously or asynchronously
- How you want to be notified of request completion

**MODE=SYNCSUSPEND**

The request is suspended until it can complete synchronously. To use this option, your program must be enabled for I/O and external interrupts.

**MODE=SYNCECB**

The request processes synchronously if possible. If the request processes asynchronously, the ECB specified by REQECB is posted when the request completes.

**MODE=SYNCEXIT**

The request processes synchronously if possible. If the request processes asynchronously, your complete exit is given control when the request completes.

**MODE=SYNCTOKEN**

The request processes synchronously if possible. If the request processes asynchronously, an asynchronous request token is returned in the area specified by REQTOKEN on invocation of the request. Use the returned request token on the IXLFComp macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=SYNCTOKEN is specified.

**MODE=ASYNCECB**

The request processes asynchronously. The ECB specified by REQECB is posted when the request completes.

**MODE=ASYNCEXIT**

The request processes asynchronously. Your complete exit is given control when the request completes.

**MODE=ASYNCTOKEN**

The request processes asynchronously. An asynchronous request token is returned in the area specified by REQTOKEN upon invocation of the request. Use the returned request token on the IXLFComp macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=ASYNCTOKEN is specified.

**MODE=ASYNCSNORESPONSE**

The request processes asynchronously. No notification of request completion is provided. The answer area (ANSAREA) fields will not contain valid information when this mode is specified.

**Note:** You cannot code MODE=ASYNCSNORESPONSE with LOCKINDEX, BUFFER, or BUFLIST.

**,MOSVECTOR=mosvector**

Use this output parameter to specify a 128-byte field to contain the bit string representing the monitored object state (empty or nonempty) of each sublist at the time the MONITOR\_SUBLISTS request was processed. The MOSVECTOR bits correspond one-to-one with the IXLYMSRI entries that were passed as input in BUFFER or BUFLIST. Only the bits corresponding to the IXLYMSRI entries that were actually processed on the current request (that is, those between STARTINDEX and ENDINDEX for a request that completed successfully, or between STARTINDEX and the returned index of the first unprocessed entry minus one for requests that completed prematurely) will contain valid monitored object state information for the sublists designated by the corresponding IXLYMSRI entries. Bits in the MOSVECTOR that lie outside the valid range are not meaningful.

When a bit in the valid range is on, the sublist designated by the corresponding IXLYMSRI entry was nonempty at the time the request was processed. When a bit in the valid range is off, the sublist designated by the corresponding IXLYMSRI entry was empty at the time the request was processed.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 128-byte field to contain the bit string indicating the monitored object state of each sublist for which monitoring was requested.

**,PAGEABLE=YES**

**,PAGEABLE=NO**

Use this input parameter to identify whether the storage areas specified by BUFFER or BUFLIST are in pageable or potentially pageable storage.

#### **YES**

Specify this option to indicate that the BUFFER or BUFLIST buffers reside in pageable virtual storage. XES performs the required page fixing to fix the buffers in real storage while the cache or list request transfers data to or from the coupling facility.

This includes storage obtained from pageable subpools, disabled reference (DREF) storage, and may include storage that has the potential to become pageable during the processing of a request. (An example is address space storage owned by any swappable address space, for which a PGSER FIX has been successfully processed, but for which the owning address space gets swapped during processing of a cache or list request.) This does not include implicitly non-pageable storage (for example, storage obtained from non-pageable subpools).

The system takes responsibility for managing binds to central storage for the duration of the cache or list request, regardless of what address space owns the storage or whether the storage-owning address space is swappable or nonswappable. The storage can be owned by any address space.

High shared virtual storage areas (above 2GB) may not be used.

#### **NO**

Specify this option to indicate that the BUFFER or BUFLIST buffers reside in non-pageable virtual storage. XES does not page fix the buffers in real storage.

This includes implicitly non-pageable storage areas (for example, storage obtained from non-pageable subpools), and may include storage that has the potential to become pageable during the processing of a request (An example is address space storage owned by any swappable address space, for which a PGSER FIX has been successfully processed, but for which the owning address space gets swapped-out during processing of a cache or list request.)

The system takes responsibility for managing binds to central storage for the duration of the cache or list request, if and only if the non-pageable storage is owned by either the requester's address space or the connector's address space. If the storage is owned by any other address space, then the invoker is responsible for ensuring that the virtual storage remains non-pageable for the duration of the request (including the case in which the storage is owned by a swappable address space that is swapped during processing of an IXLCACHE or IXLLIST request). Subject to this consideration, the storage can be owned by any address space. See [z/OS MVS Programming: Sysplex Services Guide](#).

**,PLISTVER=IMPLIED\_VERSION**

**,PLISTVER=MAX**

**,PLISTVER=plistver**

Use this input parameter to specify the version of the macro. See [“Understanding IXLLIST Version Support” on page 965](#) for a description of the options available with PLISTVER.

**,REQDATA=NO\_REQDATA**

**,REQDATA=reqdata**

Use this input parameter with MODE=SYNCEXIT or MODE=ASYNCEXIT to pass any data you choose to the complete exit. The exit will get control only if the request is processed asynchronously.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the data to be passed to the complete exit.

**,REQECB=reqecb**

Use this output parameter with either MODE=SYNCECB or MODE=ASYNCECB to specify the address of an ECB, which is to be posted when the request completes if the request was processed asynchronously.

Before coding REQECB, you must ensure that:

- You initialize the ECB before you issue the request.
- The ECB resides in either common storage or the home address space where IXLCONN was issued.
- Any tasks that wait for the ECB to be posted reside in the home address space where IXLCONN was issued.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that contains the address of the ECB to be posted when the request completes. The ECB must be aligned on a fullword boundary.

**,REQID=NO\_REQID****,REQID=reqid**

Use this input parameter to specify a user-defined request identifier to be associated with the request. You can specify this request identifier on the IXPURGE macro to cancel a request that has not yet been processed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the user-defined request identifier.

**,REQTOKEN=reqtoken**

Use this output parameter with either MODE=SYNCTOKEN or MODE=ASYNCTOKEN to specify the address of a storage area to receive the request token that is returned when the request will be processed asynchronously. This token, which uniquely identifies the request, must be used as input to the IXLFCOMP macro, which you use to determine if the request has completed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field where the system will put the request token.

**,RETCODE=retcode**

Use this output parameter to specify a field to contain the return code. (The return code is also returned in register 15.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the return code when the request has completed.

**,RSNCODE=rsncode**

Use this output parameter to specify a field to contain the reason code returned, if applicable. (The reason code is also returned in register 0.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the reason code (if any) when the request has completed.

**,STARTINDEX=startindex**

Use this input parameter to specify the index number of the first IXLYMSRI entry to be processed. The valid range is from 1 to the ENDINDEX value.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a halfword field that contains the index of the first entry to be processed.

## ABEND Codes

---

Abend X'026' (See *z/OS MVS System Codes* for more information on this abend.)

## Return and Reason Codes

---

When the system returns control to your program:

- GPR 15 (and RETCODE, if specified) contains a return code.
- If the return code is not zero, GPR 0 (and RSNCODE, if specified) contains a reason code.

**Note:** The return and reason codes will also be put into the answer area (mapped by IXLYLAA), if there is one, when the request completes.

The IXLYCON macro provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

<b>0</b>	IXLRETCODEOK
<b>4</b>	IXLRETCODEWARNING
<b>8</b>	IXLRETCODEPARMERROR
<b>C</b>	IXLRETCODEENVERROR
<b>10</b>	IXLRETCODECOMPERROR

The following table contains hexadecimal return and reason codes, the equate symbols associated with each reason code, and the meaning and suggested action for each return and reason code.

Table 65. Return and Reason Codes for IXLLIST REQUEST=MONITOR_SUBLISTS Macro		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0	None.	<p><b>Meaning:</b> If you specified a MODE value of ASYNCECB, ASYNCEXIT, ASYNCTOKEN, or ASYNCSNORESPONSE, the request has been successfully initiated. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed. For any other MODE value, the request has successfully completed.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=ASYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=ASYNCEXIT, the connection's complete exit will be given control when the request has completed.</li> <li>• If you specified MODE=ASYNCTOKEN, use REQTOKEN when calling IXLFCOMP to determine when the request has completed.</li> </ul>

Table 65. Return and Reason Codes for IXLLIST REQUEST=MONITOR\_SUBLISTS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	xxxx0402	<p><b>Equate Symbol:</b> IXLRNSNCODEASYNCH</p> <p><b>Meaning:</b> The request could not be processed synchronously. It will be processed asynchronously. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed.</p> <ul style="list-style-type: none"> <li>• If MODE=SYNCECB was specified, the ECB specified for REQECB will be posted when the request has finished.</li> <li>• If MODE=SYNCEXIT was specified, the connection's complete exit will be given control when the request has finished.</li> <li>• If MODE=SYNCTOKEN was specified, an asynchronous request token was returned in the area specified by REQTOKEN. This request token may be specified on the IXLFCOMP macro to determine when the request has finished.</li> </ul> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=SYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=SYNCEXIT, the connection's complete exit will be given control when the request has completed.</li> <li>• If you specified MODE=SYNCTOKEN, use REQTOKEN when calling IXLFCOMP to determine when the request has completed.</li> </ul>
4	xxxx0409	<p><b>Equate Symbol:</b> IXLRNSNCODETIMEOUT</p> <p><b>Meaning:</b> The request completed prematurely because it exceeded the coupling facility model-dependent time-out criteria. The following information has been returned in the answer area:</p> <ul style="list-style-type: none"> <li>• The number of event monitor controls in use (field LAAMNSLS_EMCCNT)</li> <li>• The maximum number of event monitor controls for the structure (field LAAMNSLS_MAXEMCCNT)</li> <li>• The index of the next IXLYMSRI entry to be processed (field LAAMNSLS_FAILINDEX).</li> </ul> <p><b>Action:</b> After processing all returned data, reissue the request beginning with the first unprocessed entry. For more information about premature completion, see <a href="#">z/OS MVS Programming: Sysplex Services Guide</a>.</p>

Table 65. Return and Reason Codes for IXLLIST REQUEST=MONITOR\_SUBLISTS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0801	<p><b>Equate Symbol:</b> IXLRSNCODEBADPARMLIST</p> <p><b>Meaning:</b> Program error. The parameter list for this request is not addressable. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Verify that:</p> <ul style="list-style-type: none"> <li>• The parameter list address is uncorrupted.</li> <li>• The parameter list is addressable in the caller's primary address space.</li> <li>• If you are issuing this macro while disabled, the parameter list resides in either page-fixed or DREF storage.</li> <li>• If you are issuing this macro in AR-mode and you specified the parameter list address using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If you are issuing this macro in AR-mode, SYSSTATE ASCENV=AR must be issued before issuing this macro.</li> </ul>
8	xxxx0804	<p><b>Equate Symbol:</b> IXLRSNCODEBADVERSIONNUM (or IXLRSNCODEBADVERSION#)</p> <p><b>Meaning:</b> The version number in the macro parameter list is not compatible with the level of XES currently being used. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that your program did not overlay the parameter list storage.</li> <li>• Verify that your program was assembled with the correct macro library for the release of MVS on which your program is running.</li> </ul>



Table 65. Return and Reason Codes for IXLLIST REQUEST=MONITOR\_SUBLISTS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx080A	<p><b>Equate Symbol:</b> IXLRSNCODEBADCONTOKEN</p> <p><b>Meaning:</b> Program error. The specified connect token (CONTOKEN) was not valid for one of the following reasons:</p> <ol style="list-style-type: none"> <li>1. The user with the connection identifier represented by the token has disconnected from the structure.</li> <li>2. The connector's task (the task that issued IXLCONN) ended.</li> <li>3. The specified token is not the token that was returned from IXLCONN.</li> <li>4. The request was issued from an address space other than the address space in which IXLCONN was issued.</li> <li>5. The connect token was invalidated during rebuild.</li> <li>6. The connect token was invalidated by XES.</li> </ol> <p><b>Note:</b> The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Take the action with the corresponding meaning.</p> <ol style="list-style-type: none"> <li>1. Discontinue use of the structure. Perform recovery and cleanup for the structure and check the protocol for the use of the structure to determine why you did not know of the disconnection.</li> <li>2. Discontinue use of the structure. Perform recovery and cleanup for the structure and check the protocol for the use of the structure to determine why you did not know that the task ended.</li> <li>3. The contoken is returned in the CONACONTOKEN field of the answer area specified on the IXLCONN request.</li> <li>4. Issue your request from the same address space the IXLCONN was issued in.</li> <li>5. Wait for the rebuild to complete, and try again.</li> <li>6. Discontinue use of the structure. Perform recovery and cleanup for the structure.</li> </ol>
8	xxxx0824	<p><b>Equate Symbol:</b> IXLRSNCODEWRONGSTRTYPE</p> <p><b>Meaning:</b> Program error. The connection specified by CONTOKEN is not to a list structure.</p> <p><b>Action:</b> Specify the appropriate CONTOKEN value, and rerun your program. The CONTOKEN is returned in the answer area provided on the IXLCONN invocation. The type of structure built is passed on the TYPE parameter of the IXLCONN macro.</p>

Table 65. Return and Reason Codes for IXLLIST REQUEST=MONITOR\_SUBLISTS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx082B	<p><b>Equate Symbol:</b> IXLRNCOEBADIDINDEX</p> <p><b>Meaning:</b> Program error. The value specified for either STARTINDEX or ENDINDEX was not valid. No entries were processed. The index of the first entry that was not processed is returned in the answer area.</p> <p><b>Action:</b> Ensure that the STARTINDEX and ENDINDEX values are valid and resubmit the request.</p>
8	xxxx0833	<p><b>Equate Symbol:</b> IXLRNCOEBADPGBLATTR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list is specified as being pageable (PAGEABLE=YES), but is not.</p> <p><b>Action:</b> Change the buffer area(s) to pageable storage, or specify PAGEABLE=NO. See the PAGEABLE parameter description for specification instructions.</p>
8	xxxx0834	<p><b>Equate Symbol:</b> IXLRNCOEBADNONPGBLATTR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list, is specified as being nonpageable (PAGEABLE=NO), but is either pageable or not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The type of storage in the buffer area corresponds to the value you specified on the PAGEABLE parameter. See the PAGEABLE parameter description for specification instructions.</li> <li>• The correct buffer address was used.</li> <li>• The buffer area was not previously freed.</li> <li>• If you are calling IXLLIST while disabled, the buffers must reside in either page-fixed or DREF storage.</li> <li>• The buffer areas were allocated in a storage key that matches the key specified by the BUFSTGKEY parameter, or, if BUFSTGKEY is omitted, the caller's PSW key.</li> <li>• If BUFLIST was specified and your program is running in AR-mode: <ul style="list-style-type: none"> <li>– If the BUFFER or BUFLIST parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>– The BUFALET specification is correct.</li> <li>– You specified SYSSTATE ASCENV=AR before issuing the IXLLIST macro.</li> </ul> </li> </ul>

Table 65. Return and Reason Codes for IXLLIST REQUEST=MONITOR\_SUBLISTS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0835	<p><b>Equate Symbol:</b> IXLRNCODEBADDATAADDR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list, is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The correct buffer address was used.</li> <li>• The buffer area was not previously freed.</li> <li>• The buffer area was allocated in a storage key that matches the key specified using the BUFSTGKEY parameter, or, if BUFSTGKEY is omitted, the caller's PSW key.</li> <li>• If BUFLIST was specified and your program is running in AR mode: <ul style="list-style-type: none"> <li>– The BUFALET specification is correct.</li> <li>– You specified SYSSTATE ASCENV=AR before issuing the macro.</li> </ul> </li> </ul>
8	xxxx0836	<p><b>Equate Symbol:</b> IXLRNCODEBADREALADDR</p> <p><b>Meaning:</b> Program error. Real storage addresses were provided in the BUFLIST list, but one of the buffers is not addressable in central storage.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that BUFADDRTYPE was specified as you intended.</li> <li>• Ensure that the buffer addresses specified by BUFLIST are valid.</li> </ul>
8	xxxx0838	<p><b>Equate Symbol:</b> IXLRNCODEBADANSAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by ANSAREA is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The answer area address specified by ANSAREA is valid.</li> <li>• If the caller is running in AR-mode and the ANSAREA parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• The address specified for ANSAREA is addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are calling IXLLIST while disabled, ANSAREA must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLLIST in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLIST macro.</li> </ul>

Table 65. Return and Reason Codes for IXLLIST REQUEST=MONITOR\_SUBLISTS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0839	<p><b>Equate Symbol:</b> IXLRSNCODEBADREQTOKENAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by REQTOKEN is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The request token area specified by REQTOKEN is valid.</li> <li>• If the caller is running in AR-mode and the REQTOKEN parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If you are calling IXLLIST while disabled, REQTOKEN must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLLIST in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLIST macro.</li> </ul>
8	xxxx083D	<p><b>Equate Symbol:</b> IXLRSNCODEBADANSLEN</p> <p><b>Meaning:</b> Program error. The length of the answer area, as specified by ANSLEN, is not sufficient to contain answer area information. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Increase the size of the answer area, and rerun your program. The minimum size of the ANSAREA is in the LAA_LEN field of the IXLYLAA mapping macro.</p>
8	xxxx0847	<p><b>Equate Symbol:</b> IXLRSNCODEBADLISTNUMBER</p> <p><b>Meaning:</b> Program error. The specified LISTNUM value exceeds the number of lists for the structure.</p> <p><b>Action:</b> Correct LISTNUM to specify a list number that exists in the structure. The number of lists allocated for a structure is determined on the LISTHEADERS parameter of the IXLCONN macro. The list numbers go from 0 to n-1, where n is the number of lists specified.</p>
8	xxxx084A	<p><b>Equate Symbol:</b> IXLRSNCODENOKEYS</p> <p><b>Meaning:</b> Program error. The structure does not support the use of entry keys. The IXLLIST request type either required the structure to support entry keys or designated a list entry or list position by list number and entry key.</p> <p><b>Action:</b> Ensure that you are connected to the intended structure. The use of keys for a structure is determined by the REFOPTION keyword on the IXLCONN macro.</p>
8	xxxx0851	<p><b>Equate Symbol:</b> IXLRSNCODENOSUSPENDISABLE</p> <p><b>Meaning:</b> Program error. The request failed because MODE=SYNCSUSPEND was specified, but the caller is disabled and cannot be suspended.</p> <p><b>Action:</b> Either specify another MODE value, or become enabled (release the CPU lock), then reissue the request.</p>

Table 65. Return and Reason Codes for IXLLIST REQUEST=MONITOR\_SUBLISTS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0852	<p><b>Equate Symbol:</b> IXLRSNCODENOLISTVECTOR</p> <p><b>Meaning:</b> Program error. The request failed because no local vector for monitoring lists or the user's event queue exists for this connection. Either a list notification vector for monitoring lists or the user's event queue was not requested for this connection or the list notification vector has been deleted.</p> <p><b>Action:</b> Either you are not connected to this structure or the VECTORLEN parameter was not specified when the IXLCONN was done for this structure.</p>
8	xxxx0864	<p><b>Equate Symbol:</b> IXLRSNCODEBADBUFSIZE</p> <p><b>Meaning:</b> Program error. The size of the BUFFER area or the buffer areas specified by BUFLIST is not large enough to contain the data being read. No data is returned.</p> <p><b>Action:</b> If more space is available, specify a larger buffer size, and reissue the request.</p>
8	xxxx0865	<p><b>Equate Symbol:</b> IXLRSNCODEBADBUFSPEC</p> <p><b>Meaning:</b> Program error. There is an error in the buffer specification.</p> <p><b>Action:</b> Check the following:</p> <ul style="list-style-type: none"> <li>• The requirements for BUFLIST or BUFFER.</li> <li>• Buffer pointer(s) in the BUFLIST.</li> <li>• Buffer boundaries.</li> </ul>
8	xxxx0866	<p><b>Equate Symbol:</b> IXLRSNCODEBADBUFKEY</p> <p><b>Meaning:</b> Program error. The buffer storage key specified by BUFSTGKEY is incorrect or BUFSTGKEY was not specified and the PSW key does not match the key of the buffers.</p> <p>The data cannot be stored in the specified buffer area.</p> <p><b>Action:</b> Check the following:</p> <ul style="list-style-type: none"> <li>• Determine if the key of the storage being used for the buffers is different from the PSW key.</li> <li>• If BUFSTGKEY was specified, verify that the key was put in the correct bits (see the explanation of this parameter for more information).</li> <li>• If you are calling IXLLIST in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLIST macro.</li> </ul>
8	xxxx0867	<p><b>Equate Symbol:</b> IXLRSNCODEBADBUFLIST</p> <p><b>Meaning:</b> Program error. The 128-byte storage area specified by BUFLIST is not addressable.</p> <p><b>Action:</b> Ensure that the address specified by BUFLIST is valid.</p>

Table 65. Return and Reason Codes for IXLLIST REQUEST=MONITOR\_SUBLISTS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0880	<p><b>Equate Symbol:</b> IXLRNCODEBADMOSVECTOR</p> <p><b>Meaning:</b> Program error. The storage area specified by MOSVECTOR is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The correct storage area address was used.</li> <li>• If you are running in AR-mode and the MOSVECTOR was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If you are calling IXLLIST in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLIST macro.</li> </ul>
C	xxxx0C06	<p><b>Equate Symbol:</b> IXLRNCODENOCNN</p> <p><b>Meaning:</b> Environmental error. This system does not have connectivity to the coupling facility that contains the list structure. Possible reasons for this are:</p> <ul style="list-style-type: none"> <li>• The operator issued VARY PATH,OFFLINE.</li> <li>• The operator issued CONFIG CHP,OFFLINE.</li> <li>• Hardware errors to the coupling facility.</li> <li>• Facility or path failure to the coupling facility.</li> </ul> <p><b>Action:</b> Begin rebuilding the structure on a different coupling facility, or disconnect from the structure.</p>
C	xxxx0C13	<p><b>Equate Symbol:</b> IXLRNCODEREQPURGED</p> <p><b>Meaning:</b> Environmental error. The request was purged prior to completion of the request. Possible reasons include:</p> <ul style="list-style-type: none"> <li>• The connector failed.</li> <li>• The connector disconnected.</li> <li>• The requestor failed.</li> <li>• The request was purged by IXLPURGE.</li> <li>• Requests were purged when the connector provided an IXLEERSP response for the Rebuild Stop or Rebuild Cleanup event.</li> <li>• The secondary address space was no longer valid.</li> </ul> <p><b>Action:</b> None if this is expected. Otherwise, determine why the connector failed.</p>

Table 65. Return and Reason Codes for IXLLIST REQUEST=MONITOR\_SUBLISTS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C14	<p><b>Equate Symbol:</b> IXLRNCODESTATUSUNKNOWN</p> <p><b>Meaning:</b> Environmental error. The request has completed, but the final disposition of the request cannot be determined. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the validity of your data by comparing the expected results with what is in the coupling facility.</li> <li>• Consider running rebuild processing to move the structure to attempt to protect it from further errors of this kind.</li> </ul>
C	xxxx0C17	<p><b>Equate Symbol:</b> IXLRNCODESTRFULL</p> <p><b>Meaning:</b> Environmental error. The request attempted to create a new event monitor controls (EMC) object, but the structure is full and cannot accommodate any more EMCs.</p> <p><b>Action:</b> Determine why the structure is full. You should be monitoring the usage of the structure every time a REQUEST=MONITOR_SUBLISTS is done (LAAMNSLS_EMCCNT is the total count of EMCs in use in the list structure and LAAMNSLS_MAXEMCCNT is the maximum number of EMCs for the list structure.). This data should be evaluated periodically to ensure structure resources are being used efficiently.</p> <p>Field LAAMNSLS_FAILINDEX contains the index of the entry in IXLYMSRI that experienced the structure full condition. IXLYMSRI entries prior to LAAMNSLS_FAILINDEX were processed successfully.</p>
C	xxxx0C25	<p><b>Equate Symbol:</b> IXLRNCODESTRFAILURE</p> <p><b>Meaning:</b> Environmental error. The list structure failed prior to completion of the request.</p> <p><b>Action:</b> Either rebuild or disconnect from the structure.</p>
C	xxxx0C68	<p><b>Equate Symbol:</b> IXLRNCODEBADREQCFLEVEL</p> <p><b>Meaning:</b> Environmental error. The request type is not permitted for the level of coupling facility in which the target structure is allocated.</p> <p><b>Action:</b> Either disconnect from the structure (using IXLDISC) or initiate a rebuild of the structure, if allowed (using IXLREBLD) in a coupling facility of the correct CFLEVEL.</p>
C	xxxx0CA0	<p><b>Equate Symbol:</b> IXLRNCODEQUIESCEDSUSPENDFAIL</p> <p><b>Meaning:</b> Environmental error. The request is failed because the structure is quiesced for a system-managed process and SUSPEND=FAIL is specified on the IXLCONN.</p> <p><b>Action:</b> None, if this is expected.</p>

Table 65. Return and Reason Codes for IXLLIST REQUEST=MONITOR\_SUBLISTS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxxFFFF	<b>Equate Symbol:</b> IXLRSNCODENOTAVAILABLE <b>Meaning:</b> Environmental error. XES functions are not available. This could occur because the coupling facility hardware necessary to provide XES functions is not present. <b>Action:</b> Re-IPL the system, or follow your particular failure management protocol.
10	xxxx10xx	<b>Meaning:</b> System error. XES processing failure. The state of the involved structure and the disposition of the request are unpredictable. The answer area (ANSAREA) fields do not contain valid information. <b>Action:</b> Contact the IBM support center.



## Chapter 66. IXLLIST REQUEST=MOVE

### Description

A MOVE request allows you to perform one of the following types of operations:

- Move an existing list entry from one list to another list or within the same list. In addition to moving the entry, you can also read or update that entry's data.
- Create a new list entry (if the entry does not exist on a specified list), place it on either the same or another list, and write data to it.

Use the DATAOPER parameter to specify the operation to be performed:

- **DATAOPER=NONE** Only the existing entry is moved.
- **DATAOPER=READ** The existing entry is moved, and its data is read.
- **DATAOPER=WRITE**
  - **ENTRYTYPE=OLD** The existing entry is moved, and its data is updated.
  - **ENTRYTYPE=ANY** If the designated entry exists, it is moved and its data is updated. If the entry does not exist, a new entry is created and data is written to it.

If this request creates a new entry (**ENTRYTYPE=ANY** must be specified) you must provide the list number (**MOVETOLIST**) on which the new entry will be placed. You can also provide additional identification for the new entry by specifying one of the following:

- Entry name (**ENTRYNAME**)
- Entry key (**ENTRYKEY**)

**Note:** You cannot assign an entry identifier (**ENTRYID**) to a new entry; the system assigns entry identifiers to all new entries.

A MOVE request allows you to read (**DATAOPER=READ**) or write (**DATAOPER=WRITE**) entry data and/or adjunct data. Entry data is read from or written to the **data entry** segment of an entry, while the adjunct data is read from or written to the **adjunct data area** segment. The following describes from where each type of data is written (if **DATAOPER=WRITE**) or where each type of data is returned (if **DATAOPER=READ**):

- A buffer (**BUFFER**) or list of buffers (**BUFLIST**) holds entry data. If neither **BUFFER** nor **BUFLIST** is specified, no entry data is read or written.
- An adjunct area (**ADJAREA**) holds adjunct data. If **ADJAREA** is not specified, no adjunct data is read or written.

If you specify **DATAOPER=WRITE**, you can assign the number of data elements comprising the data entry segment of the existing or new entry by specifying **ELEMNUM**.

Additionally, you can perform locking functions with a MOVE request by specifying **LOCKINDEX**. The lock entry designated by **LOCKINDEX** will be operated on as specified by **LOCKOPER**.

With a coupling facility of **CFLEVEL=1** or higher, the following additional functions are supported for a MOVE request:

- You can specify that a list entry be moved only if the list authority value for the list associated with the entry is of a certain value. You specify the list authority comparison requirements using **AUTHCOMP** and **AUTHCOMPTYPE**. If the request is successful, you also can update the list authority value to a new value that you specify with **NEWAUTH**.
- Version number comparison is enhanced to allow for an additional equal-or-less-than-equal comparison operation.
- The system can assign an entry key value automatically from a list control list key value.

- There are additional list cursor options. You can update the list cursor conditionally, and you can set the list cursor to point to the current entry instead of the previous or next entry.

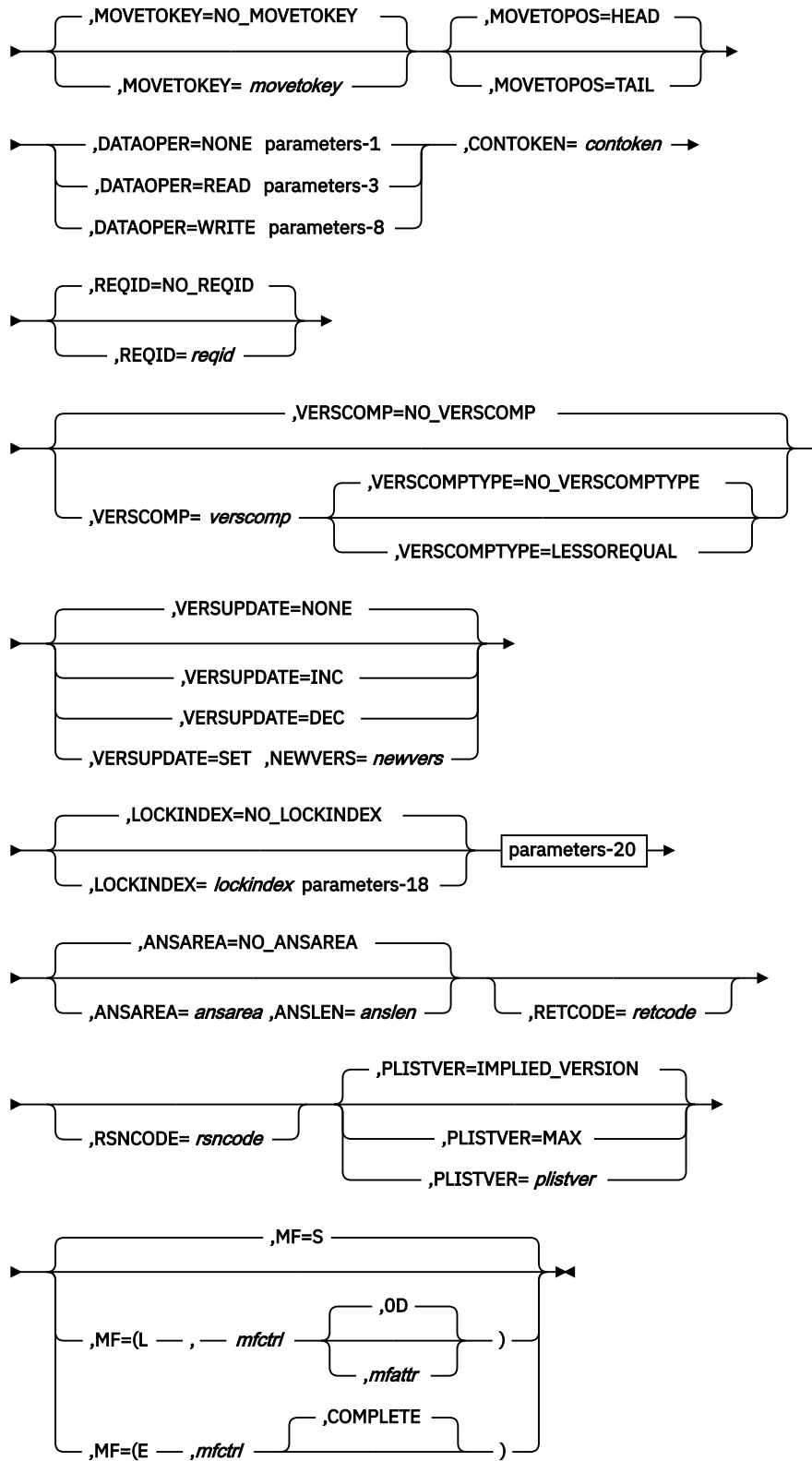
## Syntax Diagram

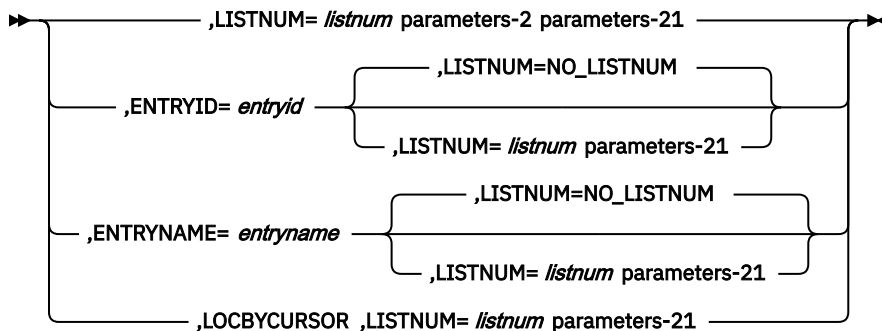
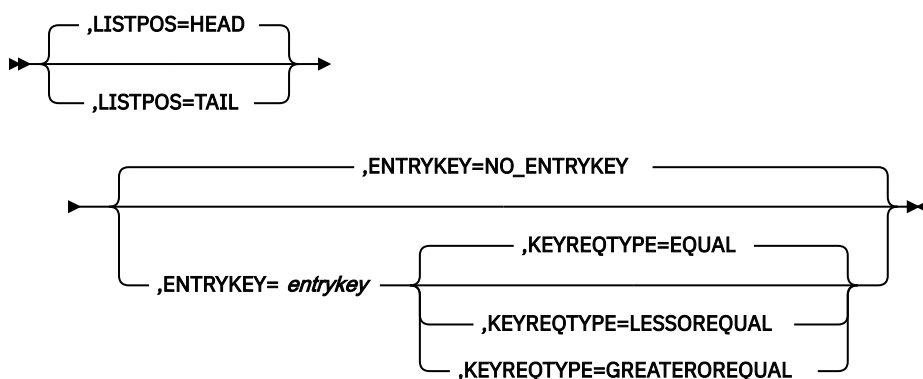
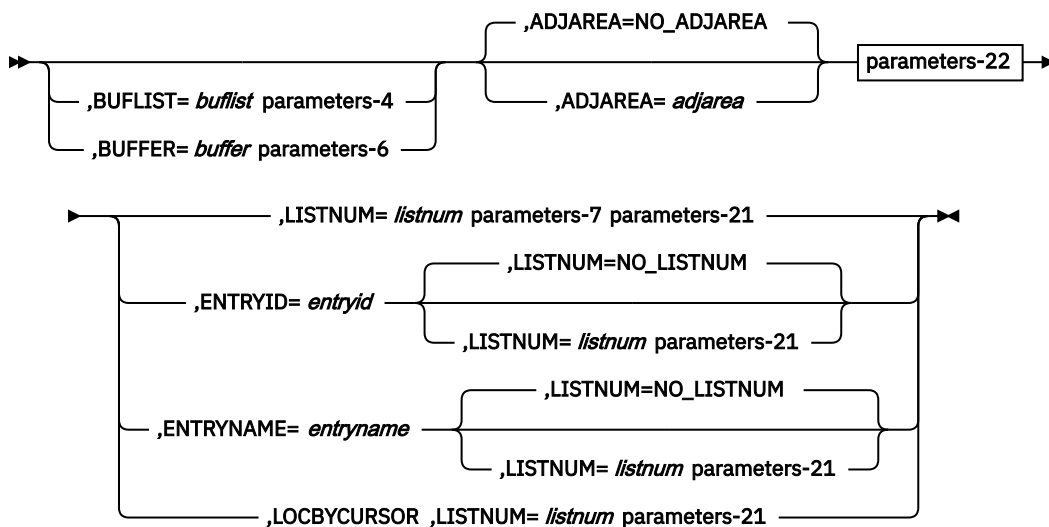
---

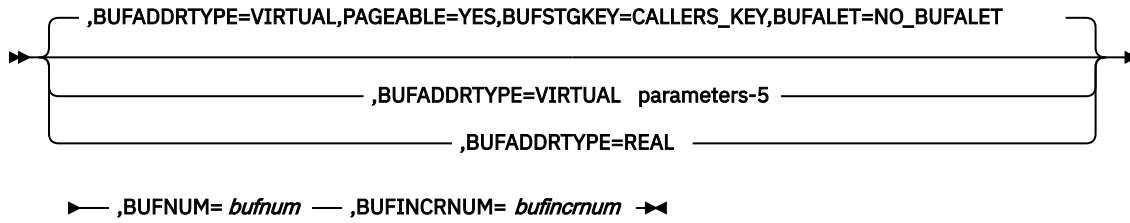
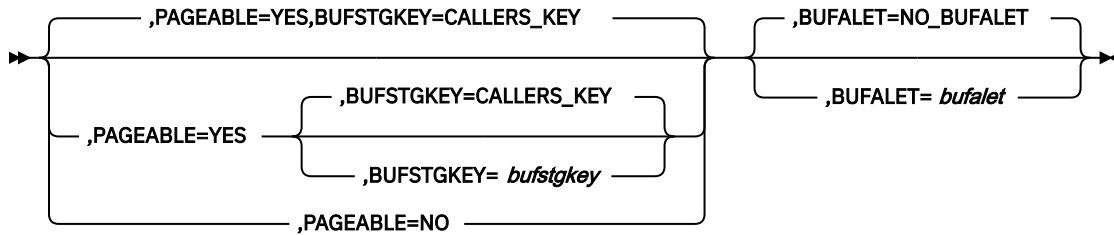
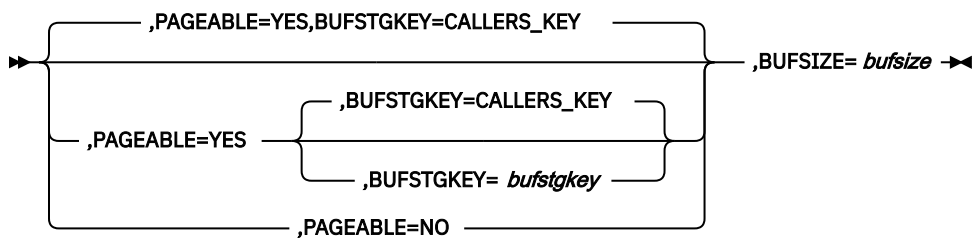
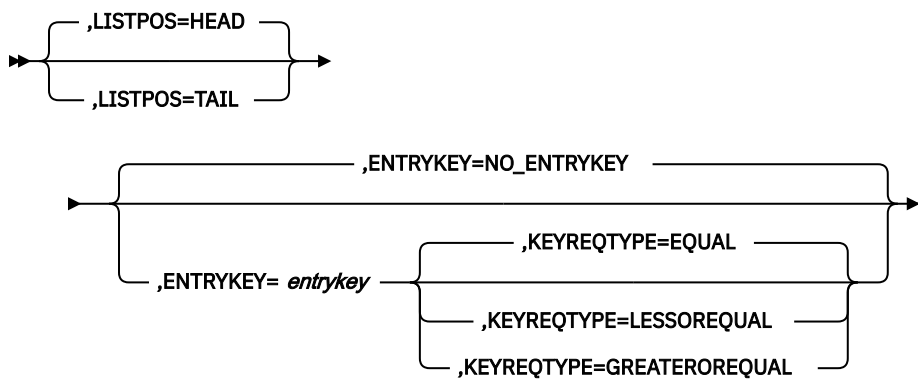
The syntax diagram for IXLLIST REQUEST=MOVE is as follows:

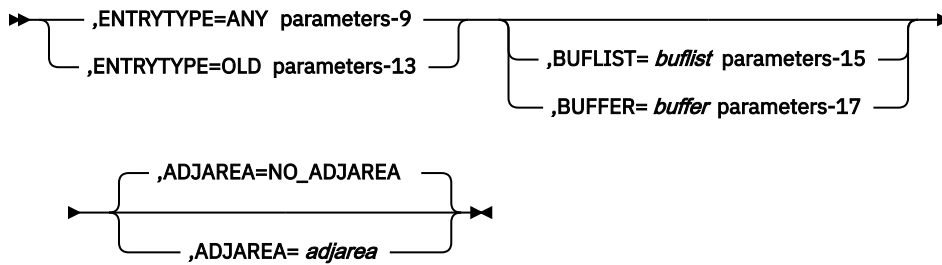
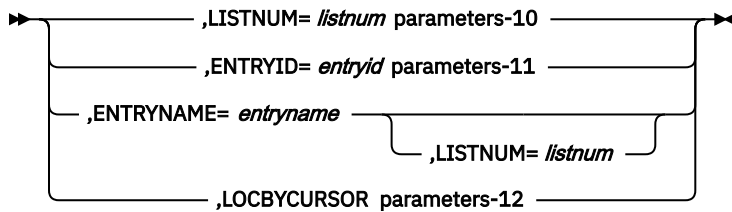
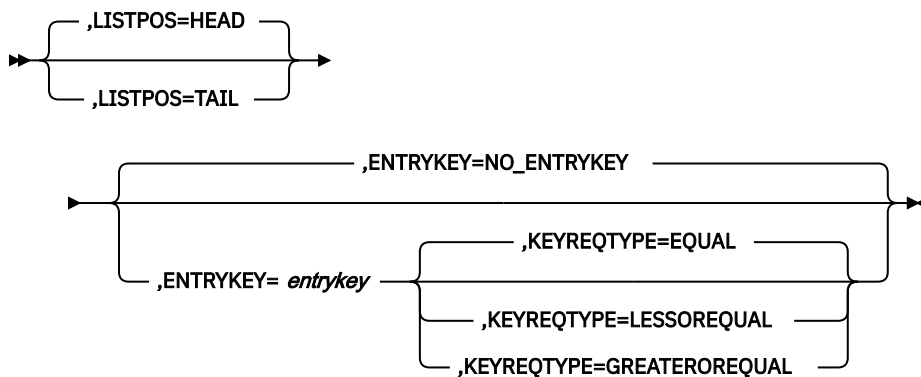
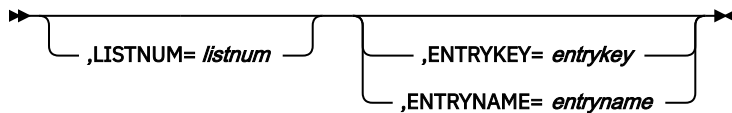
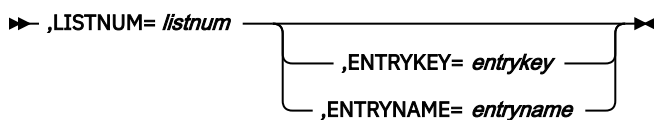
## main diagram

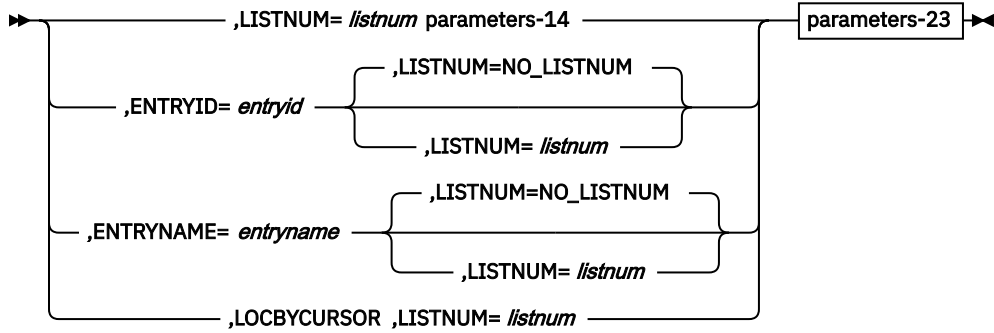
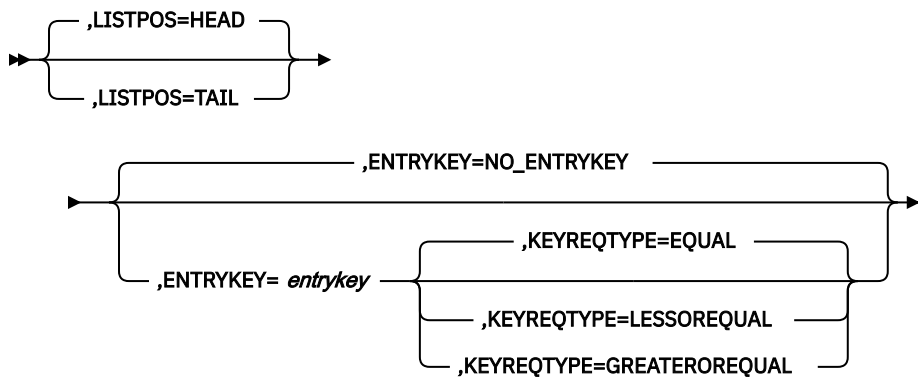
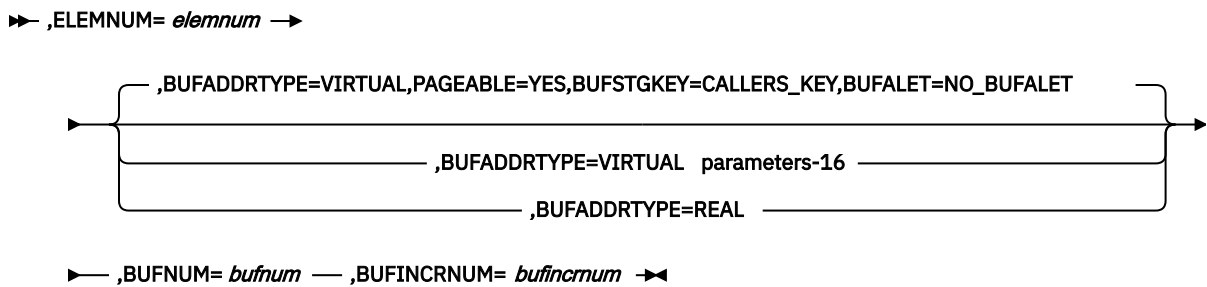
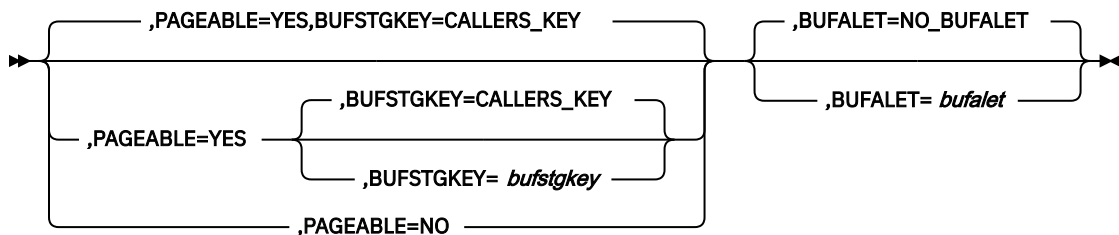
➤ IXLLIST — REQUEST=MOVE — ,MOVETOLIST= *movetolist* ➔

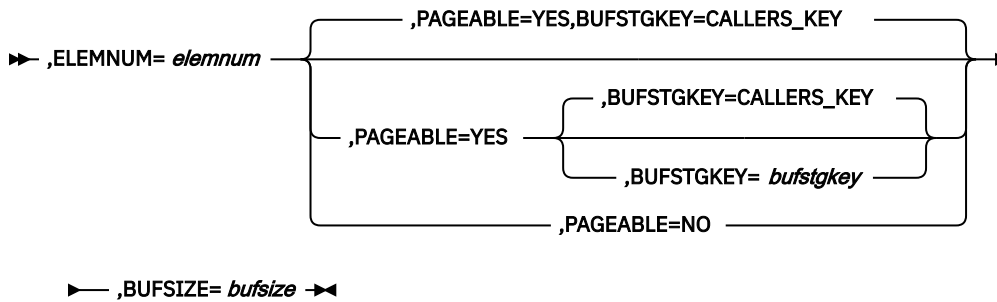
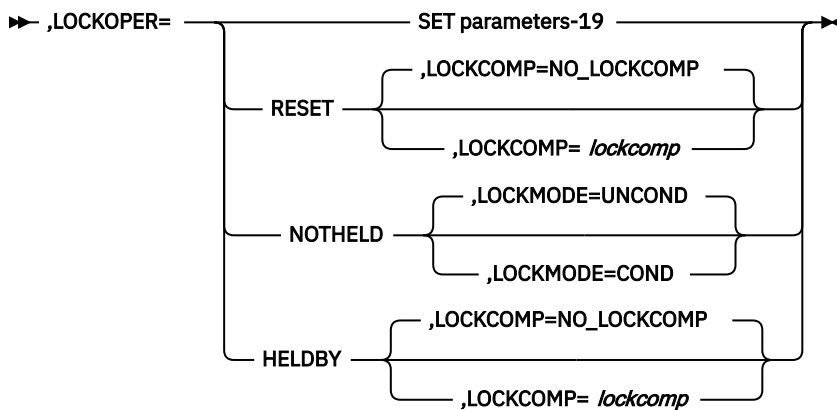
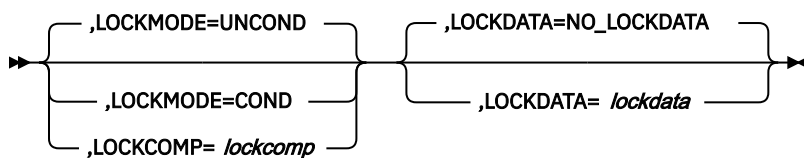


**parameters-1****parameters-2****parameters-3**

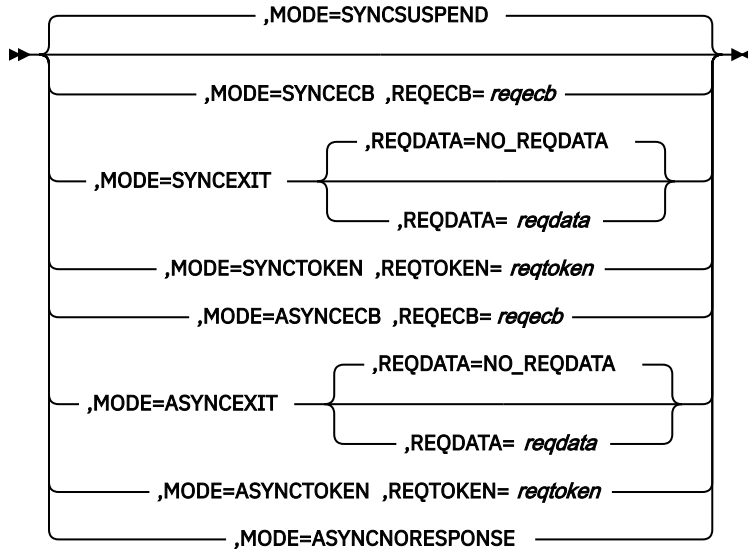
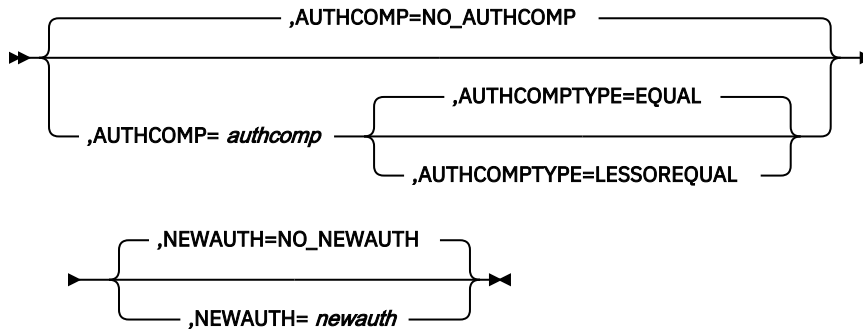
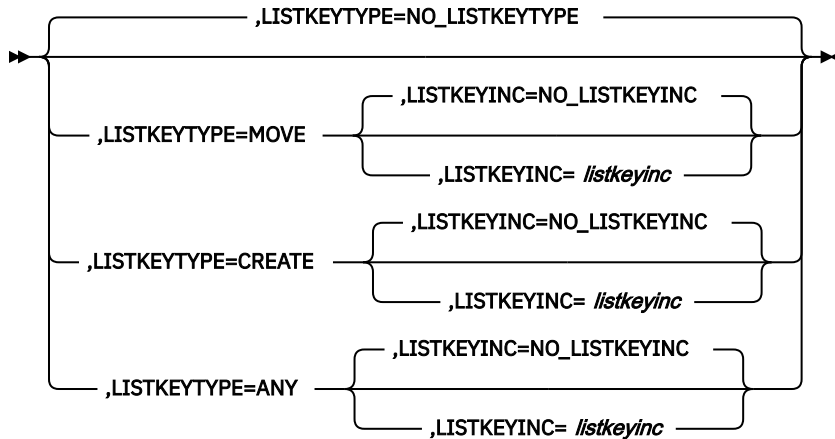
**parameters-4****parameters-5****parameters-6****parameters-7**

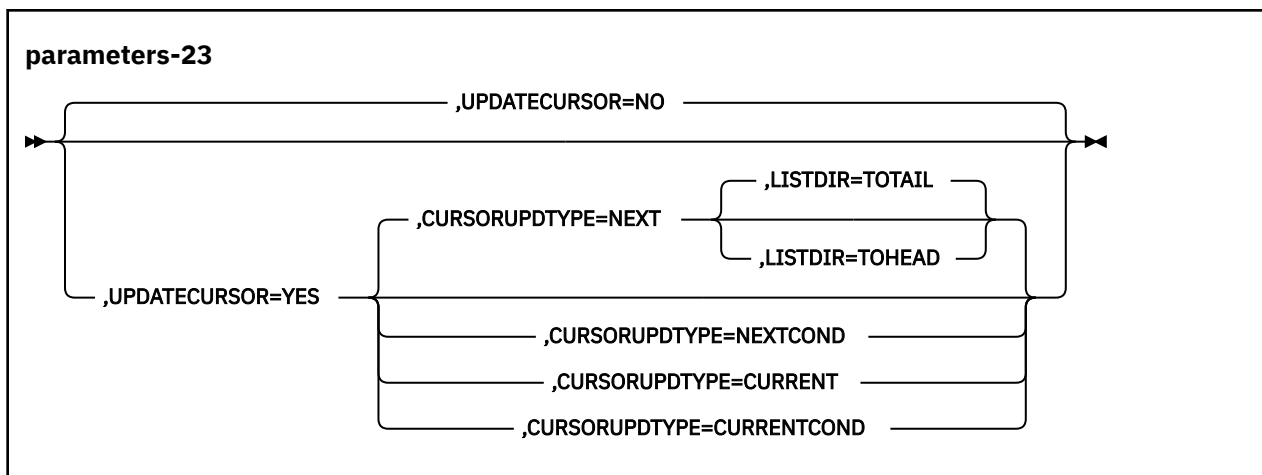
**parameters-8****parameters-9****parameters-10****parameters-11****parameters-12**

**parameters-13****parameters-14****parameters-15****parameters-16**

**parameters-17****parameters-18****parameters-19**



**parameters-20****parameters-21****parameters-22**

**Note:**

1. In the main diagram, if DATAOPER is not specified, DATAOPER=NONE is the default and you must code the required parameters shown in the | parameters-1 | fragment.
2. In the | parameters-3 | fragment, one of the following must be specified:
  - BUFLIST=*buflist*
  - BUFFER=*buffer*
  - ADJAREA=*adjarea*

In addition, ADJAREA=*adjarea* can be specified with either BUFLIST=*buflist* or BUFFER=*buffer*.
3. In the | parameters-8 | fragment, if ENTRYTYPE is not specified, ENTRYTYPE=ANY is the default and you must code the required parameters specified in the | parameters-9 | fragment.
4. If MODE=SYNCTOKEN or MODE=ASYNCTOKEN is specified, ANSAREA= *ansarea* and ANSLEN=*anslen* are required.
5. If MODE=ASYNCSNORESPONSE is specified, BUFFER, BUFLIST, and LOCKINDEX may not be specified.

## Parameter Descriptions

The parameter descriptions for REQUEST=MOVE are listed in alphabetical order. Default values are underlined:

**REQUEST=MOVE**

Use this input parameter to move an entry.

**,ADJAREA=NO\_ADJAREA****,ADJAREA=adjarea**

Use this input parameter to specify a storage area to contain the adjunct data that is read from or written to an entry.

Specify ADJAREA only for structures that support adjunct data. (Adjunct areas for a structure are established through the IXLCONN macro.)

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 64-byte area that contains or will contain the adjunct data.

**,ANSAREA=NO\_ANSAREA****,ANSAREA=ansarea**

Use this output parameter to specify an answer area to contain information returned from the request. The format of the answer area is described by mapping macro IXLYLAA.

On a successful completion (for a synchronous request: RC=0, for an asynchronous request: ECB is posted, IXLFComp indicates completion, or your complete exit ran) of the request, the following information is returned to the answer area:

- List entry controls of the existing or new entry (field LAALCTL).
- The number of list entries or elements residing on the list (field LAALISTCNT).
- The total number of allocated entries in the structure (field LAATOTALCNT).
- The total number of allocated elements in the structure (field LAATOTALELECNT).

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of an area (with a length of ANSLEN) that will contain the information that is returned by the request.

**,ANSLEN=anslen**

Use this input parameter to specify the size of the storage area specified by ANSAREA.

Check the prologue of the IXLYLAA mapping macro for the minimum required size of the answer area, or use the length field (LAA\_LEN) in the LAA.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 2-byte field that contains the size, in bytes, of the answer area.

**,AUTHCOMP=NO\_AUTHCOMP**

**,AUTHCOMP=authcomp**

Use this input parameter to specify a value to be compared to the list authority value of the list specified by LISTNUM. You must supply the LISTNUM parameter.

If the comparison does not meet the condition specified by the AUTHCOMPTYPE parameter (EQUAL or LESSOREQUAL), the request fails.

**Note:** The AUTHCOMP parameter is valid only for list structures allocated in a coupling facility with CFLEVEL=1 or higher.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12 2 to ) of the 16-byte field that contains the list authority value.

**,AUTHCOMPTYPE=EQUAL**

**,AUTHCOMPTYPE=LESSOREQUAL**

Use this input parameter specify how the list audit comparison is to be performed.

**EQUAL**

The list authority for the list specified by LISTNUM must be equal to the value specified for AUTHCOMP.

**LESSOREQUAL**

The list authority for the list specified by LISTNUM must be less than or equal to the value specified for AUTHCOMP.

**Note:** The AUTHCOMPTYPE parameter is valid only for list structures allocated in a coupling facility with CFLEVEL=1 or higher.

**,BUFADDRTYPE=VIRTUAL**

**,BUFADDRTYPE=REAL**

Use this input parameter to specify whether the buffer addresses specified in the BUFLIST list are virtual storage or real storage addresses.

**VIRTUAL**

The buffer addresses are virtual storage addresses. The virtual storage can be pageable or nonpageable. See the PAGEABLE parameter for information about managing storage binds when specifying virtual storage addresses.

**REAL**

The buffer addresses are real storage addresses.

It is the caller's responsibility to manage the binds between the data buffer virtual storage and the real storage addresses provided. The caller must ensure that the data buffer virtual storage remains bound to the real storage addresses provided until the request completes.

**,BUFALET=NO\_BUFALET****,BUFALET=*bufalet***

Use this input parameter to specify an access list entry token (ALET) to be used in referencing all of the buffers specified by BUFLIST.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 4-byte field that contains the ALET.

**,BUFFER=*buffer***

Use this output or input parameter to hold entry data to be read from or written to the designated entry. The purpose of the BUFFER area depends on the DATAOPER value you specify:

- If you specify DATAOPER=READ, the buffers hold output data—entry data that was read from an existing entry.
- If you specify DATAOPER=WRITE, the buffers hold input data—entry data to be written to an existing or new entry.

You can define the buffer size to be a total size of up to 65536 bytes. Depending on the size you select, the following restrictions apply:

- If you specify a buffer size of less than or equal to 4096 bytes, you must ensure that the buffer:
  - Is 256, 512, 1024, 2048, or 4096 bytes.
  - Starts on a 256-byte boundary.
  - Does not cross a 4096-byte boundary.
- If you specify a buffer size of greater than 4096 bytes, you must ensure that the buffer:
  - Is a multiple of 4096 bytes.
  - Is less than or equal to 65536 bytes.
  - Starts on a 4096-byte boundary.

See the BUFSIZE parameter description for defining the size of the buffer.

**Note:** You cannot code BUFFER with MODE=ASYNCRESPONSE.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an area (with a length of BUFSIZE) that contains the entry data.

**,BUFINCRNUM=*bufincrnum***

Use this input parameter to specify the number of 256-byte segments comprising each buffer in the BUFLIST list.

Valid BUFINCRNUM values are 1, 2, 4, 8, or 16, which correspond to BUFLIST buffer sizes of 256, 512, 1024, 2048, and 4096 bytes respectively.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 1-byte field that contains 1, 2, 4, 8, or 16.

**,BUFLIST=*buflist***

Use this output or input parameter to specify a list of buffers to hold entry data to be read from or written to the designated entry. The purpose of the BUFLIST buffers depend on the DATAOPER value you specify:

- If you specify DATAOPER=READ, the buffers hold output data—entry data that was read from an existing entry.
- If you specify DATAOPER=WRITE, the buffers hold input data—entry data to be written to an existing or new entry.

BUFLIST specifies a 128-byte storage area that consists of a list of 0 to 16 buffer addresses.

**The 128-byte storage area must:**

- Consist of 0 to 16 elements.
- Each element must consist of an 8-byte field in which:

- The left (high-order) four bytes are reserved and
- The right (low-order) four bytes contain the address of a buffer.

**The BUFLIST buffers must:**

- Reside in either the same address space or data space.
- Be the same size: either 256, 512, 1024, 2048, or 4096 bytes.
- Start on a 256-byte boundary and not cross a 4096-byte boundary.

**Note:** The buffers do not have to be contiguous in storage. (List services treats BUFLIST buffers as a single, contiguous buffer, even if the buffers are not contiguous.)

See the BUFNUM and BUFINCRNUM parameter descriptions for defining the number and size of buffers.

**Note:** You cannot code BUFLIST with MODE=ASYNCHRESPONSE.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 128-byte area that contains the list of buffer addresses.

**,BUFNUM=*bufnum***

Use this input parameter to specify the number of buffers in the BUFLIST list. Valid BUFNUM values are from 0 to 16. A value of zero indicates that no data is to be read into the buffers or written to the list entry.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 1-byte field that contains the number of buffers in the buffer list.

**,BUFSIZE=*bufsize***

Use this input parameter to specify the size of the BUFFER area. See the BUFFER parameter description for valid buffer sizes.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a fullword field that contains the size of the buffer (BUFFER) in bytes.

**,BUFSTGKEY=CALLERS\_KEY**

**,BUFSTGKEY=*bufstgkey***

Use this input parameter to specify a storage key that you define and use when referencing the buffers specified by BUFLIST or the buffer that is specified by BUFFER.

If you do not specify BUFSTGKEY, or if you specify BUFSTGKEY=CALLERS\_KEY, all references to one or more buffers are performed by using the caller's PSW key.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of an 8-bit field that contains the storage key in the format B'kkkkxxxx', where kkkk is the key and xxxx is ignored.

**,CONTOKEN=*contoken***

Use this input parameter to specify the connect token that was returned by the IXLCONN service. The connect token uniquely identifies your connection to the list structure, and must be specified on each IXLLIST invocation.

The connect token is available in the IXLCONN answer area that is mapped by IXLYCONA.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 16-byte field that contains the connect token.

**,CURSORUPDTYPE=NEXT**

**,CURSORUPDTYPE=NEXTCOND**

**,CURSORUPDTYPE=CURRENT**

**,CURSORUPDTYPE=CURRENTCOND**

Use this input parameter to specify how the list cursor is to be updated when UPDATECURSOR=YES is specified.

**Note:** The NEXTCOND, CURRENT, and CURRENTCOND values are valid only for list structures allocated in a coupling facility with CFLEVEL=1 or higher.

**CURSORUPDTYPE=NEXT**

Update the list cursor to point to the list entry before or after the target entry.

- For MOVE requests that specify DATAOPER=WRITE and ENTRYTYPE=ANY and that result in the creation of a new entry, the cursor for the list specified by MOVETOLIST is updated. The direction of the cursor update depends on the value that is specified for MOVETOPOS.
- For all other requests, the cursor for the source list is updated. The direction of the cursor update depends on the value that is specified for LISTPOS or LISTDIR.

**CURSORUPDTYPE=NEXTCOND**

Update the list cursor to point to the list entry before or after the target entry only if the cursor points to the target list entry, and that list entry is moved or deleted.

Set the list cursor direction with SETCURSOR on a WRITE\_LCONTROLS request.

The list cursor is reset to binary zeros if either:

- The entry is the last entry on the list and the list cursor direction is set to a head-to-tail direction.
- The entry is the first entry on the list and the list cursor is set to a tail-to-head direction.

**CURSORUPDTYPE=CURRENT**

Set the list cursor to point to the list entry processed for the request.

If the list entry is deleted or moved to another list, then the list cursor is reset to binary zeros.

**CURSORUPDTYPE=CURRENTCOND**

Set the list cursor to point to the list entry processed only if the list cursor value currently is zero and the target list entry is not deleted or moved to another list.

If the list entry is deleted or moved to another list, then the list cursor will remain binary zeros.

**,DATAOPER=NONE****,DATAOPER=READ****,DATAOPER=WRITE**

Use this input parameter to specify an operation to be performed on the designated entry.

**NONE**

The existing entry is moved; no additional operation is performed.

**READ**

The existing entry is moved, and its entry data and/or adjunct data is read:

- If you specified BUFFER or BUFLIST, the entry data is read into whichever buffer area you have specified.
- If you specified ADJAREA, the adjunct data is read into the adjunct area.

**WRITE**

The existing entry is moved, and its entry data and/or adjunct data is updated. If this request creates a new entry, the entry data and/or adjunct data is written to the new entry:

- If you specified BUFFER or BUFLIST, the entry data is written to the existing or new entry.
- If you specified ADJAREA, the adjunct data is written to the existing or new entry.

**,ELEMNUM=elemnum**

Use this input parameter to specify the number of elements to be allocated to the existing or new data entry. Valid ELEMNUM values are from zero to the MAXELEMNUM value that was specified on the IXLCONN macro.

Considerations for specifying ELEMNUM are:

- If this request creates a new entry, the number of elements is set to the ELEMNUM value.
- If the entry exists, the number of elements is updated to the ELEMNUM value specified.

**Note:** If the data from the buffers exceeds the amount of space in the elements, the data is truncated. If the data from the buffers is less than the amount of space in the elements, the remaining space is padded with binary zeros.

If you specify an ELEMNUM value of zero:

- No entry data is written to the new entry.
- Any entry data in the existing entry will be deleted.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 1-byte field that contains the number of elements.

#### **,ENTRYID=entryid**

Use this input parameter to designate the entry identifier of an existing entry to be moved. ENTRYID applies only to locating an existing entry; it does not determine the identifier of a new entry should one be created.

If an entry with this identifier does not exist (and ENTRYTYPE=ANY is specified), a new entry is created on the MOVETOLIST list with a unique entry identifier assigned by list services. (The entry identifier you specify will be ignored.)

When ENTRYID is specified with ENTRYNAME, ENTRYKEY, or LISTPOS, list services uses only ENTRYID to determine if the entry already exists.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 12-byte field that contains the entry identifier.

#### **,ENTRYKEY=NO\_ENTRYKEY**

#### **,ENTRYKEY=entrykey**

Use this input parameter to either:

- Designate the entry key of an existing entry to be moved.
- Assign an entry key to a new entry to be created, if MOVETOKEY is not specified. If MOVETOKEY is specified, the MOVETOKEY key is assigned to the new entry.

The existing entry must reside on the LISTNUM list. The existing entry will be placed on the MOVETOLIST list. If a new entry is created it will be placed on the MOVETOLIST list.

If DATAOPER=NONE or DATAOPER=READ is specified, the specified entry key designates an existing entry.

If DATAOPER=WRITE is specified, the specified entry key designates an existing entry or can be assigned to a new entry. See *z/OS MVS Programming: Sysplex Services Guide* for information about how entry keys are assigned to existing or created entries.

Specify ENTRYKEY only for structures that support keyed entries.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-byte field that contains the entry key.

#### **,ENTRYNAME=entryname**

Use this input parameter to either:

- Designate an existing entry to be moved.
- Assign an entry name to a new entry to be created.

Both existing and new entries will be placed on the MOVETOLIST list.

If DATAOPER=NONE or DATAOPER=READ, the specified entry name designates an existing entry.

If DATAOPER=WRITE is specified, whether the specified entry name designates an existing entry or is assigned to a new entry depends on the following:

- If either of the following is true, the specified entry name designates an existing entry:
  - ENTRYTYPE=OLD, and the entry already exists.
  - ENTRYTYPE=ANY, neither ENTRYID nor LOCBYCURSOR is also specified, and the entry already exists.
- If ENTRYTYPE=ANY and the entry does not already exist, a new entry is created with the ENTRYNAME name assigned to it.

Specify ENTRYNAME only for structures that support named entries. Each entry name is unique within the structure.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-byte field that contains the entry name.

**,ENTRYTYPE=ANY**

**,ENTRYTYPE=OLD**

Use this input parameter to specify whether you want to either:

- Update an existing entry or create a new entry
- Update an existing entry

**ANY**

Either update an existing entry or create a new entry. If the entry exists, its data is replaced with the new data. If the entry does not exist, one will be created with the new data.

**OLD**

Update an existing entry. The entry must already exist, or the request fails.

**,KEYREQTYPE=EQUAL**

**,KEYREQTYPE=LESSOREQUAL**

**,KEYREQTYPE=GREATEROREQUAL**

Use this input parameter to specify how, if at all, the entry key of the existing entry to be moved can differ from the entry key specified by ENTRYKEY. KEYREQTYPE applies only to locating an existing entry; it does not determine the key of a new entry should one be created.

**EQUAL**

The entry must have a key that equals the ENTRYKEY key.

**LESSOREQUAL**

The entry must have a key that is less than or equal to the ENTRYKEY key.

**GREATEROREQUAL**

The entry must have a key that is greater than or equal to the ENTRYKEY key.

**Note:**

1. When no entries on the list meet the requirements of the KEYREQTYPE, and a new entry cannot be created (TYPE=OLD was specified), the request will be failed with a return code X'8' and reason code IXLRSCODENOENTRY, or a new list entry will be allocated, depending on the other options specified.
2. When LESSOREQUAL or GREATEROREQUAL are specified, if no entries on the list have an entry key value equal to the specified value, but entries exist with an entry key value greater than (if GREATEROREQUAL was specified), or less than (if LESSOREQUAL was specified) the entry key value specified, then the entry with an entry key value closest to the value specified will be selected. When multiple entries have the same entry key value, LISTPOS is used to resolve whether the first or last entry with the entry key value is selected.

**,LISTDIR=TOTAIL**

**,LISTDIR=TOHEAD**

Use this input parameter with UPDATECURSOR to specify the direction in which the list cursor is moved as a result of this request. See the UPDATECURSOR parameter for a full description of LISTDIR.

**,LISTKEYINC=NO\_LISTKEYINC**

**,LISTKEYINC=listkeyinc**

Use this input parameter to specify a value to be added to the list key after the entry key is set to the list key value. If the entry key is not set to the list key value, the list key value will not be changed. If the result of adding the value that is specified by LISTKEYINC to the list key value is greater than the maximum list key value, the system fails the request.

**Note:** The LISTKEYINC parameter is valid only for list structures allocated in a coupling facility with CFLEVEL=1 or higher.



**To Code:** Specify the RS-type name or address (by using a register 2 - 12 2 to ) of the fullword field that contains the value to be added to the list key after the entry key is set the list key value.

**,LISTKEYTYPE=NOLISTKEY**

**,LISTKEYTYPE=MOVE**

**,LISTKEYTYPE=ANY**

Use this input parameter to specify when the designated entry key is to be set to the current list key value. You can set the entry key to the current list key value when you move the entry.

(Set the list key and maximum list key values with a WRITE\_LCONTROLS request.)

**Note:** The LISTKEYTYPE parameter is valid only for list structures allocated in a coupling facility with CFLEVEL=1.

**LISTKEYTYPE=NOLISTKEY**

Do not set the entry key for the target list entry to the current list key value. If the list entry is moved and the list structure supports entries with keys, then the other parameters specified (such as ENTRYKEY) will determine the resultant entry key value.

**LISTKEYTYPE=MOVE**

Set the entry key for the designated list entry to the current list key value if the entry is moved by this request.

**LISTKEYTYPE=ANY**

Set the entry key for the designated list entry to the current list key value if the entry is moved or created by this request. (This parameter also applies when you WRITE an entry.)

**,LISTNUM=NO LISTNUM**

**,LISTNUM=listnum**

Use this input parameter to specify the number of the list on which the entry to be moved currently resides. Use LISTNUM in the following ways:

- With LISTDIR and/or ENTRYKEY to designate an existing entry to be moved.
- With ENTRYID or ENTRYNAME to:
  - Check if the entry exists on the list (when ENTRYTYPE=ANY) before proceeding with the request.
  - Confirm that the entry exists on the list (when ENTRYTYPE=OLD) before proceeding with the request.
- With LOCBYCURSOR to designate an existing entry to be moved.

**To Code:** Specify the RS-type name or address (using a 2 register from 2 to 12) of a 4-byte field that contains the number of the list.

**,LISTPOS=HEAD**

**,LISTPOS=TAIL**

Use this input parameter with LISTNUM to designate the existing entry to be moved. LISTPOS applies only to locating an existing entry; it does not determine where a new entry would be placed should one be created. (See the MOVETOPOS parameter for a description of where a new entry is placed.)

LISTPOS designates the entry at the HEAD or the TAIL of a list or sublist:

- If you specify ENTRYKEY to designate the entry and the list contains a sublist of one or more entries with the same key (or that satisfies the KEYREQTYPE criteria), then:
  - LISTPOS=HEAD designates the entry at the head of the sublist.
  - LISTPOS=TAIL designates the entry at the tail of the sublist.
- If you do not specify ENTRYKEY to designate the entry:
  - LISTPOS=HEAD designates the entry at the head of the list.
  - LISTPOS=TAIL designates the entry at the tail of the list.

**,LOCBYCURSOR**

Use this input parameter to designate an existing entry to be moved. The designated entry is the entry to which the LISTNUM list cursor is pointing.

Be aware that the list cursor could have been reset to zeros by a previous request that, for instance, deleted or moved the entry to which the list cursor points, or updated the list cursor after the last entry on the list had been processed. See *z/OS MVS Programming: Sysplex Services Guide* for more information about using the list cursor.

**,LOCKCOMP=NO LOCKCOMP**

**,LOCKCOMP=lockcomp**

This parameter has slightly different meanings based on the value that is specified for the LOCKOPER parameter. Generally, this input parameter specifies a connection identifier to be verified as the owner of the lock specified by LOCKINDEX. This verification is a prerequisite to the successful completion of the request.

When LOCKCOMP is specified, the completion of the request is conditional on there being no contention for the lock. If contention exists, the request fails .

The connection identifier is available from the IXLCONN answer area, which is mapped which is by IXLYCONA, in field CONACONID.

The effect of LOCKCOMP is based on the LOCKOPER value specified. See the description under LOCKOPER to see how each request is affected by this parameter.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 1-byte field that contains the connection identifier.

**,LOCKDATA=NO LOCKDATA**

**,LOCKDATA=lockdata**

Use this input parameter to specify information that is to be passed to your notify exit when another user requests the LOCKINDEX lock after you have obtained the lock by using LOCKOPER=SET. This user-defined information will be passed to your notify exit when the other user issues a request specifying either one of the following:

- LOCKOPER=SET
- LOCKOPER=NOTHELD with LOCKMODE=UNCOND

**To Code:** Specify the RS-type name or address (by using a register 2 - 12 2 to ) of an 8-byte field that contains the user-defined information.

**,LOCKINDEX=NO LOCKINDEX**

**,LOCKINDEX=lockindex**

Use this input parameter to specify the index of the lock to be operated on as specified by LOCKOPER. The lock indexes begin with zero.

**Note:** You cannot code LOCKINDEX with MODE=ASYNCSNORESPONSE.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 4-byte field that contains the lock index.

**,LOCKMODE=UNCOND**

**,LOCKMODE=COND**

Use this input parameter to specify how contention for the lock that is specified by LOCKINDEX should be handled if your request causes lock contention.

**UNCOND**

If the specified lock is held by another user, your request is either:

- Suspended until the lock becomes available (if MODE=SYNCSUSPEND is specified).
- Performed asynchronously with notification to you when the request completes (if any MODE value other than SYNCSUSPEND is specified).

**COND**

The lock operation is performed conditionally; that is, only if there is no contention for the lock. If the specified lock is held by another user, the request is terminated with no change to the structure, and return and reason codes describing the termination are returned to the caller.

**,LOCKOPER=SET**  
**,LOCKOPER=RESET**  
**,LOCKOPER=NOTHELD**  
**,LOCKOPER=HELD****BY**

Use this input parameter to specify the type of operation to be performed on the lock specified by LOCKINDEX.

#### **SET**

Set the lock.

- When LOCKCOMP is not specified, your connection gets the lock, providing no other connection holds the lock. If another connection holds the lock, the request either continues once the lock is released or fails, depending on the LOCKMODE value you specify.
- When LOCKCOMP is specified, if the connection specified by LOCKCOMP holds the lock, the lock is transferred to your connection.

#### **RESET**

Reset the lock.

- When LOCKCOMP is not specified, the lock is released if it is held by your connection.
- When LOCKCOMP is specified, the lock is released if it is held by the connection that is specified by LOCKCOMP.

#### **NOTHELD**

The request is performed only if the lock is not held by any connection. This option also ensures that the lock remains free for the duration of the request. If another connection holds the lock, your task is suspended until the lock is released or your request fails, depending on the LOCKMODE value you specify. See the LOCKMODE description for how to handle possible lock contention.

#### **HELD****BY**

Processing is determined by which connector is holding the lock.

- When LOCKCOMP is not specified, the request is performed only if your connection holds the lock.
- When LOCKCOMP is specified, the request is performed only if the lock is held by the connection that is specified by LOCKCOMP.

**,MF=S**  
**,MF=(L,mfctrl)**  
**,MF=(L,mfctrl,mfattr)**  
**,MF=(L,mfctrl,0D)**  
**,MF=(M,mfctrl)**  
**,MF=(M,mfctrl,COMPLETE)**  
**,MF=(M,mfctrl,NOCHECK)**  
**,MF=(E,mfctrl)**  
**,MF=(E,mfctrl,COMPLETE)**  
**,MF=(E,mfctrl,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into

the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

**,mfctrl**

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

**,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

**,COMPLETE**

**,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

**COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if *SMILE=var* were an optional parameter and the default is *SMILE=NO\_SMILE* then it would not be documented. However, if the default was *SMILE=-*), then it would be documented because a value would be the default.

**NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

**,MODE=SYNCSUSPEND**

**,MODE=SYNCECB**

**,MODE=SYNCEXIT**

**,MODE=SYNCTOKEN**

**,MODE=ASYNCECB**

**,MODE=ASYNCEXIT**

**,MODE=ASYNCTOKEN**

**,MODE=ASYNCSUSPEND**

Use this input parameter to specify:

- Whether the request is to be performed synchronously or asynchronously
- How you want to be notified of request completion

**MODE=SYNCSUSPEND**

The request is suspended until it can complete synchronously. To use this option, your program must be enabled for I/O and external interrupts.

**MODE=SYNCECB**

The request processes synchronously if possible. If the request processes asynchronously, the ECB specified by REQECB is posted when the request completes.

**MODE=SYNCEXIT**

The request processes synchronously if possible. If the request processes asynchronously, your complete exit is given control when the request completes.

**MODE=SYNCTOKEN**

The request processes synchronously if possible. If the request processes asynchronously, an asynchronous request token is returned in the area specified by REQTOKEN on invocation of the request. Use the returned request token on the IXLFCOMP macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=SYNCTOKEN is specified.

**MODE=ASYNCECB**

The request processes asynchronously. The ECB specified by REQECB is posted when the request completes.

**MODE=ASYNCEXIT**

The request processes asynchronously. Your complete exit is given control when the request completes.

**MODE=ASYNCTOKEN**

The request processes asynchronously. An asynchronous request token is returned in the area specified by REQTOKEN upon invocation of the request. Use the returned request token on the IXLFCOMP macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=ASYNCTOKEN is specified.

**MODE=ASYNCNORESPONSE**

The request processes asynchronously. No notification of request completion is provided. The answer area (ANSAREA) fields will not contain valid information when this mode is specified.

**Note:** You cannot code MODE=ASYNCNORESPONSE with LOCKINDEX, BUFFER, or BUFLIST.

**,MOVETOKEY=NO\_MOVETOKEY****,MOVETOKEY=movetokey**

Use this input parameter to assign a new key, and hence the position on the MOVETOLIST, for the existing entry being moved or the new entry being created. In the case of an existing entry, the existing entry's key will be updated to the MOVETOKEY key value. If MOVETOKEY is not specified, ENTRYKEY defines the key to use for the entry.

**Note:**

1. If there is a sublist of one or more entries with a matching key on the list then the target position is the head or tail of the sublist, as specified by the MOVETOPOS parameter.
2. If none of the list entries have a matching key, and MOVETOKEY is neither the greatest nor least among the list entry keys, then the target position is according to the appropriate key sequence for the list.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-byte field that contains the key.

**,MOVETOLIST=movetolist**

Use this input parameter to specify the number of the target list where the existing entry being moved or the new entry being created will be placed.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 4-byte field that contains the number of the target list.

**,MOVETOPOS=HEAD****,MOVETOPOS=TAIL**

Use this input parameter to specify the position where the existing or new entry will be placed on the MOVETOLIST.

**HEAD**

- The entry is placed at the head of the list when:
  - Keyed entries are not supported.
  - Keyed entries are supported, a new entry is being created, and neither MOVETOKEY nor ENTRYKEY are specified.
- The entry is placed at the head of the sublist of identically keyed entries when keyed entries are supported and one or more entries with the same key already exists on the MOVETOLIST list.

**TAIL**

- The entry is placed at the tail of the list when:
  - Keyed entries are not supported.

- Keyed entries are supported, a new entry is being created, and neither MOVETOKEY nor ENTRYKEY are specified.
- The entry is placed at the tail of the sublist of identically keyed entries when keyed entries are supported and one or more entries with the same key already exists on the MOVETOLIST list.

**,NEWAUTH=NO\_NEWAUTH****,NEWAUTH=newauth**

Use this input parameter to specify a list authority value for the list specified by LISTNUM. If NEWAUTH is omitted, the list authority for the designated list is unchanged.

When the structure is allocated, the authority value for each list is initialized to binary zeros.

**Note:** The NEWAUTH parameter is valid only for list structures allocated in a coupling facility with CFLEVEL=1 or higher, except on WRITE\_LCONTROLS requests.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12 2 to ) of the 16-byte field that contains the list authority value.

**,NEWVERS=newvers**

Use this input parameter with VERSUPDATE=SET to specify a version number to either replace the version number of the existing entry, or initialize the version number of a new entry.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-byte field that contains the entry version number.

**,PAGEABLE=YES****,PAGEABLE=NO**

Use this input parameter to identify whether the storage areas specified by BUFFER or BUFLIST are in pageable or potentially pageable storage.

**YES**

Specify this option to indicate that the BUFFER or BUFLIST buffers reside in pageable virtual storage. XES performs the required page fixing to fix the buffers in real storage while the cache or list request transfers data to or from the coupling facility.

This includes storage obtained from pageable subpools, disabled reference (DREF) storage, and may include storage that has the potential to become pageable during the processing of a request. (An example is address space storage owned by any swappable address space, for which a PGSER FIX has been successfully processed, but for which the owning address space gets swapped during processing of a cache or list request.) This does not include implicitly non-pageable storage (for example, storage obtained from non-pageable subpools).

The system takes responsibility for managing binds to central storage for the duration of the cache or list request, regardless of what address space owns the storage or whether the storage-owning address space is swappable or nonswappable. The storage can be owned by any address space.

High shared virtual storage areas (above 2GB) may not be used.

**NO**

Specify this option to indicate that the BUFFER or BUFLIST buffers reside in non-pageable virtual storage. XES does not page fix the buffers in real storage.

This includes implicitly non-pageable storage areas (for example, storage obtained from non-pageable subpools), and may include storage that has the potential to become pageable during the processing of a request (An example is address space storage owned by any swappable address space, for which a PGSER FIX has been successfully processed, but for which the owning address space gets swapped-out during processing of a cache or list request.)

The system takes responsibility for managing binds to central storage for the duration of the cache or list request, if and only if the non-pageable storage is owned by either the requester's address space or the connector's address space. If the storage is owned by any other address space, then the invoker is responsible for ensuring that the virtual storage remains non-pageable for the duration of the request (including the case in which the storage is owned by a swappable address space that is swapped during processing of an IXLCACHE or IXLLIST request). Subject to

this consideration, the storage can be owned by any address space. See *z/OS MVS Programming: Sysplex Services Guide*.

**,PLISTVER=IMPLIED\_VERSION**

**,PLISTVER=MAX**

**,PLISTVER=plistver**

Use this input parameter to specify the version of the macro. See “Understanding IXLLIST Version Support” on page 965 for a description of the options available with the PLISTVER macro.

**,REQDATA=NO\_REQDATA**

**,REQDATA=reqdata**

Use this input parameter with MODE=SYNCEXIT or MODE=ASYNCEXIT to pass any data you choose to the complete exit. The exit will get control only if the request is processed asynchronously.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the data to be passed to the complete exit.

**,REQECB=reqecb**

Use this output parameter with either MODE=SYNCECB or MODE=ASYNCECB to specify the address of an ECB, which is to be posted when the request completes if the request was processed asynchronously.

Before coding REQECB, you must ensure that:

- You initialize the ECB before you issue the request.
- The ECB resides in either common storage or the home address space where IXLCONN was issued.
- Any tasks that wait for the ECB to be posted reside in the home address space where IXLCONN was issued.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that contains the address of the ECB to be posted when the request completes. The ECB must be aligned on a fullword boundary.

**,REQID=NO\_REQID**

**,REQID=reqid**

Use this input parameter to specify a user-defined request identifier to be associated with the request. You can specify this request identifier on the IXPURGE macro to cancel a request that has not yet been processed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the user-defined request identifier.

**,REQTOKEN=reqtoken**

Use this output parameter with either MODE=SYNCTOKEN or MODE=ASYNCTOKEN to specify the address of a storage area to receive the request token that is returned when the request will be processed asynchronously. This token, which uniquely identifies the request, must be used as input to the IXLFComp macro, which you use to determine if the request has completed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field where the system will put the request token.

**,RETCODE=retcode**

Use this output parameter to specify a field to contain the return code. (The return code is also returned in register 15.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the return code when the request has completed.

**,RSNCODE=rsncode**

Use this output parameter to specify a field to contain the reason code returned, if applicable. (The reason code is also returned in register 0.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the reason code (if any) when the request has completed.

**,UPDATECURSOR=NO****,UPDATECURSOR=YES**

Use this input parameter to specify whether the MOVETOLIST list cursor is to be updated to point to another entry on the list.

**NO**

The list cursor is not updated.

**YES**

The list cursor is updated to point to the entry that is adjacent to the new or moved entry.

- If LISTDIR=TOHEAD, the list cursor will point to the adjacent entry that is closest to the head of the list.
- If LISTDIR=TOTAIL, the list cursor will point to the adjacent entry that is closest to the tail of the list.

The list cursor is set to binary zeros if its new position would be before the first entry or after the last.

**Note:** UPDATECURSOR may not be specified when DATAOPER=WRITE and ENTRYTYPE=ANY are specified

**,VERSCOMP=NO\_VERSCOMP****,VERSCOMP=verscomp**

Use this input parameter to specify a version number to be compared to the version number of the existing entry. If this request creates a new entry, the VERSCOMP specification is ignored.

The existing entry is moved only if its version number meets the condition specified by the VERSCOMPTYPE parameter. If the entry does not have the specified version number, the request is terminated with no change to the structure.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-byte field that contains the version number.

**,VERSCOMPTYPE=EQUAL****,VERSCOMPTYPE=LESSOREQUAL**

Use this input parameter to specify how a list entry version comparison as specified by VERSCOMP is to be performed.

**Note:** The VERSCOMPTYPE parameter is valid only for list structures allocated in a coupling facility with CFLEVEL=1 or higher.

**VERSCOMPTYPE=EQUAL**

The version number for the list entry must be equal to the value specified for VERSCOMP.

**VERSCOMPTYPE=LESSOREQUAL**

The version number for the list entry must be less than or equal to the value specified for VERSCOMP.

**,VERSUPDATE=NONE****,VERSUPDATE=INC****,VERSUPDATE=DEC****,VERSUPDATE=SET**

Use this input parameter to specify how the entry version number of the moved entry will be updated, or for those cases where a new entry is created, initialized.

**VERSUPDATE=NONE**

The existing entry's version number is not updated. The new entry's version number is set to binary zeros.

**VERSUPDATE=INC**

The existing entry's version number is increased by one. The new entry's version number is set to binary zeros, except for the low-order bit, which is set to one.



**VERSUPDATE=DEC**

The existing entry's version number is decreased by one. The new entry's version number is set to all binary ones.

**VERSUPDATE=SET**

Both the existing and the new entry's version number is set to the value specified by NEWVERS.

## ABEND Codes

---

Abend X'026' (See *z/OS MVS System Codes* for more information on this abend.)

## Return and Reason Codes

---

When the system returns control to your program:

- GPR 15 (and RETCODE, if specified) contains a return code.
- If the return code is not zero, GPR 0 (and RSNCODE, if specified) contains a reason code.

**Note:** The return and reason codes will also be put into the answer area (mapped by IXLYLAA), if there is one, when the request completes.

The IXLYCON macro provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

**0**

IXLRETCODEOK

**4**

IXLRETCODEWARNING

**8**

IXLRETCODEPARMERROR

**C**

IXLRETCODEENVERROR

**10**

IXLRETCODECOMPERROR

The following table contains hexadecimal return and reason codes, the equate symbols associated with each reason code, and the meaning and suggested action for each return and reason code.

Table 66. Return and Reason Codes for IXLLIST REQUEST=MOVE Macro		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0	None.	<p><b>Meaning:</b> If you specified a MODE value of ASYNCECB, ASYNCEXIT, ASYNCTOKEN, or ASYNCNORESPONSE, the request has been successfully initiated. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed. For any other MODE value, the request has successfully completed.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=ASYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=ASYNCEXIT, the connection's complete exit will be given control when the request has completed.</li> <li>• If you specified MODE=ASYNCTOKEN, use REQTOKEN when calling IXLFCOMP to determine when the request has completed.</li> </ul>

Table 66. Return and Reason Codes for IXLLIST REQUEST=MOVE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	xxxx0402	<p><b>Equate Symbol:</b> IXLRNCODEASYNCH</p> <p><b>Meaning:</b> The request could not be processed synchronously. It will be processed asynchronously. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed.</p> <ul style="list-style-type: none"> <li>• If MODE=SYNCECB was specified, the ECB specified for REQECB will be posted when the request has finished.</li> <li>• If MODE=SYNCEXIT was specified, the connection's complete exit will be given control when the request has finished.</li> <li>• If MODE=SYNCTOKEN was specified, an asynchronous request token was returned in the area specified by REQTOKEN. This request token may be specified on the IXLFCOMP macro to determine when the request has finished.</li> </ul> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=SYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=SYNCEXIT, the connection's complete exit will be given control when the request has completed.</li> <li>• If you specified MODE=SYNCTOKEN, use REQTOKEN when calling IXLFCOMP to determine when the request has completed.</li> </ul>

Table 66. Return and Reason Codes for IXLLIST REQUEST=MOVE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	xxxx040D	<p><b>Equate Symbol:</b> IXLRNCODEBADREADADJDATA</p> <p><b>Meaning:</b> Program error. The request specified that adjunct data was to be read, but the storage area specified by ADJAREA is not addressable. All other requested data was retrieved, and the designated entry was moved. The following fields in the answer area have been filled in:</p> <ul style="list-style-type: none"> <li>• LAATOTALCNT - The total count of allocated entries in the list structure.</li> <li>• LAATOTALELECNT - The total count of allocated elements in the list structure.</li> <li>• LAALISTCNT - The count of entries or elements residing on the target list.</li> <li>• LAALCTL (if the request completed prematurely as well) - The list entry controls for the first UNPROCESSED entry in the list sequence.</li> <li>• LAAFIRSTKEY - Indicates a successful MOVE request and the list entry is the only entry on the list with the entry key value. This field is returned only for a list structure allocated in a coupling facility with CFLEVEL=1.</li> </ul> <p><b>Note:</b> Only the adjunct data for the first entry in the list is placed in the ADJAREA. The buffers should contain the adjunct data for the other entries in the list.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the ADJAREA address.</li> <li>• ADJAREA must be addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are calling IXLLIST while disabled, ADJAREA must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLLIST in AR-mode and you specified the ADJAREA address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are calling IXLLIST in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLIST macro.</li> </ul> <p>Correct the address specified by ADJAREA, and rerun the request asking for adjunct data only.</p>

Table 66. Return and Reason Codes for IXLLIST REQUEST=MOVE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	xxxx0410	<p><b>Equate Symbol:</b> IXLRSNCODELOCKCOND</p> <p><b>Meaning:</b> For a LOCKOPER=HELDDBY request, or a LOCKMODE=COND request, or a request that specified LOCKCOMP, the request could not be completed successfully because the specified lock is not currently held as required. The connection identifier of the lock owner is returned in the answer area (LAACONID field).</p> <p><b>Action:</b> Retry the request, or obtain the lock as required, and retry the request. If you are unable to get the lock, check the ID in the LAACONID field and determine if some recovery is necessary.</p>
4	xxxx0412	<p><b>Equate Symbol:</b> IXLRSNCODELOCKHELDDBYSYS</p> <p><b>Meaning:</b> The lock is not held by any connection, but instead is held by the system. For a request that specified any of the following, the request could not be completed successfully because the specified lock is not currently held as required:</p> <ul style="list-style-type: none"> <li>• LOCKOPER=SET or LOCKOPER=NOTHELD with either of: <ul style="list-style-type: none"> <li>– LOCKMODE=COND</li> <li>– LOCKCOMP</li> </ul> </li> <li>• LOCKOPER=HELDDBY</li> </ul> <p><b>Action:</b> Retry the request, or obtain the lock as required, and retry the request.</p>
8	xxxx0801	<p><b>Equate Symbol:</b> IXLRSNCODEBADPARMLIST</p> <p><b>Meaning:</b> Program error. The parameter list for this request is not addressable. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Verify that:</p> <ul style="list-style-type: none"> <li>• The parameter list address is uncorrupted.</li> <li>• The parameter list is addressable in the caller's primary address space.</li> <li>• If you are issuing this macro while disabled, the parameter list resides in either page-fixed or DREF storage.</li> <li>• If you are issuing this macro in AR-mode and you specified the parameter list address using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If you are issuing this macro in AR-mode, SYSSTATE ASCENV=AR must be issued before issuing this macro.</li> </ul>

Table 66. Return and Reason Codes for IXLLIST REQUEST=MOVE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0804	<p><b>Equate Symbol:</b> IXLRNCODEBADVERSIONNUM (or IXLRNCODEBADVERSION#)</p> <p><b>Meaning:</b> The version number in the macro parameter list is not compatible with the level of XES currently being used. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that your program did not overlay the parameter list storage.</li> <li>• Verify that your program was assembled with the correct macro library for the release of MVS on which your program is running.</li> </ul>
8	xxxx080A	<p><b>Equate Symbol:</b> IXLRNCODEBADCONTOKEN</p> <p><b>Meaning:</b> Program error. The specified connect token (CONTOKEN) was not valid for one of the following reasons:</p> <ol style="list-style-type: none"> <li>1. The user with the connection identifier represented by the token has disconnected from the structure.</li> <li>2. The connector's task (the task that issued IXLCONN) ended.</li> <li>3. The specified token is not the token that was returned from IXLCONN.</li> <li>4. The request was issued from an address space other than the address space in which IXLCONN was issued.</li> <li>5. The connect token was invalidated during rebuild.</li> <li>6. The connect token was invalidated by XES.</li> </ol> <p><b>Note:</b> The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Take the action with the corresponding meaning.</p> <ol style="list-style-type: none"> <li>1. Discontinue use of the structure. Perform recovery and cleanup for the structure and check the protocol for the use of the structure to determine why you did not know of the disconnection.</li> <li>2. Discontinue use of the structure. Perform recovery and cleanup for the structure and check the protocol for the use of the structure to determine why you did not know that the task ended.</li> <li>3. The contoken is returned in the CONACONTOKEN field of the answer area specified on the IXLCONN request.</li> <li>4. Issue your request from the same address space the IXLCONN was issued in.</li> <li>5. Wait for the rebuild to complete, and try again.</li> <li>6. Discontinue use of the structure. Perform recovery and cleanup for the structure.</li> </ol>

Table 66. Return and Reason Codes for IXLLIST REQUEST=MOVE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0824	<p><b>Equate Symbol:</b> IXLRSNCODEWRONGSTRTYPE</p> <p><b>Meaning:</b> Program error. The connection specified by CONTOKEN is not to a list structure.</p> <p><b>Action:</b> Specify the appropriate CONTOKEN value, and rerun your program. The CONTOKEN is returned in the answer area provided on the IXLCONN invocation. The type of structure built is passed on the TYPE parameter of the IXLCONN macro.</p>
8	xxxx0825	<p><b>Equate Symbol:</b> IXLRSNCODENOENTRY</p> <p><b>Meaning:</b> Program error. The designated list entry does not exist; therefore, no entries were processed.</p> <p><b>Action:</b> None necessary. However, if this return code and reason code are unexpected, you should check the way the list entry was created. Examine the parameters specified for locating the list entry on the invocation of this macro.</p>
8	xxxx0833	<p><b>Equate Symbol:</b> IXLRSNCODEBADPGBLATTR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list is specified as being pageable (PAGEABLE=YES), but is not.</p> <p><b>Action:</b> Change the buffer area(s) to pageable storage, or specify PAGEABLE=NO. See the PAGEABLE parameter description for specification instructions.</p>

Table 66. Return and Reason Codes for IXLLIST REQUEST=MOVE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0834	<p><b>Equate Symbol:</b> IXLSRNCODEBADNONPGBLATTR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list, is specified as being nonpageable (PAGEABLE=NO), but is either pageable or not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The type of storage in the buffer area corresponds to the value you specified on the PAGEABLE parameter. See the PAGEABLE parameter description for specification instructions.</li> <li>• The correct buffer address was used.</li> <li>• The buffer area was not previously freed.</li> <li>• If you are calling IXLLIST while disabled, the buffers must reside in either page-fixed or DREF storage.</li> <li>• The buffer areas were allocated in a storage key that matches the key specified by the BUFSTGKEY parameter, or, if BUFSTGKEY is omitted, the caller's PSW key.</li> <li>• If BUFLIST was specified and your program is running in AR-mode: <ul style="list-style-type: none"> <li>– If the BUFFER or BUFLIST parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>– The BUFALET specification is correct.</li> <li>– You specified SYSSTATE ASCENV=AR before issuing the IXLLIST macro.</li> </ul> </li> </ul>
8	xxxx0835	<p><b>Equate Symbol:</b> IXLSRNCODEBADDATAADDR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list, is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The correct buffer address was used.</li> <li>• The buffer area was not previously freed.</li> <li>• The buffer area was allocated in a storage key that matches the key specified using the BUFSTGKEY parameter, or, if BUFSTGKEY is omitted, the caller's PSW key.</li> <li>• If BUFLIST was specified and your program is running in AR mode: <ul style="list-style-type: none"> <li>– The BUFALET specification is correct.</li> <li>– You specified SYSSTATE ASCENV=AR before issuing the macro.</li> </ul> </li> </ul>

Table 66. Return and Reason Codes for IXLLIST REQUEST=MOVE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0836	<p><b>Equate Symbol:</b> IXLRSNCODEBADREALADDR</p> <p><b>Meaning:</b> Program error. Real storage addresses were provided in the BUFLIST list, but one of the buffers is not addressable in central storage.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that BUFADDRTYPE was specified as you intended.</li> <li>• Ensure that the buffer addresses specified by BUFLIST are valid.</li> </ul>
8	xxxx0837	<p><b>Equate Symbol:</b> IXLRSNCODEBADWRITEADJDATA</p> <p><b>Meaning:</b> Program error. The request specified that adjunct data was to be written, but the area specified by ADJAREA is not addressable. There was no change to the structure.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the ADJAREA address.</li> <li>• ADJAREA must be addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are calling IXLLIST while disabled, ADJAREA must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLLIST in AR-mode and you specified the ADJAREA address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are calling IXLLIST in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLIST macro.</li> </ul>
8	xxxx0838	<p><b>Equate Symbol:</b> IXLRSNCODEBADANSAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by ANSAREA is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The answer area address specified by ANSAREA is valid.</li> <li>• If the caller is running in AR-mode and the ANSAREA parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• The address specified for ANSAREA is addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are calling IXLLIST while disabled, ANSAREA must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLLIST in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLIST macro.</li> </ul>



Table 66. Return and Reason Codes for IXLLIST REQUEST=MOVE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0839	<p><b>Equate Symbol:</b> IXLRSNCODEBADREQTOKENAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by REQTOKEN is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The request token area specified by REQTOKEN is valid.</li> <li>• If the caller is running in AR-mode and the REQTOKEN parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If you are calling IXLLIST while disabled, REQTOKEN must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLLIST in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLIST macro.</li> </ul>
8	xxxx083D	<p><b>Equate Symbol:</b> IXLRSNCODEBADANSLEN</p> <p><b>Meaning:</b> Program error. The length of the answer area, as specified by ANSLEN, is not sufficient to contain answer area information. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Increase the size of the answer area, and rerun your program. The minimum size of the ANSAREA is in the LAA_LEN field of the IXLYLAA mapping macro.</p>
8	xxxx083E	<p><b>Equate Symbol:</b> IXLRSNCODEMAXLISTKEY</p> <p><b>Meaning:</b> Program error. The list key to be assigned to an entry that was being created or moved was not valid. Either the list key or the list key plus the list key increment value was greater than the maximum list key.</p> <p><b>Action:</b> Ensure that you specified a correct list key increment. Depending on the protocol you are using for assigning list key values, you might want to issue a WRITE_LCONTROLS request to update either the list key value to a lower value or the maximum list key value to a higher value.</p>
8	xxxx083F	<p><b>Equate Symbol:</b> IXLRSNCODEBADENTRYVERSION</p> <p><b>Meaning:</b> The designated entry has a version number that failed the comparison specified by VERSCOMPTYPE. The list entry controls for the entry are returned in the answer area (field LAALCTL).</p> <p><b>Action:</b> None necessary. However, if this return code and reason code are unexpected, you might want to verify the value specified in VERSCOMP and look at the version returned in the answer area.</p>

Table 66. Return and Reason Codes for IXLLIST REQUEST=MOVE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0840	<p><b>Equate Symbol:</b> IXLRSNCODEBADENTRYLIST</p> <p><b>Meaning:</b> The entry designated by ENTRYID or ENTRYNAME exists in the structure, but does not reside on the list specified by LISTNUM. The list entry controls for the entry are returned in the answer area (field LAALCTL).</p> <p><b>Action:</b> None necessary. However, if you did not expect this return and reason code, you might want to verify what list this entry is on. Maybe the entry was moved, or you designated the wrong list in LISTNUM. Check the protocol for using this list.</p> <p>The information returned in the LAALCTL field of the answer area contains the number of the list that this list entry is on as well as other control information.</p>
8	xxxx0841	<p><b>Equate Symbol:</b> IXLRSNCODEBADENTRYNAME</p> <p><b>Meaning:</b> Program error. Entry creation was attempted, but no entry was created for one of the following reasons:</p> <ul style="list-style-type: none"> <li>• No entry name was specified (and the structure supports names).</li> <li>• The specified entry name is not unique within the structure.</li> </ul> <p>The list entry controls for the first entry on the list are returned in the answer area (field LAALCTL).</p> <p><b>Action:</b> Be sure to provide a unique entry name when creating entries to be written to structures that support entry names.</p>
8	xxxx0842	<p><b>Equate Symbol:</b> IXLRSNCODEPERSISTENTLOCK</p> <p><b>Meaning:</b> Program error. The request specifying LOCKOPER=SET with LOCKMODE=UNCOND, or LOCKOPER=NOTHELD with LOCKMODE=UNCOND failed because the lock is held by a connection that is in the failed-persistent state. The connection identifier of the lock owner is returned in the answer area (field LAACONID).</p> <p><b>Action:</b> Either perform recovery for the connection, or wait until recovery for the connection is performed.</p>
8	xxxx0845	<p><b>Equate Symbol:</b> IXLRSNCODENONAMES</p> <p><b>Meaning:</b> Program error. A list entry was designated by entry name, but the structure does not support entry names.</p> <p><b>Action:</b> Ensure that you are connected to the intended structure. Ensure that the correct specification was made for REFOPTION on the IXLCONN macro. You may want to issue IXLMG to get more information about the structure.</p>

Table 66. Return and Reason Codes for IXLLIST REQUEST=MOVE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0846	<p><b>Equate Symbol:</b> IXLRSNCODEBADLOCKINDEX</p> <p><b>Meaning:</b> Program error. The specified LOCKINDEX exceeds the size of the lock table for the structure.</p> <p><b>Action:</b> Correct LOCKINDEX to specify an index that is contained within the lock table. The maximum value for the LOCKINDEX is one less than the value specified by the LOCKENTRIES parameter on the IXLCONN macro.</p>
8	xxxx0847	<p><b>Equate Symbol:</b> IXLRSNCODEBADLISTNUMBER</p> <p><b>Meaning:</b> Program error. The specified LISTNUM value exceeds the number of lists for the structure.</p> <p><b>Action:</b> Correct LISTNUM to specify a list number that exists in the structure. The number of lists allocated for a structure is determined on the LISTHEADERS parameter of the IXLCONN macro. The list numbers go from 0 to n-1, where n is the number of lists specified.</p>
8	xxxx0848	<p><b>Equate Symbol:</b> IXLRSNCODEBADRESET</p> <p><b>Meaning:</b> Program error. LOCKOPER=RESET was specified for a lock not currently held by the invoker. The value of the connection ID holding the lock is returned in the answer area (field LAACONID).</p> <p><b>Action:</b> Check your code to ensure that your connection did not already reset the lock.</p>
8	xxxx084A	<p><b>Equate Symbol:</b> IXLRSNCODENOKEYS</p> <p><b>Meaning:</b> Program error. The structure does not support the use of entry keys. The IXLLIST request type either required the structure to support entry keys or designated a list entry or list position by list number and entry key.</p> <p><b>Action:</b> Ensure that you are connected to the intended structure. The use of keys for a structure is determined by the REFOPTION keyword on the IXLCONN macro.</p>
8	xxxx084B	<p><b>Equate Symbol:</b> IXLRSNCODENOLOCKS</p> <p><b>Meaning:</b> Program error. A locking operation was requested, but the structure does not contain a lock table.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• You are connected to the intended structure.</li> <li>• You intended to perform a locking operation.</li> </ul> <p>The use of locks is determined by the LOCKENTRIES keyword on the IXLCONN macro.</p>

Table 66. Return and Reason Codes for IXLLIST REQUEST=MOVE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx084E	<p><b>Equate Symbol:</b> IXLRSNCODEBADMOVETOLIST</p> <p><b>Meaning:</b> Program error. The list number specified for MOVETOLIST exceeds the number of lists defined for the structure.</p> <p><b>Action:</b> Correct MOVETOLIST to specify a list that exists in the structure. The maximum number of lists in a structure is determined by the LISTHEADERS parameter on the IXLCONN macro.</p>
8	xxxx0851	<p><b>Equate Symbol:</b> IXLRSNCODENOSUSPENDISABLE</p> <p><b>Meaning:</b> Program error. The request failed because MODE=SYNCSUSPEND was specified, but the caller is disabled and cannot be suspended.</p> <p><b>Action:</b> Either specify another MODE value, or become enabled (release the CPU lock), then reissue the request.</p>
8	xxxx0854	<p><b>Equate Symbol:</b> IXLRSNCODEBADLOCKCOMP</p> <p><b>Meaning:</b> Program error. The connection identifier specified for LOCKCOMP is not valid.</p> <p><b>Action:</b> The connection identifier is returned in the answer area (mapped by IXLYCONA) when the IXLCONN macro was issued.</p>
8	xxxx0859	<p><b>Equate Symbol:</b> IXLRSNCODEBADLISTAUTH</p> <p><b>Meaning:</b> The list authority specified for AUTHCOMP failed the comparison specified by AUTHCOMPTYPE for the list authority of the specified list. The current list authority (field LAALISTAUTH) and description (field LAALISTDESC) are returned in the answer area.</p> <p><b>Action:</b> None required; however you might want to take some action depending on your application. The list authority is set to binary zeros when the structure is allocated. When IXLLIST is issued with the NEWAUTH keyword, the list authority is changed if the correct comparison list authority value is specified. Check the answer area to determine the list authority value for the list specified. Verify that the correct list number was specified.</p>
8	xxxx0864	<p><b>Equate Symbol:</b> IXLRSNCODEBADBUFSIZE</p> <p><b>Meaning:</b> Program error. The size of the BUFFER area or the buffer areas specified by BUFLIST is not large enough to contain the data for the first entry to be read in the list. No data is returned.</p> <p><b>Action:</b> If more space is available, specify a larger buffer size, or more buffers, and reissue the request. Check the information returned in the answer area (mapped by IXLYLAA) in the field LAALCTL (mapped by IXLYLCTL).</p>

Table 66. Return and Reason Codes for IXLLIST REQUEST=MOVE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0865	<p><b>Equate Symbol:</b> IXLRSNCODEBADBUFSPEC</p> <p><b>Meaning:</b> Program error. There is an error in the buffer specification.</p> <p><b>Action:</b> Check the following:</p> <ul style="list-style-type: none"> <li>• If BUFLIST was specified, check the requirements for BUFLIST, BUFNUM, and BUFINCRNUM.</li> <li>• If BUFFER was specified, check the requirements for BUFFER and BUFSIZE.</li> <li>• Buffer pointer(s) in BUFLIST</li> <li>• Buffer boundaries.</li> </ul>
8	xxxx0866	<p><b>Equate Symbol:</b> IXLRSNCODEBADBUFKEY</p> <p><b>Meaning:</b> Program error. The buffer storage key specified by BUFSTGKEY is incorrect or BUFSTGKEY was not specified and the PSW key does not match the key of the buffers. For a request that writes data, the data cannot be fetched from the specified buffer area. For a request that reads data, the data cannot be stored into the specified buffer area.</p> <p><b>Action:</b> Check the following:</p> <ul style="list-style-type: none"> <li>• Determine if the key of the storage being used for the buffers is different from the PSW key.</li> <li>• If BUFSTGKEY was specified, verify that the key was put in the correct bits (see the explanation of this parameter for more information).</li> <li>• If you are calling IXLLIST in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLIST macro.</li> </ul>
8	xxxx0867	<p><b>Equate Symbol:</b> IXLRSNCODEBADBUFLIST</p> <p><b>Meaning:</b> Program error. The 128-byte storage area specified by BUFLIST is not addressable.</p> <p><b>Action:</b> Ensure that the address specified by BUFLIST is valid.</p>
8	xxxx086A	<p><b>Equate Symbol:</b> IXLRSNCODEBADELEMNUM</p> <p><b>Meaning:</b> Program error. The value specified for ELEMNUM is not valid.</p> <p><b>Action:</b> Correct the ELEMNUM specification to be within the allowable range: from zero to whatever MAXELEMNUM value was returned to the connector in the connect answer area.</p>

Table 66. Return and Reason Codes for IXLLIST REQUEST=MOVE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C06	<p><b>Equate Symbol:</b> IXLRNSCODENOCNN</p> <p><b>Meaning:</b> Environmental error. This system does not have connectivity to the coupling facility that contains the list structure. Possible reasons for this are:</p> <ul style="list-style-type: none"> <li>• The operator issued VARY PATH,OFFLINE.</li> <li>• The operator issued CONFIG CHP,OFFLINE.</li> <li>• Hardware errors to the coupling facility.</li> <li>• Facility or path failure to the coupling facility.</li> </ul> <p><b>Action:</b> Begin rebuilding the structure on a different coupling facility, or disconnect from the structure.</p>
C	xxxx0C13	<p><b>Equate Symbol:</b> IXLRNSCODEREQPURGED</p> <p><b>Meaning:</b> Environmental error. The request was purged prior to completion of the request. Possible reasons include:</p> <ul style="list-style-type: none"> <li>• The connector failed.</li> <li>• The connector disconnected.</li> <li>• The requestor failed.</li> <li>• The request was purged by IXLPURGE.</li> <li>• Requests were purged when the connector provided an IXLEERSP response for the Rebuild Stop or Rebuild Cleanup event.</li> <li>• The secondary address space was no longer valid.</li> </ul> <p><b>Action:</b> None if this is expected. Otherwise, determine why the connector failed.</p>
C	xxxx0C14	<p><b>Equate Symbol:</b> IXLRNSCODESTATUSUNKNOWN</p> <p><b>Meaning:</b> Environmental error. The request has completed, but the final disposition of the request cannot be determined. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the validity of your data by comparing the expected results with what is in the coupling facility.</li> <li>• Consider running rebuild processing to move the structure to attempt to protect it from further errors of this kind.</li> </ul>

Table 66. Return and Reason Codes for IXLLIST REQUEST=MOVE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C17	<p><b>Equate Symbol:</b> IXLRSNCODESTRFULL</p> <p><b>Meaning:</b> Environmental error. The request attempted to create a new entry or overwrite an existing entry, but the structure is full and cannot accommodate any more entries.</p> <p><b>Action:</b> Determine why the structure is full. You should be monitoring the usage of the structure every time a read or write is done (LAATOTALCNT is the total count of allocated entries in the list structure, and LAATOTALELECNT is the total count of allocated elements in the list structure). This data should be evaluated periodically to ensure structure resources are being used efficiently. You might be able to delete existing entries to free up space, rebuild the structure to make it larger if rebuild is allowed (IXLCONN macro, ALLOWREBLD parameter), or alter the size of the structure (IXLALTER macro).</p>
C	xxxx0C18	<p><b>Equate Symbol:</b> IXLRSNCODELISTFULL</p> <p><b>Meaning:</b> Environmental error. The entry could not be placed on the designated target list because the list is full.</p> <p><b>Action:</b> Either change the maximum number of entries allowed on the list by specifying a new LISTLIMIT on a WRITE_LCONTROLS request. You should be monitoring the usage of the list structure every time a read or write is done. (LAATOTALCNT is the total count of allocated entries in the list structure, and LAATOTALELECNT is the total count of allocated elements in the list structure.) This data should be evaluated periodically to ensure structure resources are being used efficiently.</p>
C	xxxx0C25	<p><b>Equate Symbol:</b> IXLRSNCODESTRFAILURE</p> <p><b>Meaning:</b> Environmental error. The list structure failed prior to completion of the request.</p> <p><b>Action:</b> Either rebuild or disconnect from the structure.</p>
C	xxxx0CA0	<p><b>Equate Symbol:</b> IXLRSNCODEQUIESCEDSUSPENDFAIL</p> <p><b>Meaning:</b> Environmental error. The request is failed because the structure is quiesced for a system-managed process and SUSPEND=FAIL is specified on the IXLCONN.</p> <p><b>Action:</b> None, if this is expected.</p>
C	xxxxFFFF	<p><b>Equate Symbol:</b> IXLRSNCODENOTAVAILABLE</p> <p><b>Meaning:</b> Environmental error. XES functions are not available. This could occur because the coupling facility hardware necessary to provide XES functions is not present.</p> <p><b>Action:</b> Re-IPL the system, or follow your particular failure management protocol.</p>

Table 66. Return and Reason Codes for IXLLIST REQUEST=MOVE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
10	xxxx10xx	<b>Meaning:</b> System error. XES processing failure. The state of the involved structure and the disposition of the request are unpredictable. The answer area (ANSAREA) fields do not contain valid information. <b>Action:</b> Contact the IBM support center.



## Chapter 67. IXLLIST REQUEST=READ

### Description

A READ request allows you to read a single list entry from the list structure into the storage areas specified by BUFLIST or BUFFER and/or ADJAREA. To designate an entry to be read, specify one of the following parameters or parameter combinations:

- List number (LISTNUM) with list position (LISTPOS)
- Entry identifier (ENTRYID)
- Entry name (ENTRYNAME)
- Locate by list cursor (LOCBYCURSOR) with a list number (LISTNUM)
- Entry key (ENTRYKEY) with a list number (LISTNUM)

Entry keys, however, are not unique within the structure such that different entries may have the same key (a sublist) on the same list. If there is a sublist of entries with the same key as the ENTRYKEY key (or that satisfies the KEYREQTYPE criteria, if specified), the entry that is designated is the entry at the HEAD or TAIL of the sublist as specified by LISTPOS. (KEYREQTYPE allows you to designate an entry with a key that varies from, but is as close as possible to, the specified key.)

With a coupling facility of CFLEVEL=1 or higher, the following additional functions are supported for a READ request:

- You can specify that a list entry is to be read only after a successful comparison of the list authority value for the target list with a user-specified value. You specify the list authority comparison requirements using AUTHCOMP and AUTHCOMPTYPE. If the request is successful, you can also update the list authority value to a new value that you specify with NEWAUTH.
- You can specify that a list entry be read only if it passes a version number comparison. You specify the version comparison requirements using VERSCOMP and VERSCOMPTYPE.
- There are additional list cursor options. You can update the list cursor conditionally, and you can set the list cursor to point to the current entry instead of the previous or next entry.

A READ request reads any combination of the following types of data for the designated entry:

- List entry controls
- Entry data
- Adjunct data

Whether a particular type of data is returned depends on whether you specified the local storage area to contain that data. Listed below are the types of data that are returned when the indicated storage areas are specified:

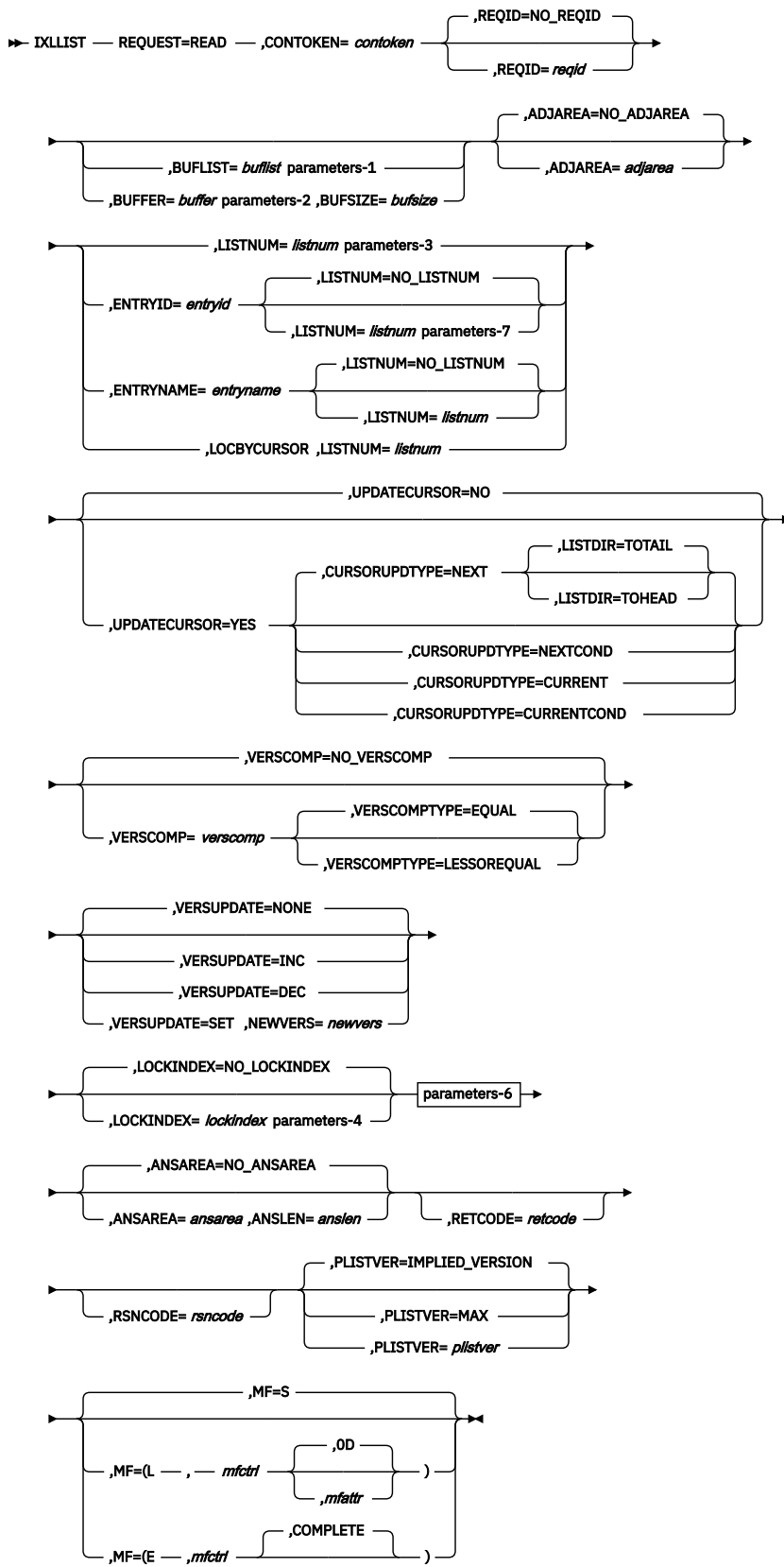
- List entry controls, the number of list entries or elements residing on the list, and the total number of allocated entries in the structure are returned in the answer area (ANSAREA) mapped by IXLYLAA.
- Entry data is returned in a buffer (BUFFER) or list of buffers (BUFLIST).
- Adjunct data is returned to the adjunct area (ADJAREA).
- If you do not specify either BUFFER or BUFLIST, or ADJAREA, list entry controls are returned in the answer area.

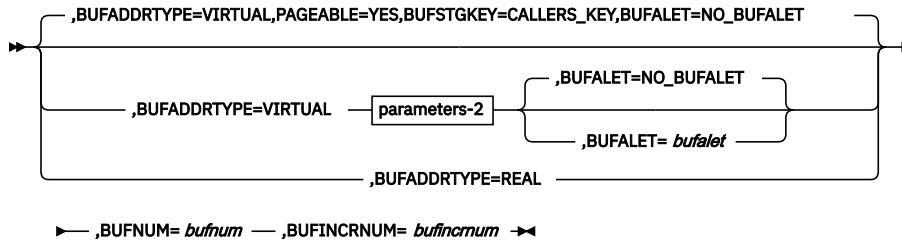
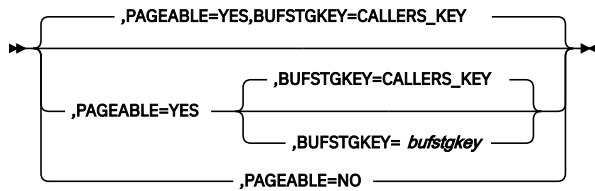
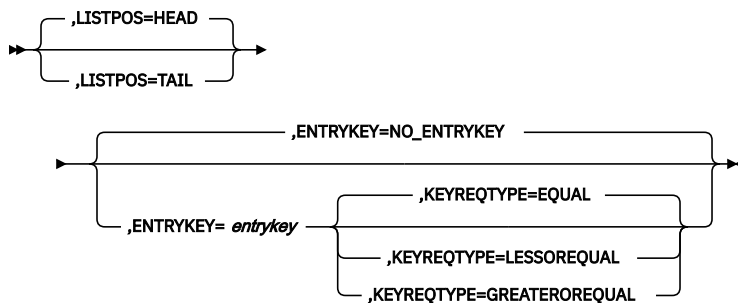
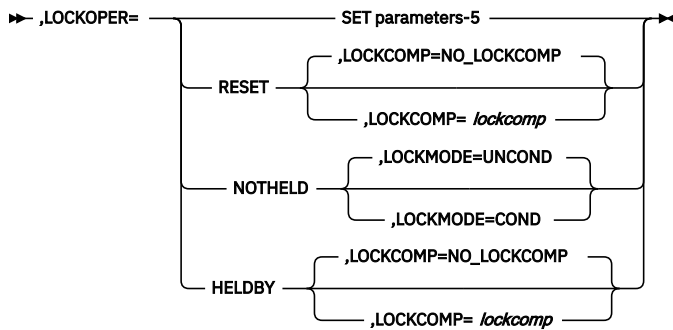
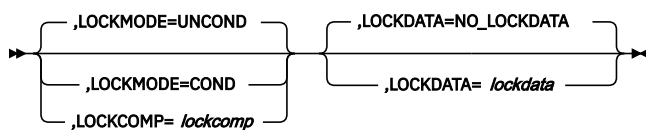
Additionally, you can perform locking functions with a READ request by specifying LOCKOPER. The lock designated by LOCKINDEX will be operated on as specified by LOCKOPER.

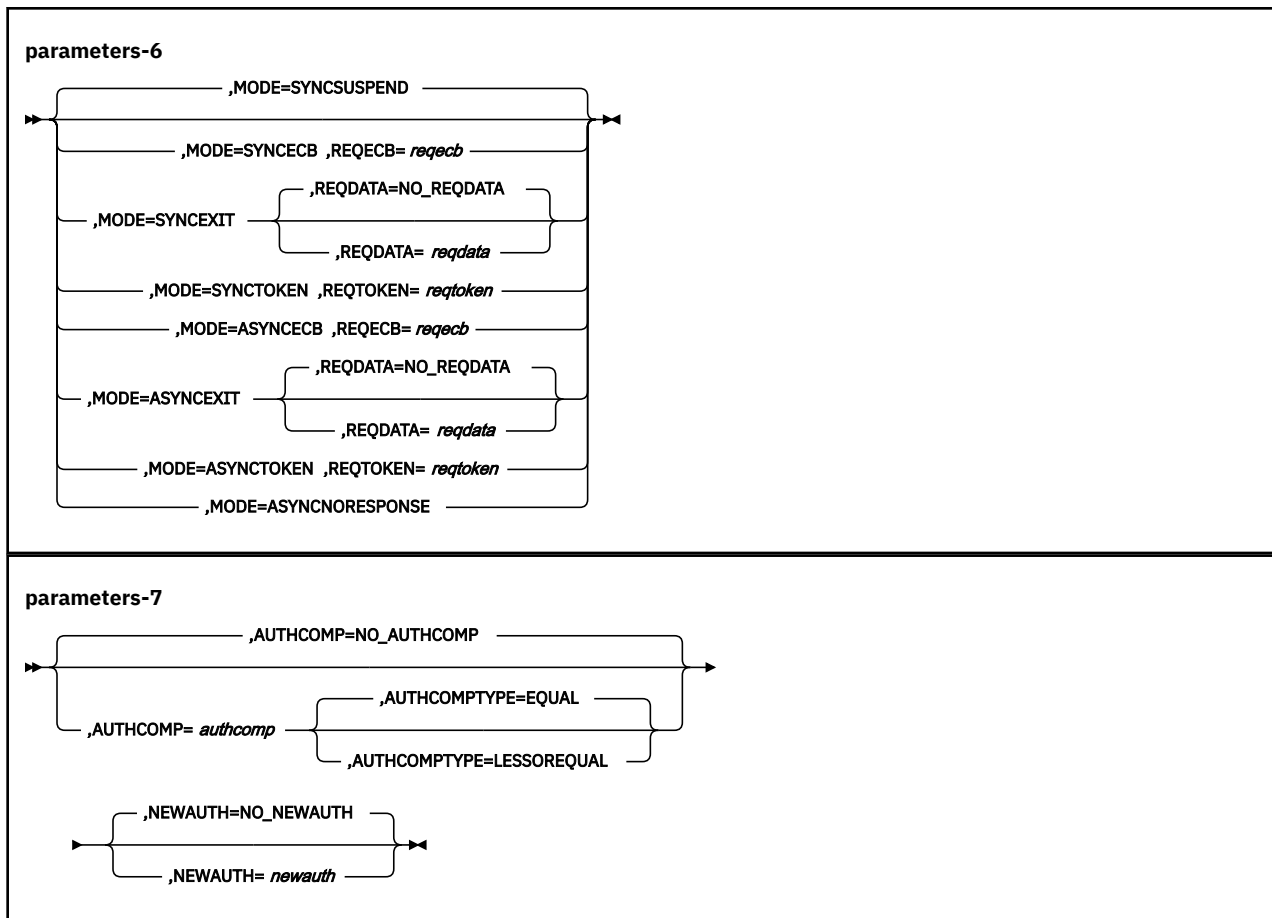
### Syntax Diagram

The syntax diagram for IXLLIST REQUEST=READ is as follows:

## main diagram



**parameters-1****parameters-2****parameters-3****parameters-4****parameters-5**



**Note:** If MODE=SYNCTOKEN or MODE=ASYNCTOKEN is specified, ANSAREA=*ansarea* and ANSLN=*anslen* are required.

## Parameter Descriptions

The parameter descriptions for REQUEST=READ are listed in alphabetical order. Default values are underlined:

### **REQUEST=READ**

Use this input parameter to read a single list entry.

#### **,ADJAREA=NO\_ADJAREA**

#### **,ADJAREA=adjarea**

Use this output parameter to specify a storage area to contain the adjunct data that was read from the designated entry.

Specify ADJAREA only for structures that support adjunct data. (Adjunct areas for a structure are established through the IXLCONN macro.)

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 64-byte area where the adjunct data will be put.

#### **,ANSAREA=NO\_ANSAREA**

#### **,ANSAREA=ansarea**

Use this output parameter to specify an answer area to contain information returned from the request. The format of the answer area is described by mapping macro IXLYLAA.

On a successful completion of the request, the following information is returned to the answer area:

- List entry controls of the designated entry (field LAALCTL).
- The number of list entries or elements residing on the list (field LAALISTCNT).

- The total number of allocated entries in the structure (field LAATOTALCNT).
- The total number of allocated elements in the structure (field LAATOTALELECNT).
- If none of BUFLIST, BUFFER, or ADJAREA is specified, only list entry controls are returned.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an area (with a length of ANSLEN) that will contain the information returned when the request completes.

**,ANSLEN=*anslen***

Use this input parameter to specify the size of the storage area specified by ANSAREA.

Check the prologue of the IXLYLAA mapping macro for the minimum required size of the answer area, or use the length field (LAA\_LEN) in the LAA.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 2-byte field that contains the size, in bytes, of the answer area.

**,AUTHCOMP=NO\_AUTHCOMP**

**,AUTHCOMP=*authcomp***

Use this input parameter to specify a value to be compared to the list authority value of the list specified by LISTNUM. You must supply the LISTNUM parameter.

If the comparison does not meet the condition specified by the AUTHCOMPTYPE parameter (EQUAL or LESSOREQUAL), the request fails.

**Note:** The AUTHCOMP parameter is valid only for list structures allocated in a coupling facility with CFLEVEL=1 or higher.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12 2 to ) of the 16-byte field that contains the list authority value.

**,AUTHCOMPTYPE=EQUAL**

**,AUTHCOMPTYPE=LESSOREQUAL**

Use this input parameter specify how the list audit comparison is to be performed.

**EQUAL**

The list authority for the list specified by LISTNUM must be equal to the value specified for AUTHCOMP.

**LESSOREQUAL**

The list authority for the list specified by LISTNUM must be less than or equal to the value specified for AUTHCOMP.

**Note:** The AUTHCOMPTYPE parameter is valid only for list structures allocated in a coupling facility with CFLEVEL=1 or higher.

**,BUFADDRTYPE=VIRTUAL**

**,BUFADDRTYPE=REAL**

Use this input parameter to specify whether the buffer addresses specified in the BUFLIST list are virtual storage or real storage addresses.

**VIRTUAL**

The buffer addresses are virtual storage addresses. The virtual storage can be pageable or nonpageable. See the PAGEABLE parameter for information about managing storage binds when specifying virtual storage addresses.

**REAL**

The buffer addresses are real storage addresses.

It is the caller's responsibility to manage the binds between the data buffer virtual storage and the real storage addresses provided. The caller must ensure that the data buffer virtual storage remains bound to the real storage addresses provided until the request completes.

**,BUFALET=NO\_BUFALET**

**,BUFALET=*bufalet***

Use this input parameter to specify an access list entry token (ALET) to be used in referencing all of the buffers specified by BUFLIST.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 4-byte field that contains the ALET.

**,BUFFER=buffer**

Use this output parameter to specify a buffer area to hold the entry data that is read from the designated entry.

You can define the buffer size to be a total of up to 65536 bytes. Depending on the size you select, the following restrictions apply:

- If you specify a buffer size of less than or equal to 4096 bytes, you must ensure that the buffer:
  - Is 256, 512, 1024, 2048, or 4096 bytes.
  - Starts on a 256-byte boundary.
  - Does not cross a 4096-byte boundary.
- If you specify a buffer size of greater than 4096 bytes, you must ensure that the buffer:
  - Is a multiple of 4096.
  - Is less than or equal to 65536 bytes.
  - Starts on a 4096-byte boundary.

See the BUFSIZE parameter description for defining the size of the buffer.

**Note:** You cannot code BUFFER with MODE=ASYNCRESPONSE.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an area (with a length of BUFSIZE) that will contain the entry data when the request completes.

**,BUFINCRNUM=bufincrnum**

Use this input parameter to specify the number of 256-byte segments comprising each buffer in the BUFLIST list.

Valid BUFINCRNUM values are 1, 2, 4, 8, or 16, which correspond to BUFLIST buffer sizes of 256, 512, 1024, 2048, and 4096 bytes respectively.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 1-byte field that contains 1, 2, 4, 8, or 16.

**,BUFLIST=buflist**

Use this output parameter to specify a list of buffers to hold the entry data that is read from the designated entry. BUFLIST specifies a 128-byte storage area that consists of a list of 0 to 16 buffer addresses.

**The 128-byte storage area must:**

- Consist of 0 to 16 elements.
- Each element must consist of an 8-byte field in which:
  - The left (high-order) four bytes are reserved and
  - The right (low-order) four bytes contain the address of a buffer.

**The BUFLIST buffers must:**

- Reside in the same address space or data space.
- Be the same size: either 256, 512, 1024, 2048, or 4096 bytes.
- Start on a 256-byte boundary and not cross a 4096-byte boundary.

**Note:** The buffers do not have to be contiguous in storage. List services treats BUFLIST buffers as a single buffer, even if the buffers are not contiguous.

See the BUFNUM and BUFINCRNUM keyword descriptions for specifying the number and size of buffers.

**Note:** You cannot code BUFLIST with MODE=ASYNCRESPONSE.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 128-byte area that contains a list of buffer addresses.

**,BUFNUM=*bufnum***

Use this input parameter to specify the number of buffers in the BUFLIST list. Valid BUFNUM values are 0 - 16. A value of zero indicates that no data is to be read into the buffers.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 1-byte field that contains the number of buffers (0 - 16) in the list (BUFLIST).

**,BUFSIZE=*bufsize***

Use this input parameter to specify the size of the BUFFER area. See the BUFFER parameter description for valid buffer sizes.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a fullword field that contains the size of the buffer (BUFFER) in bytes.

**,BUFSTGKEY=CALLERS\_KEY**

**,BUFSTGKEY=*bufstgkey***

Use this input parameter to specify a storage key that you define and use when referencing the buffers specified by BUFLIST or the buffer that is specified by BUFFER.

If you do not specify BUFSTGKEY, or if you specify BUFSTGKEY=CALLERS\_KEY, all references to one or more buffers are performed by using the caller's PSW key.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of an 8-bit field that contains the storage key in the format B'kkkkxxxx', where kkkk is the key and xxxx is ignored.

**,CONTOKEN=*contoken***

Use this input parameter to specify the connect token that was returned by the IXLCONN service. The connect token uniquely identifies your connection to the list structure, and must be specified on each IXLLIST invocation.

The connect token is available in the IXLCONN answer area that is mapped by IXLYCONA.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 16-byte field that contains the connect token.

**,CURSORUPDTYPE=NEXT**

**,CURSORUPDTYPE=NEXTCOND**

**,CURSORUPDTYPE=CURRENT**

**,CURSORUPDTYPE=CURRENTCOND**

Use this input parameter to specify how the list cursor is to be updated when UPDATECURSOR=YES is specified.

**Note:** The NEXTCOND, CURRENT, and CURRENTCOND values are valid only for list structures allocated in a coupling facility with CFLEVEL=1 or higher.

**CURSORUPDTYPE=NEXT**

Update the list cursor to point to the list entry before or after the target entry.

- For MOVE requests that specify DATAOPER=WRITE and ENTRYTYPE=ANY and that result in the creation of a new entry, the cursor for the list specified by MOVETOLIST is updated. The direction of the cursor update depends on the value that is specified for MOVETOPOS.
- For all other requests, the cursor for the source list is updated. The direction of the cursor update depends on the value that is specified for LISTPOS or LISTDIR.

**CURSORUPDTYPE=NEXTCOND**

Update the list cursor to point to the list entry before or after the target entry only if the cursor points to the target list entry, and that list entry is moved or deleted.

Set the list cursor direction with SETCURSOR on a WRITE\_LCONTROLS request.

The list cursor is reset to binary zeros if either:

- The entry is the last entry on the list and the list cursor direction is set to a head-to-tail direction.

- The entry is the first entry on the list and the list cursor is set to a tail-to-head direction.

**CURSORUPDTYPE=CURRENT**

Set the list cursor to point to the list entry processed for the request.

If the list entry is deleted or moved to another list, then the list cursor is reset to binary zeros.

**CURSORUPDTYPE=CURRENTCOND**

Set the list cursor to point to the list entry processed only if the list cursor value currently is zero and the target list entry is not deleted or moved to another list.

If the list entry is deleted or moved to another list, then the list cursor will remain binary zeros.

**,ENTRYID=entryid**

Use this input parameter to designate the entry identifier of the entry to be read.

Each entry identifier, which is assigned by list services when the entry is created, is unique within the structure.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 12-byte field that contains the entry identifier.

**,ENTRYKEY=NO ENTRYKEY****,ENTRYKEY=entrykey**

Use this input parameter with LISTNUM to designate the entry key of the entry to be read.

If there is a sublist of entries on the list all with the same key as the ENTRYKEY key, the LISTPOS parameter determines whether entry at the HEAD or the TAIL of the sublist is designated.

Specify ENTRYKEY only for structures that support keyed entries.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-byte field that contains the entry key.

**,ENTRYNAME=entryname**

Use this input parameter to designate the name of the entry to be read.

Specify ENTRYNAME only for structures that support named entries. Each entry name is unique within the structure.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-byte field that contains the entry name.

**,KEYREQTYPE=EQUAL****,KEYREQTYPE=LESSOREQUAL****,KEYREQTYPE=GREATEROREQUAL**

Use this input parameter to specify how, if at all, the entry key of the entry to be read can differ from the entry key that is specified by ENTRYKEY. KEYREQTYPE applies only to locating an existing entry; it does not determine the key of a new entry.

**EQUAL**

The entry must have a key that equals the ENTRYKEY key.

**LESSOREQUAL**

The entry must have a key that is less than or equal to the ENTRYKEY key.

**GREATEROREQUAL**

The entry must have a key that is greater than or equal to the ENTRYKEY key.

**Note:**

1. When no entries on the list meet the requirements of the KEYREQTYPE, the request is failed with a return code X'8' and reason code IXLRSNCODEOENTRY.
2. When LESSOREQUAL or GREATEROREQUAL is specified, if no entries on the list have an entry key value equal to the specified value, but entries exist with an entry key value greater than (if GREATEROREQUAL was specified), or less than (if LESSOREQUAL was specified) the entry key value that is specified, the entry with an entry key value closest to the value specified will



be selected. When multiple entries have the same entry key value, LISTPOS is used to resolve whether the first, or last entry with the entry key value is selected.

**,LISTDIR=TOTAL**

**,LISTDIR=TOHEAD**

Use this input parameter with UPDATECURSOR to specify the direction in which the list cursor is moved as a result of this request. See the UPDATECURSOR parameter for a full description of LISTDIR.

**,LISTNUM=NO\_LISTNUM**

**,LISTNUM=listnum**

Use this input parameter to specify the number of the list on which the entry to be read resides. Use LISTNUM in the following ways:

- With LISTPOS and/or ENTRYKEY to designate the entry to be read
- With either ENTRYID or ENTRYNAME to verify that the designated entry resides on the LISTNUM list before proceeding with the request
- With LOCBYCURSOR to designate the entry to be read

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 4-byte field that contains the number of the list.

**,LISTPOS=HEAD**

**,LISTPOS=TAIL**

Use this input parameter with LISTNUM to designate the entry to be read. The designated entry is at the HEAD or the TAIL of a list or sublist:

- If you specify ENTRYKEY and the list contains a sublist of one or more entries with the same key (or that satisfies the KEYREQTYPE criteria), then:
  - LISTPOS=HEAD designates the entry at the head of the sublist.
  - LISTPOS=TAIL designates the entry at the tail of the sublist.
- If you do not specify ENTRYKEY, then:
  - LISTPOS=HEAD designates the entry at the head of the list.
  - LISTPOS=TAIL designates the entry at the tail of the list.

**,LOCBYCURSOR**

Use this input parameter with LISTNUM to designate the entry to be read. The designated entry is the entry to which the LISTNUM list cursor is pointing.

Be aware that the list cursor could have been reset to zeros by a previous request that, for instance, deleted or moved the entry to which the list cursor points, or updated the list cursor after the last entry on the list had been processed. See *z/OS MVS Programming: Sysplex Services Guide* for more information about using the list cursor.

**,LOCKCOMP=NO\_LOCKCOMP**

**,LOCKCOMP=lockcomp**

This parameter has slightly different meanings based on the value that is specified for the LOCKOPER parameter. Generally, this input parameter specifies a connection identifier to be verified as the owner of the lock specified by LOCKINDEX. This verification is a prerequisite to the successful completion of the request.

When LOCKCOMP is specified, the completion of the request is conditional on there being no contention for the lock. If contention exists, the request fails .

The connection identifier is available from the IXLCONN answer area, which is mapped which is by IXLYCONA, in field CONACONID.

The effect of LOCKCOMP is based on the LOCKOPER value specified. See the description under LOCKOPER to see how each request is affected by this parameter.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 1-byte field that contains the connection identifier.

**,LOCKDATA=NO LOCKDATA****,LOCKDATA=lockdata**

Use this input parameter to specify information that is to be passed to your notify exit when another user requests the LOCKINDEX lock after you have obtained the lock by using LOCKOPER=SET. This user-defined information will be passed to your notify exit when the other user issues a request specifying either one of the following:

- LOCKOPER=SET
- LOCKOPER=NOTHELD with LOCKMODE=UNCOND

**To Code:** Specify the RS-type name or address (by using a register 2 - 12 2 to ) of an 8-byte field that contains the user-defined information.

**,LOCKINDEX=NO LOCKINDEX****,LOCKINDEX=lockindex**

Use this input parameter to specify the index of the lock to be operated on as specified by LOCKOPER. The lock indexes begin with zero.

**Note:** You cannot code LOCKINDEX with MODE=ASYNCSUSPEND.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 4-byte field that contains the lock index.

**,LOCKMODE=UNCOND****,LOCKMODE=COND**

Use this input parameter to specify how contention for the lock that is specified by LOCKINDEX should be handled if your request causes lock contention.

**UNCOND**

If the specified lock is held by another user, your request is either:

- Suspended until the lock becomes available (if MODE=SYNCSUSPEND is specified).
- Performed asynchronously with notification to you when the request completes (if any MODE value other than SYNCSUSPEND is specified).

**COND**

The lock operation is performed conditionally; that is, only if there is no contention for the lock. If the specified lock is held by another user, the request is terminated with no change to the structure, and return and reason codes describing the termination are returned to the caller.

**,LOCKOPER=SET****,LOCKOPER=RESET****,LOCKOPER=NOTHELD****,LOCKOPER=HELD BY**

Use this input parameter to specify the type of operation to be performed on the lock specified by LOCKINDEX.

**SET**

Set the lock.

- When LOCKCOMP is not specified, your connection gets the lock, providing no other connection holds the lock. If another connection holds the lock, the request either continues once the lock is released or fails, depending on the LOCKMODE value you specify.
- When LOCKCOMP is specified, if the connection specified by LOCKCOMP holds the lock, the lock is transferred to your connection.

**RESET**

Reset the lock.

- When LOCKCOMP is not specified, the lock is released if it is held by your connection.
- When LOCKCOMP is specified, the lock is released if it is held by the connection that is specified by LOCKCOMP.

**NOTHELD**

The request is performed only if the lock is not held by any connection. This option also ensures that the lock remains free for the duration of the request. If another connection holds the lock, your task is suspended until the lock is released or your request fails, depending on the LOCKMODE value you specify. See the LOCKMODE description for how to handle possible lock contention.

**HELD BY**

Processing is determined by which connector is holding the lock.

- When LOCKCOMP is not specified, the request is performed only if your connection holds the lock.
- When LOCKCOMP is specified, the request is performed only if the lock is held by the connection that is specified by LOCKCOMP.

**,MF=S****,MF=(L,*mfctrl*)****,MF=(L,*mfctrl*,*mfattr*)****,MF=(L,*mfctrl*,0D)****,MF=(M,*mfctrl*)****,MF=(M,*mfctrl*,COMPLETE)****,MF=(M,*mfctrl*,NOCHECK)****,MF=(E,*mfctrl*)****,MF=(E,*mfctrl*,COMPLETE)****,MF=(E,*mfctrl*,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

**,*mfctrl***

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

**,*mfattr***

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

**,COMPLETE****,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

**COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if SMILE=*var* were an optional parameter and the default is SMILE=NO\_SMILE

then it would not be documented. However, if the default was SMILE=:-), then it would be documented because a value would be the default.

**NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

**,MODE=SYNCSUSPEND**

**,MODE=SYNCECB**

**,MODE=SYNCEXIT**

**,MODE=SYNCTOKEN**

**,MODE=ASYNCECB**

**,MODE=ASYNCEXIT**

**,MODE=ASYNCTOKEN**

**,MODE=ASYNCSNORESPONSE**

Use this input parameter to specify:

- Whether the request is to be performed synchronously or asynchronously
- How you want to be notified of request completion

**MODE=SYNCSUSPEND**

The request is suspended until it can complete synchronously. To use this option, your program must be enabled for I/O and external interrupts.

**MODE=SYNCECB**

The request processes synchronously if possible. If the request processes asynchronously, the ECB specified by REQECB is posted when the request completes.

**MODE=SYNCEXIT**

The request processes synchronously if possible. If the request processes asynchronously, your complete exit is given control when the request completes.

**MODE=SYNCTOKEN**

The request processes synchronously if possible. If the request processes asynchronously, an asynchronous request token is returned in the area specified by REQTOKEN on invocation of the request. Use the returned request token on the IXLFComp macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=SYNCTOKEN is specified.

**MODE=ASYNCECB**

The request processes asynchronously. The ECB specified by REQECB is posted when the request completes.

**MODE=ASYNCEXIT**

The request processes asynchronously. Your complete exit is given control when the request completes.

**MODE=ASYNCTOKEN**

The request processes asynchronously. An asynchronous request token is returned in the area specified by REQTOKEN upon invocation of the request. Use the returned request token on the IXLFComp macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=ASYNCTOKEN is specified.

**MODE=ASYNCSNORESPONSE**

The request processes asynchronously. No notification of request completion is provided. The answer area (ANSAREA) fields will not contain valid information when this mode is specified.

**Note:** You cannot code MODE=ASYNCSNORESPONSE with LOCKINDEX, BUFFER, or BUFLIST.

**,NEWAUTH=NO\_NEWAUTH**

**,NEWAUTH=newauth**

Use this input parameter to specify a list authority value for the list specified by LISTNUM. If NEWAUTH is omitted, the list authority for the designated list is unchanged.

When the structure is allocated, the authority value for each list is initialized to binary zeros.

**Note:** The NEWAUTH parameter is valid only for list structures allocated in a coupling facility with CFLEVEL=1 or higher, except on WRITE\_LCONTROLS requests.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12 to ) of the 16-byte field that contains the list authority value.

**,NEWVERS=newvers**

Use this input parameter with VERSUPDATE=SET to specify a new version number to replace the old version number of the entry to be read.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-byte field that contains the entry version number.

**,PAGEABLE=YES**

**,PAGEABLE=NO**

Use this input parameter to identify whether the storage areas specified by BUFFER or BUFLIST are in pageable or potentially pageable storage.

**YES**

Specify this option to indicate that the BUFFER or BUFLIST buffers reside in pageable virtual storage. XES performs the required page fixing to fix the buffers in real storage while the cache or list request transfers data to or from the coupling facility.

This includes storage obtained from pageable subpools, disabled reference (DREF) storage, and may include storage that has the potential to become pageable during the processing of a request. (An example is address space storage owned by any swappable address space, for which a PGSER FIX has been successfully processed, but for which the owning address space gets swapped during processing of a cache or list request.) This does not include implicitly non-pageable storage (for example, storage obtained from non-pageable subpools).

The system takes responsibility for managing binds to central storage for the duration of the cache or list request, regardless of what address space owns the storage or whether the storage-owning address space is swappable or nonswappable. The storage can be owned by any address space.

High shared virtual storage areas (above 2GB) may not be used.

**NO**

Specify this option to indicate that the BUFFER or BUFLIST buffers reside in non-pageable virtual storage. XES does not page fix the buffers in real storage.

This includes implicitly non-pageable storage areas (for example, storage obtained from non-pageable subpools), and may include storage that has the potential to become pageable during the processing of a request (An example is address space storage owned by any swappable address space, for which a PGSER FIX has been successfully processed, but for which the owning address space gets swapped-out during processing of a cache or list request.)

The system takes responsibility for managing binds to central storage for the duration of the cache or list request, if and only if the non-pageable storage is owned by either the requester's address space or the connector's address space. If the storage is owned by any other address space, then the invoker is responsible for ensuring that the virtual storage remains non-pageable for the duration of the request (including the case in which the storage is owned by a swappable address space that is swapped during processing of an IXLCACHE or IXLLIST request). Subject to this consideration, the storage can be owned by any address space. See [z/OS MVS Programming: Sysplex Services Guide](#).

**,PLISTVER=IMPLIED\_VERSION**

**,PLISTVER=MAX**

**,PLISTVER=plistver**

Use this input parameter to specify the version of the macro. See [“Understanding IXLLIST Version Support” on page 965](#) for a description of the options available with PLISTVER.

**,REQDATA=NO REQDATA****,REQDATA=reqdata**

Use this input parameter with MODE=SYNCEXIT or MODE=ASYNCEXIT to pass any data you choose to the complete exit. The exit will get control only if the request is processed asynchronously.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the data to be passed to the complete exit.

**,REQECB=reqecb**

Use this output parameter with either MODE=SYNCECB or MODE=ASYNCECB to specify the address of an ECB, which is to be posted when the request completes if the request was processed asynchronously.

Before coding REQECB, you must ensure that:

- You initialize the ECB before you issue the request.
- The ECB resides in either common storage or the home address space where IXLCONN was issued.
- Any tasks that wait for the ECB to be posted reside in the home address space where IXLCONN was issued.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that contains the address of the ECB to be posted when the request completes. The ECB must be aligned on a fullword boundary.

**,REQID=NO REQID****,REQID=reqid**

Use this input parameter to specify a user-defined request identifier to be associated with the request. You can specify this request identifier on the IXPURGE macro to cancel a request that has not yet been processed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the user-defined request identifier.

**,REQTOKEN=reqtoken**

Use this output parameter with either MODE=SYNCTOKEN or MODE=ASYNCTOKEN to specify the address of a storage area to receive the request token that is returned when the request will be processed asynchronously. This token, which uniquely identifies the request, must be used as input to the IXLFComp macro, which you use to determine if the request has completed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field where the system will put the request token.

**,RETCODE=retcode**

Use this output parameter to specify a field to contain the return code. (The return code is also returned in register 15.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the return code when the request has completed.

**,RSNCODE=rsncode**

Use this output parameter to specify a field to contain the reason code returned, if applicable. (The reason code is also returned in register 0.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the reason code (if any) when the request has completed.

**,UPDATECURSOR=NO****,UPDATECURSOR=YES**

Use this input parameter to specify whether the LISTNUM list cursor is to be updated to point to another entry on the list.

**NO**

The list cursor is not updated.

**YES**

The list cursor is updated to point to the entry that is adjacent to the entry that is read:

- If LISTDIR=TOHEAD, the list cursor will point to the adjacent entry that is closest to the head of the list.
- If LISTDIR=TOTAIL, the list cursor will point to the adjacent entry that is closest to the tail of the list.

The list cursor is set to binary zeros if its new position would be before the first entry or after the last.

**,VERSCOMP=NO\_VERSCOMP****,VERSCOMP=verscomp**

Use this input parameter to specify a version number to be compared to the version number of the designated entry to be read. The entry is read only if its version number meets the condition specified by the VERSCOMPTYPE parameter. If the designated entry does not have the specified version number, the request is terminated with no change to the structure.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-byte field that contains the version number.

**,VERSCOMPTYPE=EQUAL****,VERSCOMPTYPE=LESSOREQUAL**

Use this input parameter to specify how a list entry version comparison as specified by VERSCOMP is to be performed.

**Note:** The VERSCOMPTYPE parameter is valid only for list structures allocated in a coupling facility with CFLEVEL=1 or higher.

**VERSCOMPTYPE=EQUAL**

The version number for the list entry must be equal to the value specified for VERSCOMP.

**VERSCOMPTYPE=LESSOREQUAL**

The version number for the list entry must be less than or equal to the value specified for VERSCOMP.

**,VERSUPDATE=NONE****,VERSUPDATE=INC****,VERSUPDATE=DEC****,VERSUPDATE=SET**

Use this input parameter to specify how the version number of the designated entry will be updated.

**VERSUPDATE=NONE**

The version number is not updated.

**VERSUPDATE=INC**

The version number is increased by one.

**VERSUPDATE=DEC**

The version number is decreased by one.

**VERSUPDATE=SET**

The version number is set to the value specified by NEWVERS.

## ABEND Codes

---

Abend X'026' (See [z/OS MVS System Codes](#) for more information on this abend.)

## Return and Reason Codes

---

When the system returns control to your program:

- GPR 15 (and RETCODE, if specified) contains a return code.
- If the return code is not zero, GPR 0 (and RSNODE, if specified) contains a reason code.

**Note:** The return and reason codes will also be put into the answer area (mapped by IXLYLAA), if there is one, when the request completes.

The IXLYCON macro provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

**0**

IXLRETCODEOK

**4**

IXLRETCODEWARNING

**8**

IXLRETCODEPARMERROR

**C**

IXLRETCODEENVEERROR

**10**

IXLRETCODECOMPERROR

The following table contains hexadecimal return and reason codes, the equate symbols associated with each reason code, and the meaning and suggested action for each return and reason code.

Table 67. Return and Reason Codes for IXLLIST REQUEST=READ Macro		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0	None.	<p><b>Meaning:</b> If you specified a MODE value of ASYNCECB, ASYNCEXIT, ASYNCTOKEN, or ASYNCNORESPONSE, the request has been successfully initiated. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed. For any other MODE value, the request has successfully completed.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=ASYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=ASYNCEXIT, the connection's complete exit will be given control when the request has completed.</li> <li>• If you specified MODE=ASYNCTOKEN, use REQTOKEN when calling IXLFCOMP to determine when the request has completed.</li> </ul>



Table 67. Return and Reason Codes for IXLLIST REQUEST=READ Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	xxxx0402	<p><b>Equate Symbol:</b> IXLRSNCODEASYNCH</p> <p><b>Meaning:</b> The request could not be processed synchronously. It will be processed asynchronously. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed.</p> <ul style="list-style-type: none"> <li>• If MODE=SYNCECB was specified, the ECB specified for REQECB will be posted when the request has finished.</li> <li>• If MODE=SYNCEXIT was specified, the connection's complete exit will be given control when the request has finished.</li> <li>• If MODE=SYNCTOKEN was specified, an asynchronous request token was returned in the area specified by REQTOKEN. This request token may be specified on the IXLFCOMP macro to determine when the request has finished.</li> </ul> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=SYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=SYNCEXIT, the connection's complete exit will be given control when the request has completed.</li> <li>• If you specified MODE=SYNCTOKEN, use REQTOKEN when calling IXLFCOMP to determine when the request has completed.</li> </ul>

Table 67. Return and Reason Codes for IXLLIST REQUEST=READ Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	xxxx040D	<p><b>Equate Symbol:</b> IXLRSNCODEBADREADADJDATA</p> <p><b>Meaning:</b> Program error. The request specified that adjunct data was to be read, but the storage area specified by ADJAREA is not addressable. All other requested data was retrieved and the following fields in the answer area have been filled in:</p> <ul style="list-style-type: none"> <li>• LAATOTALCNT - The total count of allocated entries in the list structure.</li> <li>• LAATOTALELECNT - The total count of allocated elements in the list structure.</li> <li>• LAALISTCNT - The count of entries or elements residing on the processed list.</li> </ul> <p><b>Note:</b> Only the adjunct data for the first entry in the list is placed in the ADJAREA. The buffers should contain the adjunct data for the other entries in the list.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the ADJAREA address.</li> <li>• ADJAREA must be addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are calling IXLLIST while disabled, ADJAREA must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLLIST in AR-mode and you specified the ADJAREA address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are calling IXLLIST in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLIST macro.</li> </ul> <p>Correct the address specified by ADJAREA, and rerun the request asking for adjunct data only.</p>
4	xxxx0410	<p><b>Equate Symbol:</b> IXLRSNCODELOCKCOND</p> <p><b>Meaning:</b> For a LOCKOPER=HELD BY request, or a LOCKMODE=COND request, or a request that specified LOCKCOMP, the request could not be completed successfully because the specified lock is not currently held as required. The connection identifier of the lock owner is returned in the answer area (LAACONID field).</p> <p><b>Action:</b> Retry the request, or obtain the lock as required, and retry the request. If you are unable to get the lock, check the ID in the LAACONID field and determine if some recovery is necessary.</p>

Table 67. Return and Reason Codes for IXLLIST REQUEST=READ Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	xxxx0412	<p><b>Equate Symbol:</b> IXLRSNCODELOCKHELDDBYSYS</p> <p><b>Meaning:</b> The lock is not held by any connection, but instead is held by the system. For a request that specified any of the following, the request could not be completed successfully because the specified lock is not currently held as required:</p> <ul style="list-style-type: none"> <li>• LOCKOPER=SET or LOCKOPER=NOTHELD with either of: <ul style="list-style-type: none"> <li>– LOCKMODE=COND</li> <li>– LOCKCOMP</li> </ul> </li> <li>• LOCKOPER=HELDDBY</li> </ul> <p><b>Action:</b> Retry the request, or obtain the lock as required, and retry the request.</p>
8	xxxx0801	<p><b>Equate Symbol:</b> IXLRSNCODEBADPARMLIST</p> <p><b>Meaning:</b> Program error. The parameter list for this request is not addressable. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Verify that:</p> <ul style="list-style-type: none"> <li>• The parameter list address is uncorrupted.</li> <li>• The parameter list is addressable in the caller's primary address space.</li> <li>• If you are issuing this macro while disabled, the parameter list resides in either page-fixed or DREF storage.</li> <li>• If you are issuing this macro in AR-mode and you specified the parameter list address using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If you are issuing this macro in AR-mode, SYSSTATE ASCENV=AR must be issued before issuing this macro.</li> </ul>
8	xxxx0804	<p><b>Equate Symbol:</b> IXLRSNCODEBADVERSIONNUM (or IXLRSNCODEBADVERSION#)</p> <p><b>Meaning:</b> The version number in the macro parameter list is not compatible with the level of XES currently being used. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that your program did not overlay the parameter list storage.</li> <li>• Verify that your program was assembled with the correct macro library for the release of MVS on which your program is running.</li> </ul>

Table 67. Return and Reason Codes for IXLLIST REQUEST=READ Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx080A	<p><b>Equate Symbol:</b> IXLRSNCODEBADCONTOKEN</p> <p><b>Meaning:</b> Program error. The specified connect token (CONTOKEN) was not valid for one of the following reasons:</p> <ol style="list-style-type: none"> <li>1. The user with the connection identifier represented by the token has disconnected from the structure.</li> <li>2. The connector's task (the task that issued IXLCONN) ended.</li> <li>3. The specified token is not the token that was returned from IXLCONN.</li> <li>4. The request was issued from an address space other than the address space in which IXLCONN was issued.</li> <li>5. The connect token was invalidated during rebuild.</li> <li>6. The connect token was invalidated by XES.</li> </ol> <p><b>Note:</b> The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Take the action with the corresponding meaning.</p> <ol style="list-style-type: none"> <li>1. Discontinue use of the structure. Perform recovery and cleanup for the structure and check the protocol for the use of the structure to determine why you did not know of the disconnection.</li> <li>2. Discontinue use of the structure. Perform recovery and cleanup for the structure and check the protocol for the use of the structure to determine why you did not know that the task ended.</li> <li>3. The contoken is returned in the CONACONTOKEN field of the answer area specified on the IXLCONN request.</li> <li>4. Issue your request from the same address space the IXLCONN was issued in.</li> <li>5. Wait for the rebuild to complete, and try again.</li> <li>6. Discontinue use of the structure. Perform recovery and cleanup for the structure.</li> </ol>
8	xxxx0824	<p><b>Equate Symbol:</b> IXLRSNCODEWRONGSTRTYPE</p> <p><b>Meaning:</b> Program error. The connection specified by CONTOKEN is not to a list structure.</p> <p><b>Action:</b> Specify the appropriate CONTOKEN value, and rerun your program. The CONTOKEN is returned in the answer area provided on the IXLCONN invocation. The type of structure built is passed on the TYPE parameter of the IXLCONN macro.</p>

Table 67. Return and Reason Codes for IXLLIST REQUEST=READ Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0825	<p><b>Equate Symbol:</b> IXLRSNCODENOENTRY</p> <p><b>Meaning:</b> Program error. The designated list entry does not exist; therefore, no entries were processed.</p> <p><b>Action:</b> None necessary. However, if this return code and reason code are unexpected, you should check the way the list entry was created. Examine the parameters specified for locating the list entry on the invocation of this macro.</p>
8	xxxx0833	<p><b>Equate Symbol:</b> IXLRSNCODEBADPGBLATTR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list is specified as being pageable (PAGEABLE=YES), but is not.</p> <p><b>Action:</b> Change the buffer area(s) to pageable storage, or specify PAGEABLE=NO. See the PAGEABLE parameter description for specification instructions.</p>
8	xxxx0834	<p><b>Equate Symbol:</b> IXLRSNCODEBADNONPGBLATTR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list, is specified as being nonpageable (PAGEABLE=NO), but is either pageable or not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The type of storage in the buffer area corresponds to the value you specified on the PAGEABLE parameter. See the PAGEABLE parameter description for specification instructions.</li> <li>• The correct buffer address was used.</li> <li>• The buffer area was not previously freed.</li> <li>• If you are calling IXLLIST while disabled, the buffers must reside in either page-fixed or DREF storage.</li> <li>• The buffer areas were allocated in a storage key that matches the key specified by the BUFSTGKEY parameter, or, if BUFSTGKEY is omitted, the caller's PSW key.</li> <li>• If BUFLIST was specified and your program is running in AR-mode: <ul style="list-style-type: none"> <li>– If the BUFFER or BUFLIST parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>– The BUFALET specification is correct.</li> <li>– You specified SYSSTATE ASCENV=AR before issuing the IXLLIST macro.</li> </ul> </li> </ul>

Table 67. Return and Reason Codes for IXLLIST REQUEST=READ Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0835	<p><b>Equate Symbol:</b> IXLRNCODEBADDATAADDR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list, is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The correct buffer address was used.</li> <li>• The buffer area was not previously freed.</li> <li>• The buffer area was allocated in a storage key that matches the key specified using the BUFSTGKEY parameter, or, if BUFSTGKEY is omitted, the caller's PSW key.</li> <li>• If BUFLIST was specified and your program is running in AR mode: <ul style="list-style-type: none"> <li>– The BUFALET specification is correct.</li> <li>– You specified SYSSTATE ASCENV=AR before issuing the macro.</li> </ul> </li> </ul>
8	xxxx0836	<p><b>Equate Symbol:</b> IXLRNCODEBADREALADDR</p> <p><b>Meaning:</b> Program error. Real storage addresses were provided in the BUFLIST list, but one of the buffers is not addressable in central storage.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that BUFADDRTYPE was specified as you intended.</li> <li>• Ensure that the buffer addresses specified by BUFLIST are valid.</li> </ul>
8	xxxx0838	<p><b>Equate Symbol:</b> IXLRNCODEBADANSAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by ANSAREA is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The answer area address specified by ANSAREA is valid.</li> <li>• If the caller is running in AR-mode and the ANSAREA parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• The address specified for ANSAREA is addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are calling IXLLIST while disabled, ANSAREA must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLLIST in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLIST macro.</li> </ul>

Table 67. Return and Reason Codes for IXLLIST REQUEST=READ Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0839	<p><b>Equate Symbol:</b> IXLRSNCODEBADREQTOKENAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by REQTOKEN is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The request token area specified by REQTOKEN is valid.</li> <li>• If the caller is running in AR-mode and the REQTOKEN parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If you are calling IXLLIST while disabled, REQTOKEN must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLLIST in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLIST macro.</li> </ul>
8	xxxx083D	<p><b>Equate Symbol:</b> IXLRSNCODEBADANSLEN</p> <p><b>Meaning:</b> Program error. The length of the answer area, as specified by ANSLEN, is not sufficient to contain answer area information. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Increase the size of the answer area, and rerun your program. The minimum size of the ANSAREA is in the LAA_LEN field of the IXLYLAA mapping macro.</p>
8	xxxx083F	<p><b>Equate Symbol:</b> IXLRSNCODEBADENTRYVERSION</p> <p><b>Meaning:</b> The designated entry has a version number that failed the comparison specified by VERSCOMPTYPE. The list entry controls for the entry are returned in the answer area (field LAALCTL).</p> <p><b>Action:</b> None necessary. However, if this return code and reason code are unexpected, you might want to verify the value specified in VERSCOMP and look at the version returned in the answer area.</p>
8	xxxx0840	<p><b>Equate Symbol:</b> IXLRSNCODEBADENTRYLIST</p> <p><b>Meaning:</b> The entry designated by ENTRYID or ENTRYNAME exists in the structure, but does not reside on the list specified by LISTNUM. The list entry controls for the entry are returned in the answer area (field LAALCTL).</p> <p><b>Action:</b> None necessary. However, if you did not expect this return and reason code, you might want to verify what list this entry is on. Maybe the entry was moved, or you designated the wrong list in LISTNUM. Check the protocol for using this list.</p> <p>The information returned in the LAALCTL field of the answer area contains the number of the list that this list entry is on as well as other control information.</p>

Table 67. Return and Reason Codes for IXLLIST REQUEST=READ Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0842	<p><b>Equate Symbol:</b> IXLRSNCODEPERSISTENTLOCK</p> <p><b>Meaning:</b> Program error. The request specifying LOCKOPER=SET with LOCKMODE=UNCOND, or LOCKOPER=NOTHELD with LOCKMODE=UNCOND failed because the lock is held by a connection that is in the failed-persistent state. The connection identifier of the lock owner is returned in the answer area (field LAACONID).</p> <p><b>Action:</b> Either perform recovery for the connection, or wait until recovery for the connection is performed.</p>
8	xxxx0845	<p><b>Equate Symbol:</b> IXLRSNCODENONAMES</p> <p><b>Meaning:</b> Program error. A list entry was designated by entry name, but the structure does not support entry names.</p> <p><b>Action:</b> Ensure that you are connected to the intended structure. Ensure that the correct specification was made for REFOPTION on the IXLCONN macro. You may want to issue IXLMG to get more information about the structure.</p>
8	xxxx0846	<p><b>Equate Symbol:</b> IXLRSNCODEBADLOCKINDEX</p> <p><b>Meaning:</b> Program error. The specified LOCKINDEX exceeds the size of the lock table for the structure.</p> <p><b>Action:</b> Correct LOCKINDEX to specify an index that is contained within the lock table. The maximum value for the LOCKINDEX is one less than the value specified by the LOCKENTRIES parameter on the IXLCONN macro.</p>
8	xxxx0847	<p><b>Equate Symbol:</b> IXLRSNCODEBADLISTNUMBER</p> <p><b>Meaning:</b> Program error. The specified LISTNUM value exceeds the number of lists for the structure.</p> <p><b>Action:</b> Correct LISTNUM to specify a list number that exists in the structure. The number of lists allocated for a structure is determined on the LISTHEADERS parameter of the IXLCONN macro. The list numbers go from 0 to n-1, where n is the number of lists specified.</p>
8	xxxx0848	<p><b>Equate Symbol:</b> IXLRSNCODEBADRESET</p> <p><b>Meaning:</b> Program error. LOCKOPER=RESET was specified for a lock not currently held by the invoker. The value of the connection ID holding the lock is returned in the answer area (field LAACONID).</p> <p><b>Action:</b> Check your code to ensure that your connection did not already reset the lock.</p>



Table 67. Return and Reason Codes for IXLLIST REQUEST=READ Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx084A	<p><b>Equate Symbol:</b> IXLRSNCODENOKEYS</p> <p><b>Meaning:</b> Program error. The structure does not support the use of entry keys. The IXLLIST request type either required the structure to support entry keys or designated a list entry or list position by list number and entry key.</p> <p><b>Action:</b> Ensure that you are connected to the intended structure. The use of keys for a structure is determined by the REFOPTION keyword on the IXLCONN macro.</p>
8	xxxx084B	<p><b>Equate Symbol:</b> IXLRSNCODENOLOCKS</p> <p><b>Meaning:</b> Program error. A locking operation was requested, but the structure does not contain a lock table.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• You are connected to the intended structure.</li> <li>• You intended to perform a locking operation.</li> </ul> <p>The use of locks is determined by the LOCKENTRIES keyword on the IXLCONN macro.</p>
8	xxxx0851	<p><b>Equate Symbol:</b> IXLRSNCODENOSUSPENDISABLE</p> <p><b>Meaning:</b> Program error. The request failed because MODE=SYNCSUSPEND was specified, but the caller is disabled and cannot be suspended.</p> <p><b>Action:</b> Either specify another MODE value, or become enabled (release the CPU lock), then reissue the request.</p>
8	xxxx0854	<p><b>Equate Symbol:</b> IXLRSNCODEBADLOCKCOMP</p> <p><b>Meaning:</b> Program error. The connection identifier specified for LOCKCOMP is not valid.</p> <p><b>Action:</b> The connection identifier is returned in the answer area (mapped by IXLYCONA) when the IXLCONN macro was issued.</p>
8	xxxx0859	<p><b>Equate Symbol:</b> IXLRSNCODEBADLISTAUTH</p> <p><b>Meaning:</b> The list authority specified for AUTHCOMP failed the comparison specified by AUTHCOMPTYPE for the list authority of the specified list. The current list authority (field LAALISTAUTH) and description (field LAALISTDESC) are returned in the answer area.</p> <p><b>Action:</b> None required; however you might want to take some action depending on your application. The list authority is set to binary zeros when the structure is allocated. When IXLLIST is issued with the NEWAUTH keyword, the list authority is changed if the correct comparison list authority value is specified. Check the answer area to determine the list authority value for the list specified. Verify that the correct list number was specified.</p>

Table 67. Return and Reason Codes for IXLLIST REQUEST=READ Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0864	<p><b>Equate Symbol:</b> IXLRSNCODEBADBUFSIZE</p> <p><b>Meaning:</b> Program error. The size of the BUFFER area or the buffer areas specified by BUFLIST is not large enough to contain the data for the first entry to be read in the list. No data is returned.</p> <p><b>Action:</b> If more space is available, specify a larger buffer size, or more buffers, and reissue the request. Check the information returned in the answer area (mapped by IXLYLAA) in the field LAALCTL (mapped by IXLYLCTL).</p>
8	xxxx0865	<p><b>Equate Symbol:</b> IXLRSNCODEBADBUFSPEC</p> <p><b>Meaning:</b> Program error. There is an error in the buffer specification.</p> <p><b>Action:</b> Check the following:</p> <ul style="list-style-type: none"> <li>• If BUFLIST was specified, check the requirements for BUFLIST, BUFNUM, and BUFINCRNUM.</li> <li>• If BUFFER was specified, check the requirements for BUFFER and BUFSIZE.</li> <li>• Buffer pointer(s) in BUFLIST</li> <li>• Buffer boundaries.</li> </ul>
8	xxxx0866	<p><b>Equate Symbol:</b> IXLRSNCODEBADBUFKEY</p> <p><b>Meaning:</b> Program error. The buffer storage key specified by BUFSTGKEY is incorrect or BUFSTGKEY was not specified and the PSW key does not match the key of the buffers.</p> <p>The data cannot be stored in the specified buffer area.</p> <p><b>Action:</b> Check the following:</p> <ul style="list-style-type: none"> <li>• Determine if the key of the storage being used for the buffers is different from the PSW key.</li> <li>• If BUFSTGKEY was specified, verify that the key was put in the correct bits (see the explanation of this parameter for more information).</li> <li>• If you are calling IXLLIST in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLIST macro.</li> </ul>
8	xxxx0867	<p><b>Equate Symbol:</b> IXLRSNCODEBADBUFLIST</p> <p><b>Meaning:</b> Program error. The 128-byte storage area specified by BUFLIST is not addressable.</p> <p><b>Action:</b> Ensure that the address specified by BUFLIST is valid.</p>

Table 67. Return and Reason Codes for IXLLIST REQUEST=READ Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C06	<p><b>Equate Symbol:</b> IXLRSNCODENOCNN</p> <p><b>Meaning:</b> Environmental error. This system does not have connectivity to the coupling facility that contains the list structure. Possible reasons for this are:</p> <ul style="list-style-type: none"> <li>• The operator issued VARY PATH,OFFLINE.</li> <li>• The operator issued CONFIG CHP,OFFLINE.</li> <li>• Hardware errors to the coupling facility.</li> <li>• Facility or path failure to the coupling facility.</li> </ul> <p><b>Action:</b> Begin rebuilding the structure on a different coupling facility, or disconnect from the structure.</p>
C	xxxx0C13	<p><b>Equate Symbol:</b> IXLRSNCODEREQPURGED</p> <p><b>Meaning:</b> Environmental error. The request was purged prior to completion of the request. Possible reasons include:</p> <ul style="list-style-type: none"> <li>• The connector failed.</li> <li>• The connector disconnected.</li> <li>• The requestor failed.</li> <li>• The request was purged by IXLPURGE.</li> <li>• Requests were purged when the connector provided an IXLEERSP response for the Rebuild Stop or Rebuild Cleanup event.</li> <li>• The secondary address space was no longer valid.</li> </ul> <p><b>Action:</b> None if this is expected. Otherwise, determine why the connector failed.</p>
C	xxxx0C14	<p><b>Equate Symbol:</b> IXLRSNCODESTATUSUNKNOWN</p> <p><b>Meaning:</b> Environmental error. The request has completed, but the final disposition of the request cannot be determined. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the validity of your data by comparing the expected results with what is in the coupling facility.</li> <li>• Consider running rebuild processing to move the structure to attempt to protect it from further errors of this kind.</li> </ul>
C	xxxx0C25	<p><b>Equate Symbol:</b> IXLRSNCODESTRFAILURE</p> <p><b>Meaning:</b> Environmental error. The list structure failed prior to completion of the request.</p> <p><b>Action:</b> Either rebuild or disconnect from the structure.</p>

Table 67. Return and Reason Codes for IXLLIST REQUEST=READ Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0CA0	<b>Equate Symbol:</b> IXLRSNCODEQUIESCEDSUSPENDFAIL <b>Meaning:</b> Environmental error. The request is failed because the structure is quiesced for a system-managed process and SUSPEND=FAIL is specified on the IXLCONN. <b>Action:</b> None, if this is expected.
C	xxxxFFFF	<b>Equate Symbol:</b> IXLRSNCODENOTAVAILABLE <b>Meaning:</b> Environmental error. XES functions are not available. This could occur because the coupling facility hardware necessary to provide XES functions is not present. <b>Action:</b> Re-IPL the system, or follow your particular failure management protocol.
10	xxxx10xx	<b>Meaning:</b> System error. XES processing failure. The state of the involved structure and the disposition of the request are unpredictable. The answer area (ANSAREA) fields do not contain valid information. <b>Action:</b> Contact the IBM support center.

## Chapter 68. IXLLIST REQUEST=READ\_EMCONTROLS

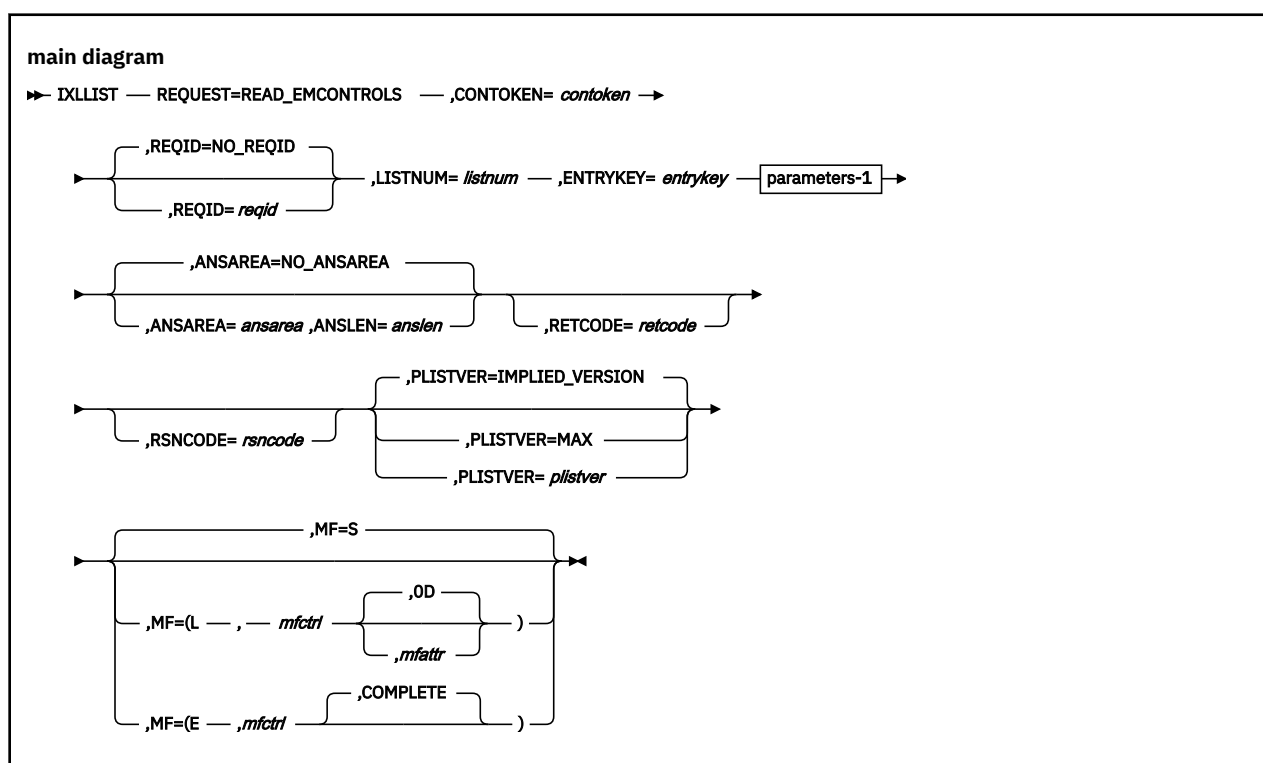
### Description

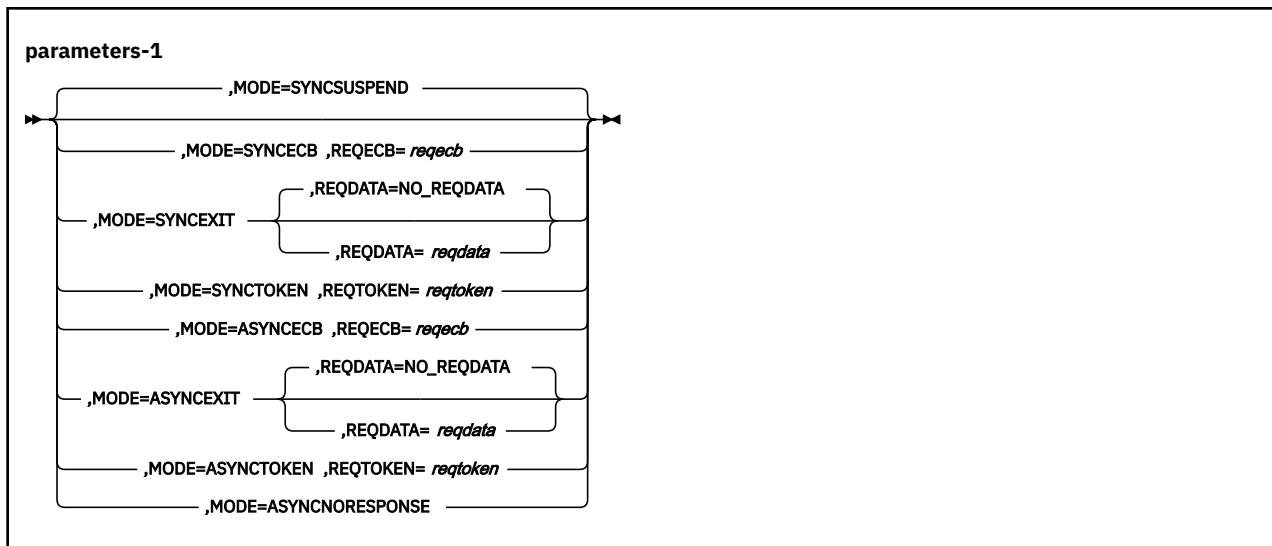
A READ\_EMCONTROLS request allows you to retrieve control information about your registered interest in a designated sublist within the list structure that you are monitoring. The sublist is identified by list number (LISTNUM) and entry key (ENTRYKEY). The information is returned in the answer area (ANSAREA).

You can issue a READ\_EMCONTROLS request only for keyed list structures allocated in a coupling facility of CFLEVEL=3 or higher. READ\_EMCONTROLS requests issued for a list structure allocated in a coupling facility of CFLEVEL=0, 1, or 2 will fail.

### Syntax Diagram

The syntax diagram for IXLLIST REQUEST=READ\_EMCONTROLS is as follows:





**Note:** If MODE=SYNCTOKEN or MODE=ASYNCTOKEN is specified, ANSAREA=*ansarea* and ANSLLEN=*anslen* are required.

## Parameter Descriptions

The parameter descriptions for REQUEST=READ\_EMCONTROLS are listed in alphabetical order. Default values are underlined:

### REQUEST=READ\_EMCONTROLS

Use this input parameter to specify that event monitor control information that represents your registered monitoring interest in the sublist specified by LISTNUM and ENTRYKEY be returned in the answer area.

#### **,ANSAREA=NO\_ANSAREA**

#### **,ANSAREA=*ansarea***

Use this output parameter to specify an answer area to contain information returned from the request. The format of the answer area is described by mapping macro IXLYLAA. Upon successful completion of a request for event monitor control information, the answer area contains:

- The connection identifier of the connector associated with the EMC (field LAAREMC\_CONID)
- An indicator as to whether the EMC is queued to the event queue of the connector identified by LAAREMC\_CONID (field LAAREMC\_EMQUEUED)
- The list number of the list with which the EMC is associated (field LAAREMC\_LISTNUM)
- The list entry key of the sublist with which the EMC is associated (field LAAREMC\_ENTRYKEY)
- The user notification control data that was supplied by the connector when this EMC was established to monitor the indicated sublist (field LAAREMC\_UNC).

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an area (with a length of ANSLLEN) to contain the information.

#### **,ANSLLEN=*anslen***

Use this input parameter to specify the size of the storage area specified by ANSAREA.

Check the prologue of the IXLYLAA mapping macro for the minimum required size of the answer area, or use the length field (LAA\_LEN) in the LAA.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 2-byte field that contains the size, in bytes, of the answer area.

#### **,CONTOKEN=*contoken***

Use this input parameter to specify the connect token that was returned by the IXLCONN service. The connect token uniquely identifies your connection to the list structure, and must be specified on each IXLLIST invocation.

The connect token is available in the IXLCONN answer area that is mapped by IXLYCONA.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 16-byte field that contains the connect token.

#### **,ENTRYKEY**

Use this input parameter to specify the entry key to be used in conjunction with LISTNUM to designate the sublist for which control information is to be returned.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-byte field that contains the entry key of the sublist.

#### **,LISTNUM=*listnum***

Use this input parameter to specify the number of the list containing the sublist for which control information is to be returned.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 4-byte field that contains the number of the list.

#### **,MF=S**

#### **,MF=(L,*mfctrl*)**

#### **,MF=(L,*mfctrl*,*mfattr*)**

#### **,MF=(L,*mfctrl*,0D)**

#### **,MF=(M,*mfctrl*)**

#### **,MF=(M,*mfctrl*,COMPLETE)**

#### **,MF=(M,*mfctrl*,NOCHECK)**

#### **,MF=(E,*mfctrl*)**

#### **,MF=(E,*mfctrl*,COMPLETE)**

#### **,MF=(E,*mfctrl*,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

#### **,*mfctrl***

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

#### **,*mfattr***

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

#### **,COMPLETE**

#### **,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

#### **COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if SMILE=*var* were an optional parameter and the default is SMILE=NO\_SMILE then it would not be documented. However, if the default was SMILE=-), then it would be documented because a value would be the default.

#### **NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

**,MODE=SYNCSUSPEND**

**,MODE=SYNCECB**

**,MODE=SYNCEXIT**

**,MODE=SYNCTOKEN**

**,MODE=ASYNCECB**

**,MODE=ASYNCEXIT**

**,MODE=ASYNCTOKEN**

**,MODE=ASYNCSNORESPONSE**

Use this input parameter to specify:

- Whether the request is to be performed synchronously or asynchronously
- How you want to be notified of request completion

#### **MODE=SYNCSUSPEND**

The request is suspended until it can complete synchronously. To use this option, your program must be enabled for I/O and external interrupts.

#### **MODE=SYNCECB**

The request processes synchronously if possible. If the request processes asynchronously, the ECB specified by REQECB is posted when the request completes.

#### **MODE=SYNCEXIT**

The request processes synchronously if possible. If the request processes asynchronously, your complete exit is given control when the request completes.

#### **MODE=SYNCTOKEN**

The request processes synchronously if possible. If the request processes asynchronously, an asynchronous request token is returned in the area specified by REQTOKEN on invocation of the request. Use the returned request token on the IXLFComp macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=SYNCTOKEN is specified.

#### **MODE=ASYNCECB**

The request processes asynchronously. The ECB specified by REQECB is posted when the request completes.

#### **MODE=ASYNCEXIT**

The request processes asynchronously. Your complete exit is given control when the request completes.

#### **MODE=ASYNCTOKEN**

The request processes asynchronously. An asynchronous request token is returned in the area specified by REQTOKEN upon invocation of the request. Use the returned request token on the IXLFComp macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=ASYNCTOKEN is specified.

#### **MODE=ASYNCSNORESPONSE**

The request processes asynchronously. No notification of request completion is provided. The answer area (ANSAREA) fields will not contain valid information when this mode is specified.

**Note:** You cannot code MODE=ASYNCSNORESPONSE with LOCKINDEX, BUFFER, or BUFLIST.



**,PLISTVER=IMPLIED\_VERSION****,PLISTVER=MAX****,PLISTVER=*plistver***

Use this input parameter to specify the version of the macro. See “[Understanding IXLLIST Version Support](#)” on page 965 for a description of the options available with PLISTVER.

**,REQDATA=NO\_REQDATA****,REQDATA=*reqdata***

Use this input parameter with MODE=SYNCEXIT or MODE=ASYNCEXIT to pass any data you choose to the complete exit. The exit will get control only if the request is processed asynchronously.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the data to be passed to the complete exit.

**,REQECB=*reqecb***

Use this output parameter with either MODE=SYNCECB or MODE=ASYNCECB to specify the address of an ECB, which is to be posted when the request completes if the request was processed asynchronously.

Before coding REQECB, you must ensure that:

- You initialize the ECB before you issue the request.
- The ECB resides in either common storage or the home address space where IXLCONN was issued.
- Any tasks that wait for the ECB to be posted reside in the home address space where IXLCONN was issued.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that contains the address of the ECB to be posted when the request completes. The ECB must be aligned on a fullword boundary.

**,REQID=NO\_REQID****,REQID=*reqid***

Use this input parameter to specify a user-defined request identifier to be associated with the request. You can specify this request identifier on the IXLPURGE macro to cancel a request that has not yet been processed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the user-defined request identifier.

**,REQTOKEN=*reqtoken***

Use this output parameter with either MODE=SYNCTOKEN or MODE=ASYNCTOKEN to specify the address of a storage area to receive the request token that is returned when the request will be processed asynchronously. This token, which uniquely identifies the request, must be used as input to the IXLFComp macro, which you use to determine if the request has completed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field where the system will put the request token.

**,RETCODE=*retcode***

Use this output parameter to specify a field to contain the return code. (The return code is also returned in register 15.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the return code when the request has completed.

**,RSNCODE=*rsncode***

Use this output parameter to specify a field to contain the reason code returned, if applicable. (The reason code is also returned in register 0.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the reason code (if any) when the request has completed.

## ABEND Codes

Abend X'026' (See *z/OS MVS System Codes* for more information on this abend.)

## Return and Reason Codes

When the system returns control to your program:

- GPR 15 (and RETCODE, if specified) contains a return code.
- GPR 0 (and RSNCODE, if specified) contains a reason code, if applicable.

**Note:** The return and reason codes will also be put into the answer area (mapped by IXLYLAA), if there is one, when the request completes.

The IXLYCON macro provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

<b>0</b>	IXLRETCODEOK
<b>4</b>	IXLRETCODEWARNING
<b>8</b>	IXLRETCODEPARMERROR
<b>C</b>	IXLRETCODEENVERROR
<b>10</b>	IXLRETCODECOMPERROR

The following table contains hexadecimal return and reason codes, the equate symbols associated with each reason code, and the meaning and suggested action for each return and reason code.

Table 68. Return and Reason Codes for IXLLIST REQUEST=READ_EMCONTROLS Macro		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0	None.	<p><b>Meaning:</b> If you specified a MODE value of ASYNCECB, ASYNCEXIT, ASYNCTOKEN, or ASYNCNORESPONSE, the request has been successfully initiated. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed. For any other MODE value, the request has successfully completed.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=ASYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=ASYNCEXIT, the connection's complete exit will be given control when the request has completed.</li> <li>• If you specified MODE=ASYNCTOKEN, use REQTOKEN when calling IXLFCOMP to determine when the request has completed.</li> </ul>

Table 68. Return and Reason Codes for IXLLIST REQUEST=READ\_EMCONTROLS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	xxxx0402	<p><b>Equate Symbol:</b> IXLRNCODEASYNCH</p> <p><b>Meaning:</b> The request could not be processed synchronously. It will be processed asynchronously. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed.</p> <ul style="list-style-type: none"> <li>• If MODE=SYNCECB was specified, the ECB specified for REQECB will be posted when the request has finished.</li> <li>• If MODE=SYNCEXIT was specified, the connection's complete exit will be given control when the request has finished.</li> <li>• If MODE=SYNCTOKEN was specified, an asynchronous request token was returned in the area specified by REQTOKEN. This request token may be specified on the IXLFCOMP macro to determine when the request has finished.</li> </ul> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=SYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=SYNCEXIT, the connection's complete exit will be given control when the request has completed.</li> <li>• If you specified MODE=SYNCTOKEN, use REQTOKEN when calling IXLFCOMP to determine when the request has completed.</li> </ul>
8	xxxx0801	<p><b>Equate Symbol:</b> IXLRNCODEBADPARMLIST</p> <p><b>Meaning:</b> Program error. The parameter list for this request is not addressable. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Verify that:</p> <ul style="list-style-type: none"> <li>• The parameter list address is uncorrupted.</li> <li>• The parameter list is addressable in the caller's primary address space.</li> <li>• If you are issuing this macro while disabled, the parameter list resides in either page-fixed or DREF storage.</li> <li>• If you are issuing this macro in AR-mode and you specified the parameter list address using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If you are issuing this macro in AR-mode, SYSSTATE ASCENV=AR must be issued before issuing this macro.</li> </ul>

Table 68. Return and Reason Codes for IXLLIST REQUEST=READ\_EMCONTROLS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0804	<p><b>Equate Symbol:</b> IXLRSNCODEBADVERSIONNUM (or IXLRSNCODEBADVERSION#)</p> <p><b>Meaning:</b> The version number in the macro parameter list is not compatible with the level of XES currently being used. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that your program did not overlay the parameter list storage.</li> <li>• Verify that your program was assembled with the correct macro library for the release of MVS on which your program is running.</li> </ul>
8	xxxx080A	<p><b>Equate Symbol:</b> IXLRSNCODEBADCONTOKEN</p> <p><b>Meaning:</b> Program error. The specified connect token (CONTOKEN) was not valid for one of the following reasons:</p> <ol style="list-style-type: none"> <li>1. The user with the connection identifier represented by the token has disconnected from the structure.</li> <li>2. The connector's task (the task that issued IXLCONN) ended.</li> <li>3. The specified token is not the token that was returned from IXLCONN.</li> <li>4. The request was issued from an address space other than the address space in which IXLCONN was issued.</li> <li>5. The connect token was invalidated during rebuild.</li> <li>6. The connect token was invalidated by XES.</li> </ol> <p><b>Note:</b> The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Take the action with the corresponding meaning.</p> <ol style="list-style-type: none"> <li>1. Discontinue use of the structure. Perform recovery and cleanup for the structure and check the protocol for the use of the structure to determine why you did not know of the disconnection.</li> <li>2. Discontinue use of the structure. Perform recovery and cleanup for the structure and check the protocol for the use of the structure to determine why you did not know that the task ended.</li> <li>3. The contoken is returned in the CONACONTOKEN field of the answer area specified on the IXLCONN request.</li> <li>4. Issue your request from the same address space the IXLCONN was issued in.</li> <li>5. Wait for the rebuild to complete, and try again.</li> <li>6. Discontinue use of the structure. Perform recovery and cleanup for the structure.</li> </ol>

Table 68. Return and Reason Codes for IXLLIST REQUEST=READ\_EMCONTROLS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0824	<p><b>Equate Symbol:</b> IXLRSNCODEWRONGSTRTYPE</p> <p><b>Meaning:</b> Program error. The connection specified by CONTOKEN is not to a list structure.</p> <p><b>Action:</b> Specify the appropriate CONTOKEN value, and rerun your program. The CONTOKEN is returned in the answer area provided on the IXLCONN invocation. The type of structure built is passed on the TYPE parameter of the IXLCONN macro.</p>
8	xxxx0825	<p><b>Equate Symbol:</b> IXLRSNCODENOENTRY</p> <p><b>Meaning:</b> Program error. The designated event monitor controls object (EMC) does not exist for the user of the designated sublist.</p> <p><b>Action:</b> None necessary. However, if this return code and reason code are unexpected, you should examine the parameters specified for the list number and entry key on the invocation of this macro.</p>
8	xxxx0838	<p><b>Equate Symbol:</b> IXLRSNCODEBADANSAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by ANSAREA is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The answer area address specified by ANSAREA is valid.</li> <li>• If the caller is running in AR-mode and the ANSAREA parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• The address specified for ANSAREA is addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are calling IXLLIST while disabled, ANSAREA must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLLIST in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLIST macro.</li> </ul>
8	xxxx0839	<p><b>Equate Symbol:</b> IXLRSNCODEBADREQTOKENAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by REQTOKEN is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The request token area specified by REQTOKEN is valid.</li> <li>• If the caller is running in AR-mode and the REQTOKEN parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If you are calling IXLLIST while disabled, REQTOKEN must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLLIST in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLIST macro.</li> </ul>

Table 68. Return and Reason Codes for IXLLIST REQUEST=READ\_EMCONTROLS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx083D	<p><b>Equate Symbol:</b> IXLRSNCODEBADANSLEN</p> <p><b>Meaning:</b> Program error. The length of the answer area, as specified by ANSLEN, is not sufficient to contain answer area information. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Increase the size of the answer area, and rerun your program. The minimum size of the ANSAREA is in the LAA_LEN field of the IXLYLAA mapping macro.</p>
8	xxxx0847	<p><b>Equate Symbol:</b> IXLRSNCODEBADLISTNUMBER</p> <p><b>Meaning:</b> Program error. The specified LISTNUM value exceeds the number of lists for the structure.</p> <p><b>Action:</b> Correct LISTNUM to specify a list number that exists in the structure. The number of lists allocated for a structure is determined on the LISTHEADERS parameter of the IXLCONN macro. The list numbers go from 0 to n-1, where n is the number of lists specified.</p>
8	xxxx084A	<p><b>Equate Symbol:</b> IXLRSNCODENOKEYS</p> <p><b>Meaning:</b> Program error. The structure does not support the use of entry keys. The IXLLIST request type either required the structure to support entry keys or designated a list entry or list position by list number and entry key.</p> <p><b>Action:</b> Ensure that you are connected to the intended structure. The use of keys for a structure is determined by the REFOPTION keyword on the IXLCONN macro.</p>
8	xxxx0851	<p><b>Equate Symbol:</b> IXLRSNCODENOSUSPENDISABLE</p> <p><b>Meaning:</b> Program error. The request failed because MODE=SYNCSUSPEND was specified, but the caller is disabled and cannot be suspended.</p> <p><b>Action:</b> Either specify another MODE value, or become enabled (release the CPU lock), then reissue the request.</p>
C	xxxx0C06	<p><b>Equate Symbol:</b> IXLRSNCODENOCONN</p> <p><b>Meaning:</b> Environmental error. This system does not have connectivity to the coupling facility that contains the list structure. Possible reasons for this are:</p> <ul style="list-style-type: none"> <li>• The operator issued VARY PATH,OFFLINE.</li> <li>• The operator issued CONFIG CHP,OFFLINE.</li> <li>• Hardware errors to the coupling facility.</li> <li>• Facility or path failure to the coupling facility.</li> </ul> <p><b>Action:</b> Begin rebuilding the structure on a different coupling facility, or disconnect from the structure.</p>

Table 68. Return and Reason Codes for IXLLIST REQUEST=READ\_EMCONTROLS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C13	<p><b>Equate Symbol:</b> IXLRSNCODEREQPURGED</p> <p><b>Meaning:</b> Environmental error. The request was purged prior to completion of the request. Possible reasons include:</p> <ul style="list-style-type: none"> <li>• The connector failed.</li> <li>• The connector disconnected.</li> <li>• The requestor failed.</li> <li>• The request was purged by IXLPURGE.</li> <li>• Requests were purged when the connector provided an IXLEERSP response for the Rebuild Stop or Rebuild Cleanup event.</li> <li>• The secondary address space was no longer valid.</li> </ul> <p><b>Action:</b> None if this is expected. Otherwise, determine why the connector failed.</p>
C	xxxx0C14	<p><b>Equate Symbol:</b> IXLRSNCODESTATUSUNKNOWN</p> <p><b>Meaning:</b> Environmental error. The request has completed, but the final disposition of the request cannot be determined. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the validity of your data by comparing the expected results with what is in the coupling facility.</li> <li>• Consider running rebuild processing to move the structure to attempt to protect it from further errors of this kind.</li> </ul>
C	xxxx0C25	<p><b>Equate Symbol:</b> IXLRSNCODESTRFAILURE</p> <p><b>Meaning:</b> Environmental error. The list structure failed prior to completion of the request.</p> <p><b>Action:</b> Either rebuild or disconnect from the structure.</p>
C	xxxx0C68	<p><b>Equate Symbol:</b> IXLRSNCODEBADREQCFLEVEL</p> <p><b>Meaning:</b> Environmental error. The request type is not permitted for the level of coupling facility in which the target structure is allocated.</p> <p><b>Action:</b> Either disconnect from the structure (using IXLDISC) or initiate a rebuild of the structure, if allowed (using IXLREBLD) in a coupling facility of the correct CFLEVEL.</p>
C	xxxx0CA0	<p><b>Equate Symbol:</b> IXLRSNCODEQUIESCEDSUSPENDFAIL</p> <p><b>Meaning:</b> Environmental error. The request is failed because the structure is quiesced for a system-managed process and SUSPEND=FAIL is specified on the IXLCONN.</p> <p><b>Action:</b> None, if this is expected.</p>

Table 68. Return and Reason Codes for IXLLIST REQUEST=READ\_EMCONTROLS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxxFFFF	<b>Equate Symbol:</b> IXLRSNCODENOTAVAILABLE <b>Meaning:</b> Environmental error. XES functions are not available. This could occur because the coupling facility hardware necessary to provide XES functions is not present. <b>Action:</b> Re-IPL the system, or follow your particular failure management protocol.
10	xxxx10xx	<b>Meaning:</b> System error. XES processing failure. The state of the involved structure and the disposition of the request are unpredictable. The answer area (ANSAREA) fields do not contain valid information. <b>Action:</b> Contact the IBM support center.



## Chapter 69. IXLLIST REQUEST=READ\_EQCONTROLS

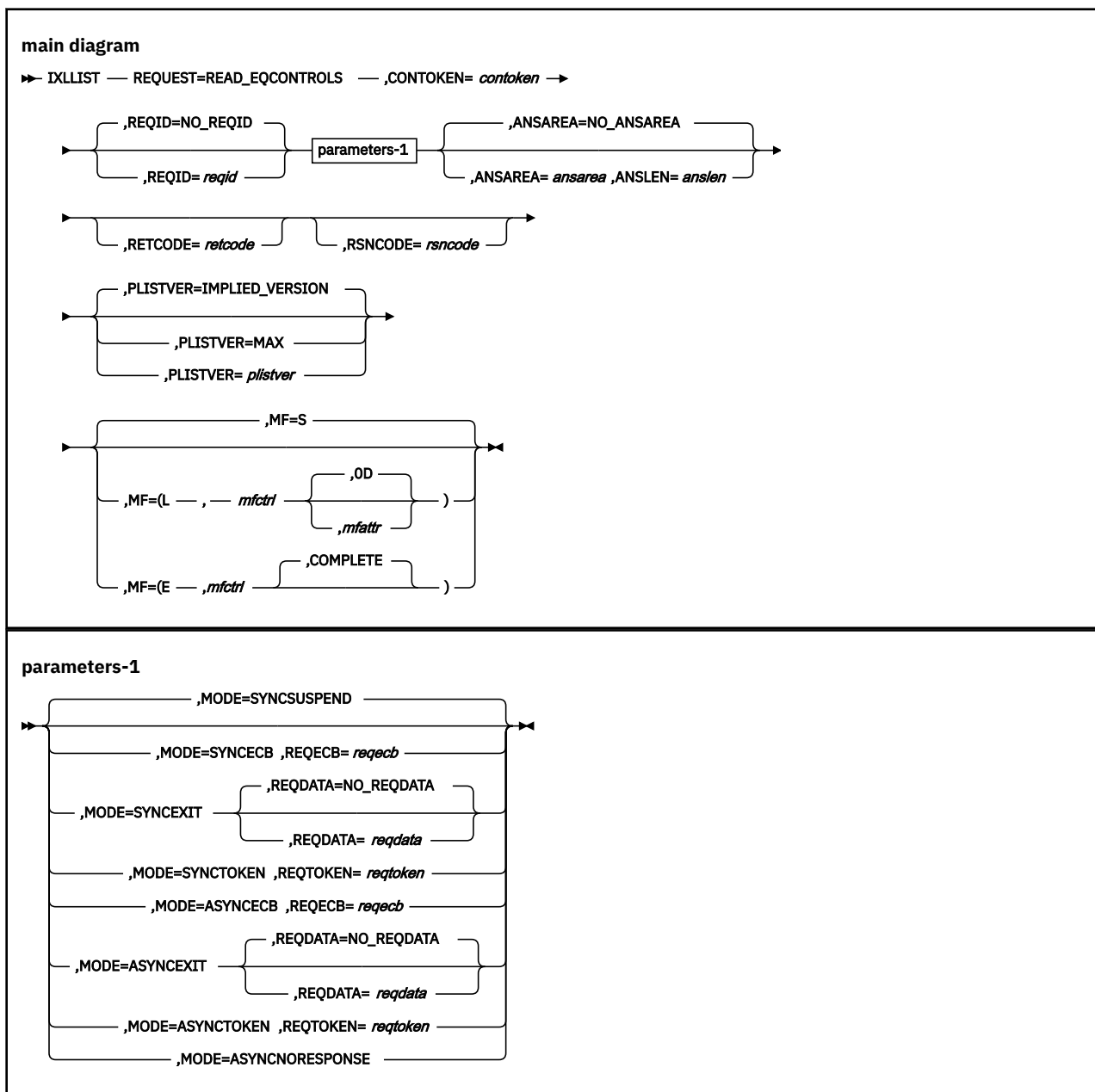
### Description

A READ\_EQCONTROLS request allows you to retrieve control information about your event queue. The information is returned in the answer area (ANSAREA).

You can issue a READ\_EQCONTROLS request only for keyed list structures allocated in a coupling facility of CFLEVEL=3 or higher. READ\_EQCONTROLS requests issued for a list structure allocated in a coupling facility of CFLEVEL=0, 1, or 2 will fail.

### Syntax Diagram

The syntax diagram for IXLLIST REQUEST=READ\_EQCONTROLS is as follows:



**Note:** If MODE=SYNCTOKEN or MODE=ASYNCTOKEN is specified, ANSAREA=*ansarea* and ANSLen=*anslen* are required.

## Parameter Descriptions

The parameter descriptions for REQUEST=READ\_EQCONTROLS are listed in alphabetical order. Default values are underlined:

### **REQUEST=READ\_EQCONTROLS**

Use this input parameter to specify that event queue control information associated with your event queue is to be retrieved.

#### **,ANSAREA=NO ANSAREA**

#### **,ANSAREA=*ansarea***

Use this output parameter to specify an answer area to contain information returned from the request. The format of the answer area is described by mapping macro IXLYLAA. Upon successful completion of a request for event queue control information, the answer area contains:

- An indicator as to whether the system is to drive the connector's list transition exit when this event queue changes from empty to nonempty (field LAAREQC\_DRIVEEXIT)
- An indicator as to whether the user is currently monitoring the event queue (field LAAREQC\_MONITORINGACTIVE)
- The vector index associated with the monitored event queue (field LAAREQC\_VECTORINDEX)
- The number of event monitor controls (EMCs) queued to the event queue (field LAAREQC\_EMCCQUEUEDCNT)
- The approximate number of empty to nonempty event queue transitions that have occurred (field LAAREQC\_EVENTTRAN).

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an area (with a length of ANSLen) to contain the information.

#### **,ANSLen=*anslen***

Use this input parameter to specify the size of the storage area specified by ANSAREA.

Check the prologue of the IXLYLAA mapping macro for the minimum required size of the answer area, or use the length field (LAA\_LEN) in the LAA.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 2-byte field that contains the size, in bytes, of the answer area.

#### **,CONTOKEN=*contoken***

Use this input parameter to specify the connect token that was returned by the IXLCONN service. The connect token uniquely identifies your connection to the list structure, and must be specified on each IXLLIST invocation.

The connect token is available in the IXLCONN answer area that is mapped by IXLYCONA.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 16-byte field that contains the connect token.

#### **,MF=S**

#### **,MF=(L,*mfctrl*)**

#### **,MF=(L,*mfctrl*,*mfattr*)**

#### **,MF=(L,*mfctrl*,0D)**

#### **,MF=(M,*mfctrl*)**

#### **,MF=(M,*mfctrl*,COMPLETE)**

#### **,MF=(M,*mfctrl*,NOCHECK)**

#### **,MF=(E,*mfctrl*)**

#### **,MF=(E,*mfctrl*,COMPLETE)**

#### **,MF=(E,*mfctrl*,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

#### ***,mfctrl***

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

#### ***,mfattr***

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

#### ***,COMPLETE***

#### ***,NOCHECK***

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

#### **COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if SMILE=*var* were an optional parameter and the default is SMILE=NO\_SMILE then it would not be documented. However, if the default was SMILE=:), then it would be documented because a value would be the default.

#### **NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

#### ***,MODE=SYNCSUSPEND***

#### ***,MODE=SYNCECB***

#### ***,MODE=SYNCEXIT***

#### ***,MODE=SYNCTOKEN***

#### ***,MODE=ASYNCECB***

#### ***,MODE=ASYNCEXIT***

#### ***,MODE=ASYNCTOKEN***

#### ***,MODE=ASYNCRESPONSE***

Use this input parameter to specify:

- Whether the request is to be performed synchronously or asynchronously
- How you want to be notified of request completion

#### **MODE=SYNCSUSPEND**

The request is suspended until it can complete synchronously. To use this option, your program must be enabled for I/O and external interrupts.

#### **MODE=SYNCECB**

The request processes synchronously if possible. If the request processes asynchronously, the ECB specified by REQECB is posted when the request completes.

**MODE=SYNCEXIT**

The request processes synchronously if possible. If the request processes asynchronously, your complete exit is given control when the request completes.

**MODE=SYNCTOKEN**

The request processes synchronously if possible. If the request processes asynchronously, an asynchronous request token is returned in the area specified by REQTOKEN on invocation of the request. Use the returned request token on the IXLFCOMP macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=SYNCTOKEN is specified.

**MODE=ASYNCECB**

The request processes asynchronously. The ECB specified by REQECB is posted when the request completes.

**MODE=ASYNCEXIT**

The request processes asynchronously. Your complete exit is given control when the request completes.

**MODE=ASYNCTOKEN**

The request processes asynchronously. An asynchronous request token is returned in the area specified by REQTOKEN upon invocation of the request. Use the returned request token on the IXLFCOMP macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=ASYNCTOKEN is specified.

**MODE=ASYNCSNORESPONSE**

The request processes asynchronously. No notification of request completion is provided. The answer area (ANSAREA) fields will not contain valid information when this mode is specified.

**Note:** You cannot code MODE=ASYNCSNORESPONSE with LOCKINDEX, BUFFER, or BUFLIST.

**,PLISTVER=IMPLIED\_VERSION****,PLISTVER=MAX****,PLISTVER=plistver**

Use this input parameter to specify the version of the macro. See [“Understanding IXLLIST Version Support”](#) on page 965 for a description of the options available with PLISTVER.

**,REQDATA=NO\_REQDATA****,REQDATA=reqdata**

Use this input parameter with MODE=SYNCEXIT or MODE=ASYNCEXIT to pass any data you choose to the complete exit. The exit will get control only if the request is processed asynchronously.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the data to be passed to the complete exit.

**,REQECB=reqecb**

Use this output parameter with either MODE=SYNCECB or MODE=ASYNCECB to specify the address of an ECB, which is to be posted when the request completes if the request was processed asynchronously.

Before coding REQECB, you must ensure that:

- You initialize the ECB before you issue the request.
- The ECB resides in either common storage or the home address space where IXLCONN was issued.
- Any tasks that wait for the ECB to be posted reside in the home address space where IXLCONN was issued.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that contains the address of the ECB to be posted when the request completes. The ECB must be aligned on a fullword boundary.

**,REQID=NO\_REQID****,REQID=reqid**

Use this input parameter to specify a user-defined request identifier to be associated with the request. You can specify this request identifier on the IXLPURGE macro to cancel a request that has not yet been processed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the user-defined request identifier.

**,REQTOKEN=reqtoken**

Use this output parameter with either MODE=SYNCTOKEN or MODE=ASYNCTOKEN to specify the address of a storage area to receive the request token that is returned when the request will be processed asynchronously. This token, which uniquely identifies the request, must be used as input to the IXLFCOMP macro, which you use to determine if the request has completed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field where the system will put the request token.

**,RETCODE=retcode**

Use this output parameter to specify a field to contain the return code. (The return code is also returned in register 15.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the return code when the request has completed.

**,RSNCODE=rsncode**

Use this output parameter to specify a field to contain the reason code returned, if applicable. (The reason code is also returned in register 0.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the reason code (if any) when the request has completed.

## ABEND Codes

---

Abend X'026' (See [z/OS MVS System Codes](#) for more information on this abend.)

## Return and Reason Codes

---

When the system returns control to your program:

- GPR 15 (and RETCODE, if specified) contains a return code.
- GPR 0 (and RSNCODE, if specified) contains a reason code, if applicable.

**Note:** The return and reason codes will also be put into the answer area (mapped by IXLYLAA), if there is one, when the request completes.

The IXLYCON macro provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

**0**

IXLRETCODEOK

**4**

IXLRETCODEWARNING

**8**

IXLRETCODEPARMERROR

**C**

IXLRETCODEENVERROR

**10**

IXLRETCODECOMPERROR

The following table contains hexadecimal return and reason codes, the equate symbols associated with each reason code, and the meaning and suggested action for each return and reason code.

Table 69. Return and Reason Codes for IXLLIST REEQUEST=READ\_EQCONTROLS Macro

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0	None.	<p><b>Meaning:</b> If you specified a MODE value of ASYNCECB, ASYNCEXIT, ASYNCTOKEN, or ASYNCNORESPONSE, the request has been successfully initiated. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed. For any other MODE value, the request has successfully completed.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=ASYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=ASYNCEXIT, the connection's complete exit will be given control when the request has completed.</li> <li>• If you specified MODE=ASYNCTOKEN, use REQTOKEN when calling IXLFCOMP to determine when the request has completed.</li> </ul>
4	xxxx0402	<p><b>Equate Symbol:</b> IXLRSNCODEASYNCH</p> <p><b>Meaning:</b> The request could not be processed synchronously. It will be processed asynchronously. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed.</p> <ul style="list-style-type: none"> <li>• If MODE=SYNCECB was specified, the ECB specified for REQECB will be posted when the request has finished.</li> <li>• If MODE=SYNCEXIT was specified, the connection's complete exit will be given control when the request has finished.</li> <li>• If MODE=SYNCTOKEN was specified, an asynchronous request token was returned in the area specified by REQTOKEN. This request token may be specified on the IXLFCOMP macro to determine when the request has finished.</li> </ul> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=SYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=SYNCEXIT, the connection's complete exit will be given control when the request has completed.</li> <li>• If you specified MODE=SYNCTOKEN, use REQTOKEN when calling IXLFCOMP to determine when the request has completed.</li> </ul>

Table 69. Return and Reason Codes for IXLLIST REEQUEST=READ\_EQCONTROLS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0801	<p><b>Equate Symbol:</b> IXLRSNCODEBADPARMLIST</p> <p><b>Meaning:</b> Program error. The parameter list for this request is not addressable. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Verify that:</p> <ul style="list-style-type: none"> <li>• The parameter list address is uncorrupted.</li> <li>• The parameter list is addressable in the caller's primary address space.</li> <li>• If you are issuing this macro while disabled, the parameter list resides in either page-fixed or DREF storage.</li> <li>• If you are issuing this macro in AR-mode and you specified the parameter list address using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If you are issuing this macro in AR-mode, SYSSTATE ASCENV=AR must be issued before issuing this macro.</li> </ul>
8	xxxx0804	<p><b>Equate Symbol:</b> IXLRSNCODEBADVERSIONNUM (or IXLRSNCODEBADVERSION#)</p> <p><b>Meaning:</b> The version number in the macro parameter list is not compatible with the level of XES currently being used. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that your program did not overlay the parameter list storage.</li> <li>• Verify that your program was assembled with the correct macro library for the release of MVS on which your program is running.</li> </ul>

Table 69. Return and Reason Codes for IXLLIST REEQUEST=READ\_EQCONTROLS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx080A	<p><b>Equate Symbol:</b> IXLRNCODEBADCONTOKEN</p> <p><b>Meaning:</b> Program error. The specified connect token (CONTOKEN) was not valid for one of the following reasons:</p> <ol style="list-style-type: none"> <li>1. The user with the connection identifier represented by the token has disconnected from the structure.</li> <li>2. The connector's task (the task that issued IXLCONN) ended.</li> <li>3. The specified token is not the token that was returned from IXLCONN.</li> <li>4. The request was issued from an address space other than the address space in which IXLCONN was issued.</li> <li>5. The connect token was invalidated during rebuild.</li> <li>6. The connect token was invalidated by XES.</li> </ol> <p><b>Note:</b> The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Take the action with the corresponding meaning.</p> <ol style="list-style-type: none"> <li>1. Discontinue use of the structure. Perform recovery and cleanup for the structure and check the protocol for the use of the structure to determine why you did not know of the disconnection.</li> <li>2. Discontinue use of the structure. Perform recovery and cleanup for the structure and check the protocol for the use of the structure to determine why you did not know that the task ended.</li> <li>3. The contoken is returned in the CONACONTOKEN field of the answer area specified on the IXLCONN request.</li> <li>4. Issue your request from the same address space the IXLCONN was issued in.</li> <li>5. Wait for the rebuild to complete, and try again.</li> <li>6. Discontinue use of the structure. Perform recovery and cleanup for the structure.</li> </ol>
8	xxxx0824	<p><b>Equate Symbol:</b> IXLRNCODEWRONGSTRTYPE</p> <p><b>Meaning:</b> Program error. The connection specified by CONTOKEN is not to a list structure.</p> <p><b>Action:</b> Specify the appropriate CONTOKEN value, and rerun your program. The CONTOKEN is returned in the answer area provided on the IXLCONN invocation. The type of structure built is passed on the TYPE parameter of the IXLCONN macro.</p>



Table 69. Return and Reason Codes for IXLLIST REEQUEST=READ\_EQCONTROLS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0838	<p><b>Equate Symbol:</b> IXLRNCODEBADANSAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by ANSAREA is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The answer area address specified by ANSAREA is valid.</li> <li>• If the caller is running in AR-mode and the ANSAREA parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• The address specified for ANSAREA is addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are calling IXLLIST while disabled, ANSAREA must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLLIST in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLIST macro.</li> </ul>
8	xxxx0839	<p><b>Equate Symbol:</b> IXLRNCODEBADREQTOKENAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by REQTOKEN is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The request token area specified by REQTOKEN is valid.</li> <li>• If the caller is running in AR-mode and the REQTOKEN parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If you are calling IXLLIST while disabled, REQTOKEN must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLLIST in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLIST macro.</li> </ul>
8	xxxx083D	<p><b>Equate Symbol:</b> IXLRNCODEBADANSLEN</p> <p><b>Meaning:</b> Program error. The length of the answer area, as specified by ANSLEN, is not sufficient to contain answer area information. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Increase the size of the answer area, and rerun your program. The minimum size of the ANSAREA is in the LAA_LEN field of the IXLYLAA mapping macro.</p>

Table 69. Return and Reason Codes for IXLLIST REEQUEST=READ_EQCONTROLS Macro (continued)		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx084A	<p><b>Equate Symbol:</b> IXLRSCODENOKEYS</p> <p><b>Meaning:</b> Program error. The structure does not support the use of entry keys. The IXLLIST request type either required the structure to support entry keys or designated a list entry or list position by list number and entry key.</p> <p><b>Action:</b> Ensure that you are connected to the intended structure. The use of keys for a structure is determined by the REFOPTION keyword on the IXLCONN macro.</p>
8	xxxx0851	<p><b>Equate Symbol:</b> IXLRSCODENOSUSPENDISABLE</p> <p><b>Meaning:</b> Program error. The request failed because MODE=SYNCSUSPEND was specified, but the caller is disabled and cannot be suspended.</p> <p><b>Action:</b> Either specify another MODE value, or become enabled (release the CPU lock), then reissue the request.</p>
C	xxxx0C06	<p><b>Equate Symbol:</b> IXLRSCODENOCOONN</p> <p><b>Meaning:</b> Environmental error. This system does not have connectivity to the coupling facility that contains the list structure. Possible reasons for this are:</p> <ul style="list-style-type: none"> <li>• The operator issued VARY PATH,OFFLINE.</li> <li>• The operator issued CONFIG CHP,OFFLINE.</li> <li>• Hardware errors to the coupling facility.</li> <li>• Facility or path failure to the coupling facility.</li> </ul> <p><b>Action:</b> Begin rebuilding the structure on a different coupling facility, or disconnect from the structure.</p>
C	xxxx0C13	<p><b>Equate Symbol:</b> IXLRSCODEREQPURGED</p> <p><b>Meaning:</b> Environmental error. The request was purged prior to completion of the request. Possible reasons include:</p> <ul style="list-style-type: none"> <li>• The connector failed.</li> <li>• The connector disconnected.</li> <li>• The requestor failed.</li> <li>• The request was purged by IXLPURGE.</li> <li>• Requests were purged when the connector provided an IXLEERSP response for the Rebuild Stop or Rebuild Cleanup event.</li> <li>• The secondary address space was no longer valid.</li> </ul> <p><b>Action:</b> None if this is expected. Otherwise, determine why the connector failed.</p>

Table 69. Return and Reason Codes for IXLLIST REEQUEST=READ\_EQCONTROLS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C14	<b>Equate Symbol:</b> IXLRSNCODESTATUSUNKNOWN <b>Meaning:</b> Environmental error. The request has completed, but the final disposition of the request cannot be determined. The answer area (ANSAREA) fields do not contain valid information. <b>Action:</b> <ul style="list-style-type: none"> <li>• Verify the validity of your data by comparing the expected results with what is in the coupling facility.</li> <li>• Consider running rebuild processing to move the structure to attempt to protect it from further errors of this kind.</li> </ul>
C	xxxx0C25	<b>Equate Symbol:</b> IXLRSNCODESTRFAILURE <b>Meaning:</b> Environmental error. The list structure failed prior to completion of the request. <b>Action:</b> Either rebuild or disconnect from the structure.
C	xxxx0C68	<b>Equate Symbol:</b> IXLRSNCODEBADREQCFLEVEL <b>Meaning:</b> Environmental error. The request type is not permitted for the level of coupling facility in which the target structure is allocated. <b>Action:</b> Either disconnect from the structure (using IXLDISC) or initiate a rebuild of the structure, if allowed (using IXLREBLD) in a coupling facility of the correct CFLEVEL.
C	xxxx0CA0	<b>Equate Symbol:</b> IXLRSNCODEQUIESCEDSUSPENDFAIL <b>Meaning:</b> Environmental error. The request is failed because the structure is quiesced for a system-managed process and SUSPEND=FAIL is specified on the IXLCONN. <b>Action:</b> None, if this is expected.
C	xxxxFFFF	<b>Equate Symbol:</b> IXLRSNCODENOTAVAILABLE <b>Meaning:</b> Environmental error. XES functions are not available. This could occur because the coupling facility hardware necessary to provide XES functions is not present. <b>Action:</b> Re-IPL the system, or follow your particular failure management protocol.
10	xxxx10xx	<b>Meaning:</b> System error. XES processing failure. The state of the involved structure and the disposition of the request are unpredictable. The answer area (ANSAREA) fields do not contain valid information. <b>Action:</b> Contact the IBM support center.



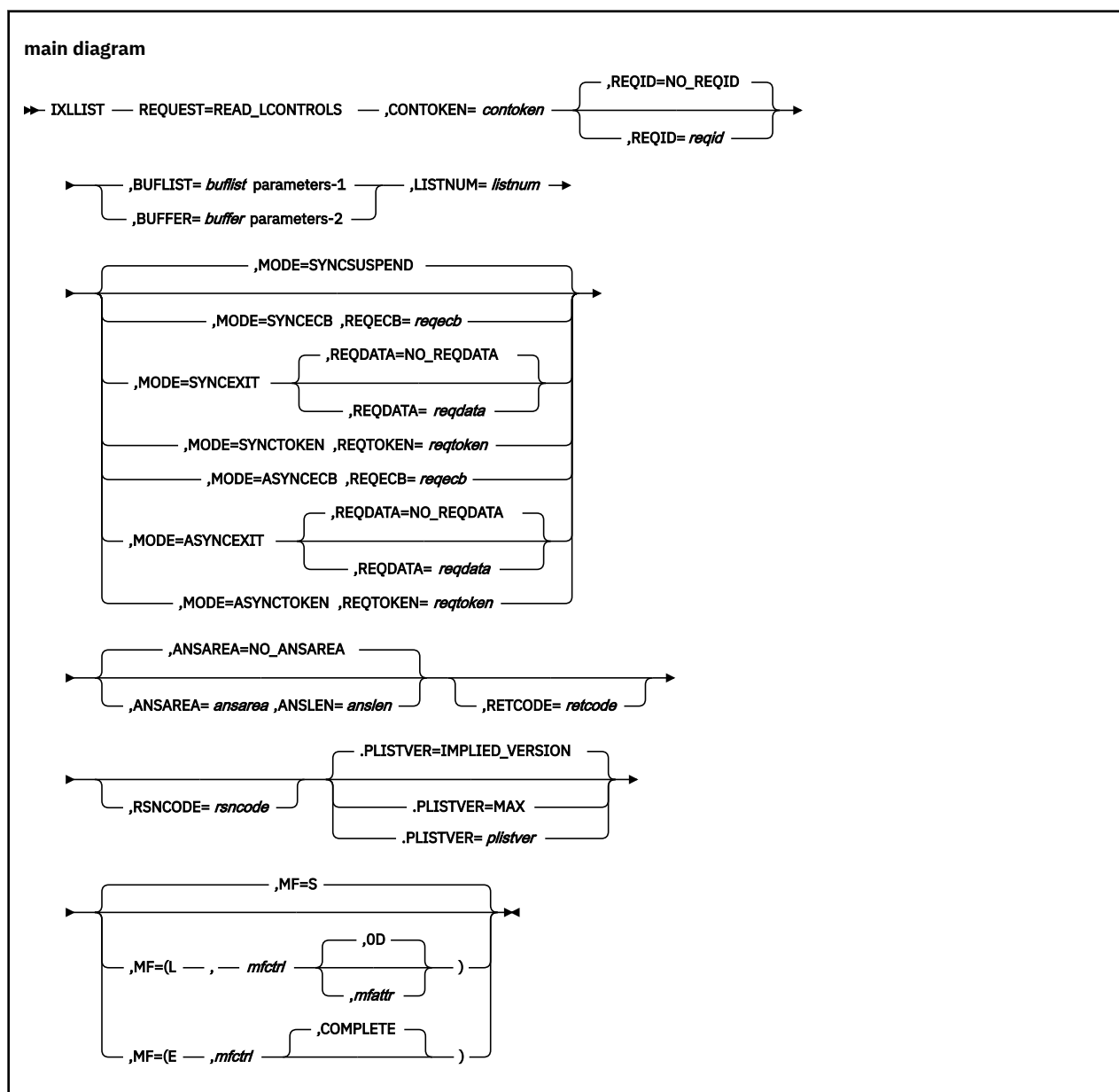
## Chapter 70. IXLLIST REQUEST=READ\_LCONTROLS

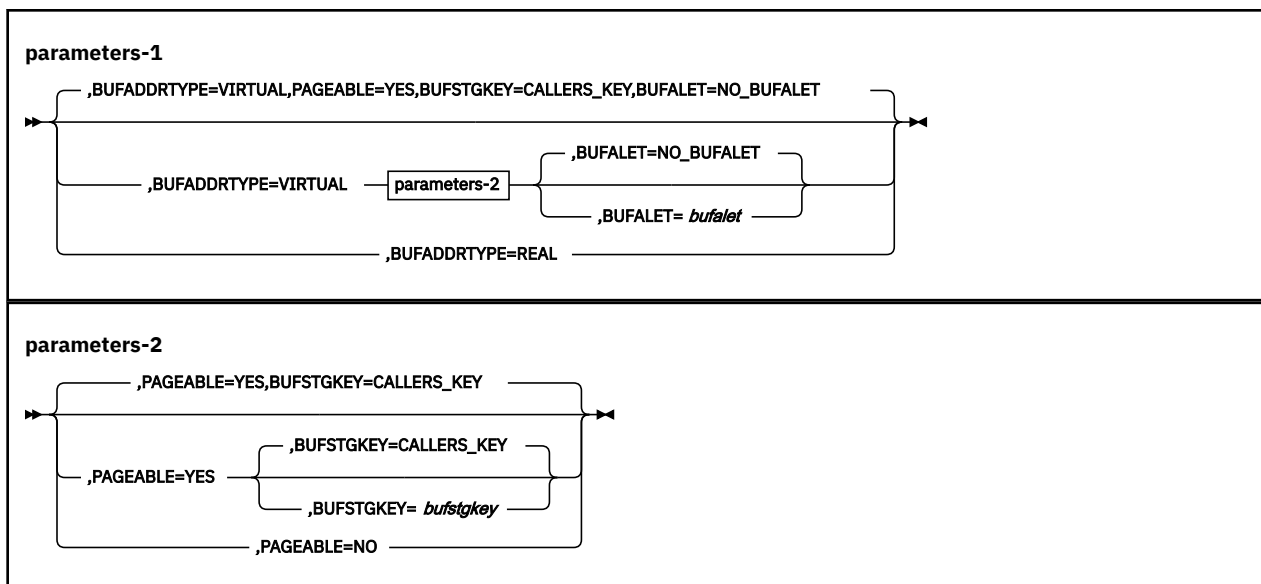
### Description

A READ\_LCONTROLS request allows you to retrieve list control and monitoring for a specified list (LISTNUM). The information is returned in the answer area (ANSAREA) and either a buffer (BUFFER) or list of buffers (BUFLIST). See the ANSAREA, BUFFER, or BUFLIST parameter descriptions for the specific information returned by this request.

### Syntax Diagram

The syntax diagram for IXLLIST REQUEST=READ\_LCONTROLS is as follows:



**Note:**

1. If MODE=SYNCTOKEN or MODE=ASYNCTOKEN is specified, ANSAREA=*ansarea* and ANSLen=*anslen* are required.
2. If MODE=ASYNCSNORESPONSE is specified, BUFLIST and BUFFER may not be specified.

## Parameter Descriptions

The parameter descriptions for REQUEST=READ\_LCONTROLS are listed in alphabetical order. Default values are underlined:

**REQUEST=READ\_LCONTROLS**

Use this input parameter to specify that control information for the list specified by LISTNUM be returned.

**,ANSAREA=NO\_ANSAREA****,ANSAREA=ansarea**

Use this output parameter to specify an answer area to contain list control information about the LISTNUM list that is returned from the request. The format of the answer area is described by mapping macro IXLYLAA.

When the request successfully completes, the following information is returned to the answer area:

- The total number of entries or elements currently on the list (field LAALISTCNT).
- The maximum number of entries or elements allowed to reside on the list (field LAALISTLIMIT).
- The approximate number of list transitions from the empty to the non-empty state (field LAALISTTRAN).
- The user-defined list description (field LAALISTDESC) and list authority (field LAALISTAUTH).
- The number of list monitoring information elements returned in the BUFFER area or BUFLIST buffers (field LAALMICNT).
- The list entry identifier of the entry to which the list cursor points (field LAALISTCURSOR).
- The current value of the list cursor direction indicator (field LAACURSORDIR). Returned only for a structure allocated in a CFLEVEL=1 or higher coupling facility.
- The list controls key value (field LAALISTKEY). Returned only for a structure allocated in a CFLEVEL=1 or higher coupling facility.
- The list controls maximum list key value (field LAAMAXLISTKEY). Returned only for a structure allocated in a CFLEVEL=1 or higher coupling facility.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an area (with a length of ANSLEN) where the information returned from the request will be put.

**,ANSLEN=*anslen***

Use this input parameter to specify the size of the storage area specified by ANSAREA.

Check the prologue of the IXLYLAA mapping macro for the minimum required size of the answer area, or use the length field (LAA\_LEN) in the LAA.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 2-byte field that contains the size, in bytes, of the answer area.

**,BUFADDRTYPE=VIRTUAL**

**,BUFADDRTYPE=REAL**

Use this input parameter to specify whether the buffer addresses specified in the BUFLIST list are virtual storage or real storage addresses.

**VIRTUAL**

The buffer addresses are virtual storage addresses. The virtual storage can be pageable or nonpageable. See the PAGEABLE parameter for information about managing storage binds when specifying virtual storage addresses.

**REAL**

The buffer addresses are real storage addresses.

It is the caller's responsibility to manage the binds between the data buffer virtual storage and the real storage addresses provided. The caller must ensure that the data buffer virtual storage remains bound to the real storage addresses provided until the request completes.

**,BUFALET=NO BUFALET**

**,BUFALET=*bufalet***

Use this input parameter to specify an access list entry token (ALET) to be used in referencing all of the buffers specified by BUFLIST.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 4-byte field that contains the ALET.

**,BUFFER=*buffer***

Use this output parameter to specify a buffer area to hold the returned list monitoring data for the LISTNUM list.

You must define the buffer to be 4096 bytes on a 4096-byte boundary.

Once the request completes, the buffer contains, starting at offset zero, an array of list monitoring information. The relative position of an element in the array associates that element with a connection identifier. The first array element is associated with a connection identifier of zero, and is reserved. The format of each array element is described by mapping macro IXLYLMI. The array elements are numbered from 0 to LAALMICNT-1.

**Note:** You cannot code BUFFER with MODE=ASYNCRNORESPONSE.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 4096-byte area to contain the data returned by the request.

**,BUFLIST=*buflist***

Use this input parameter to specify a buffer to hold the returned list monitoring information for the LISTNUM list.

BUFLIST specifies a 128-byte storage area that contains the address of a single buffer. (Unlike any other IXLLIST request, for a READ\_LCONTROLS request, only one buffer can be passed.) The address of the buffer should be placed in the second four bytes of the 128-byte area, beginning at offset zero. The rest of the area is ignored.

You must define the buffer to be 4096 bytes on a 4096-byte boundary.

Once the request completes, the buffer contains, starting at offset zero, an array of list monitoring information. The relative position of an element in the array associates that element with a connection

identifier. The first element is associated with a connection identifier of zero, and is reserved. The format of each array element is described by mapping macro IXLYLMI.

**Note:** You cannot code BUFLIST with MODE=ASYNCHRESPONSE.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 128-byte area that contains the address of the buffer.

**,BUFSTGKEY=bufstgkey**

Use this input parameter to specify a storage key that you define and use when referencing the buffer specified by BUFLIST or BUFFER.

If you do not specify BUFSTGKEY, all references to the buffer are assumed to be in your PSW key.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 1-byte field that contains the storage key in the format kkkkxxxx in which only the left four bits are used. (The right four bits are ignored.)

**,CONTOKEN=contoken**

Use this input parameter to specify the connect token that was returned by the IXLCONN service. The connect token uniquely identifies your connection to the list structure, and must be specified on each IXLLIST invocation.

The connect token is available in the IXLCONN answer area that is mapped by IXLYCONA.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 16-byte field that contains the connect token.

**,LISTNUM=listnum**

Use this input parameter to specify the number of the list for which list control and monitoring information will be returned.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 4-byte field that contains the number of the list.

**,MF=S**

**,MF=(L,mfctrl)**

**,MF=(L,mfctrl,mfattr)**

**,MF=(L,mfctrl,0D)**

**,MF=(M,mfctrl)**

**,MF=(M,mfctrl,COMPLETE)**

**,MF=(M,mfctrl,NOCHECK)**

**,MF=(E,mfctrl)**

**,MF=(E,mfctrl,COMPLETE)**

**,MF=(E,mfctrl,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

**,mfctrl**

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.



**,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

**,COMPLETE****,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

**COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if *SMILE=var* were an optional parameter and the default is *SMILE=NO\_SMILE* then it would not be documented. However, if the default was *SMILE=-*), then it would be documented because a value would be the default.

**NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

**,MODE=SYNCSUSPEND****,MODE=SYNCECB****,MODE=SYNCEXIT****,MODE=SYNCTOKEN****,MODE=ASYNCECB****,MODE=ASYNCEXIT****,MODE=ASYNCTOKEN**

Use this input parameter to specify:

- Whether the request is to be performed synchronously or asynchronously
- How you want to be notified of request completion if the request is processed asynchronously.

See *z/OS MVS Programming: Sysplex Services Guide* for more information on understanding synchronous and asynchronous cache operations.

**SYNCSUSPEND**

The request processes synchronously. If necessary, the request is suspended until it can complete synchronously. To use this option, your program must be enabled for I/O and external interrupts.

**SYNCECB**

The request processes synchronously if possible. If the request processes asynchronously, the ECB specified by REQECB is posted when the request completes.

**SYNCEXIT**

The request processes synchronously if possible. If the request processes asynchronously, your complete exit is given control when the request completes.

**SYNCTOKEN**

The request processes synchronously if possible. If the request processes asynchronously, an asynchronous request token is returned to the area specified by REQTOKEN. Use the returned request token on the IXLFComp macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=SYNCTOKEN is specified.

**ASYNCECB**

The request processes asynchronously. The ECB specified by REQECB is posted when the request completes.

**ASYNCEXIT**

The request processes asynchronously. Your complete exit is given control when the request completes.

**ASYNCTOKEN**

The request processes asynchronously. An asynchronous request token is returned to the area specified by REQTOKEN. Use the returned request token on the IXLFCOMP macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=ASYNCTOKEN is specified.

**,PAGEABLE=YES****,PAGEABLE=NO**

Use this input parameter to identify whether the storage areas specified by BUFFER or BUFLIST are in pageable or potentially pageable storage.

**YES**

Specify this option to indicate that the BUFFER or BUFLIST buffers reside in pageable virtual storage. XES performs the required page fixing to fix the buffers in real storage while the cache or list request transfers data to or from the coupling facility.

This includes storage obtained from pageable subpools, disabled reference (DREF) storage, and may include storage that has the potential to become pageable during the processing of a request. (An example is address space storage owned by any swappable address space, for which a PGSER FIX has been successfully processed, but for which the owning address space gets swapped during processing of a cache or list request.) This does not include implicitly non-pageable storage (for example, storage obtained from non-pageable subpools).

The system takes responsibility for managing binds to central storage for the duration of the cache or list request, regardless of what address space owns the storage or whether the storage-owning address space is swappable or nonswappable. The storage can be owned by any address space.

High shared virtual storage areas (above 2GB) may not be used.

**NO**

Specify this option to indicate that the BUFFER or BUFLIST buffers reside in non-pageable virtual storage. XES does not page fix the buffers in real storage.

This includes implicitly non-pageable storage areas (for example, storage obtained from non-pageable subpools), and may include storage that has the potential to become pageable during the processing of a request (An example is address space storage owned by any swappable address space, for which a PGSER FIX has been successfully processed, but for which the owning address space gets swapped-out during processing of a cache or list request.)

The system takes responsibility for managing binds to central storage for the duration of the cache or list request, if and only if the non-pageable storage is owned by either the requester's address space or the connector's address space. If the storage is owned by any other address space, then the invoker is responsible for ensuring that the virtual storage remains non-pageable for the duration of the request (including the case in which the storage is owned by a swappable address space that is swapped during processing of an IXLCACHE or IXLLIST request). Subject to this consideration, the storage can be owned by any address space. See [z/OS MVS Programming: Sysplex Services Guide](#).

**,PLISTVER=IMPLIED\_VERSION****,PLISTVER=MAX****,PLISTVER=plistver**

Use this input parameter to specify the version of the macro. See [“Understanding IXLLIST Version Support”](#) on page 965 for a description of the options available with PLISTVER.

**,REQDATA=NO\_REQDATA****,REQDATA=reqdata**

Use this input parameter with MODE=SYNCEXIT or MODE=ASYNCEXIT to pass any data you choose to the complete exit. The exit will get control only if the request is processed asynchronously.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the data to be passed to the complete exit.

**,REQECB=reqecb**

Use this output parameter with either MODE=SYNCECB or MODE=ASYNCECB to specify the address of an ECB, which is to be posted when the request completes if the request was processed asynchronously.

Before coding REQECB, you must ensure that:

- You initialize the ECB before you issue the request.
- The ECB resides in either common storage or the home address space where IXLCONN was issued.
- Any tasks that wait for the ECB to be posted reside in the home address space where IXLCONN was issued.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that contains the address of the ECB to be posted when the request completes. The ECB must be aligned on a fullword boundary.

**,REQID=NO\_REQID**

**,REQID=reqid**

Use this input parameter to specify a user-defined request identifier to be associated with the request. You can specify this request identifier on the IXPURGE macro to cancel a request that has not yet been processed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the user-defined request identifier.

**,REQTOKEN=reqtoken**

Use this output parameter with either MODE=SYNCTOKEN or MODE=ASYNCTOKEN to specify the address of a storage area to receive the request token that is returned when the request will be processed asynchronously. This token, which uniquely identifies the request, must be used as input to the IXLFCOMP macro, which you use to determine if the request has completed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field where the system will put the request token.

**,RETCODE=retcode**

Use this output parameter to specify a field to contain the return code. (The return code is also returned in register 15.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the return code when the request has completed.

**,RSNCODE=rsncode**

Use this output parameter to specify a field to contain the reason code returned, if applicable. (The reason code is also returned in register 0.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the reason code (if any) when the request has completed.

## ABEND Codes

---

Abend X'026' (see [z/OS MVS System Codes](#) for more information on this abend.)

## Return and Reason Codes

---

When the system returns control to your program:

- GPR 15 (and RETCODE, if specified) contains a return code.
- GPR 0 (and RSNCODE, if specified) contains a reason code, if applicable..

**Note:** The return and reason codes will also be put into the answer area (mapped by IXLYLAA), if there is one, when the request completes.

The IXLYCON macro provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

**0**

IXLRETCODEOK

**4**

IXLRETCODEWARNING

**8**

IXLRETCODEPARMERROR

**C**

IXLRETCODEENVERROR

**10**

IXLRETCODECOMPERROR

The following table contains hexadecimal return and reason codes, the equate symbols associated with each reason code, and the meaning and suggested action for each return and reason code.

Table 70. Return and Reason Codes for IXLLIST REQUEST=READ_LCONTROLS Macro		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0	None.	<p><b>Meaning:</b> If you specified a MODE value of ASYNCECB, ASYNCEXIT, ASYNCTOKEN, or ASYNCSNORESPONSE, the request has been successfully initiated. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed. For any other MODE value, the request has successfully completed.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=ASYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=ASYNCEXIT, the connection's complete exit will be given control when the request has completed.</li> <li>• If you specified MODE=ASYNCTOKEN, use REQTOKEN when calling IXLFCOMP to determine when the request has completed.</li> </ul>

Table 70. Return and Reason Codes for IXLLIST REQUEST=READ\_LCONTROLS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	xxxx0402	<p><b>Equate Symbol:</b> IXLRNCODEASYNCH</p> <p><b>Meaning:</b> The request could not be processed synchronously. It will be processed asynchronously. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed.</p> <ul style="list-style-type: none"> <li>• If MODE=SYNCECB was specified, the ECB specified for REQECB will be posted when the request has finished.</li> <li>• If MODE=SYNCEXIT was specified, the connection's complete exit will be given control when the request has finished.</li> <li>• If MODE=SYNCTOKEN was specified, an asynchronous request token was returned in the area specified by REQTOKEN. This request token may be specified on the IXLFCOMP macro to determine when the request has finished.</li> </ul> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=SYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=SYNCEXIT, the connection's complete exit will be given control when the request has completed.</li> <li>• If you specified MODE=SYNCTOKEN, use REQTOKEN when calling IXLFCOMP to determine when the request has completed.</li> </ul>
8	xxxx0801	<p><b>Equate Symbol:</b> IXLRNCODEBADPARMLIST</p> <p><b>Meaning:</b> Program error. The parameter list for this request is not addressable. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Verify that:</p> <ul style="list-style-type: none"> <li>• The parameter list address is uncorrupted.</li> <li>• The parameter list is addressable in the caller's primary address space.</li> <li>• If you are issuing this macro while disabled, the parameter list resides in either page-fixed or DREF storage.</li> <li>• If you are issuing this macro in AR-mode and you specified the parameter list address using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If you are issuing this macro in AR-mode, SYSSTATE ASCENV=AR must be issued before issuing this macro.</li> </ul>

Table 70. Return and Reason Codes for IXLLIST REQUEST=READ\_LCONTROLS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0804	<p><b>Equate Symbol:</b> IXLRSNCODEBADVERSIONNUM (or IXLRSNCODEBADVERSION#)</p> <p><b>Meaning:</b> The version number in the macro parameter list is not compatible with the level of XES currently being used. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that your program did not overlay the parameter list storage.</li> <li>• Verify that your program was assembled with the correct macro library for the release of MVS on which your program is running.</li> </ul>
8	xxxx080A	<p><b>Equate Symbol:</b> IXLRSNCODEBADCONTOKEN</p> <p><b>Meaning:</b> Program error. The specified connect token (CONTOKEN) was not valid for one of the following reasons:</p> <ol style="list-style-type: none"> <li>1. The user with the connection identifier represented by the token has disconnected from the structure.</li> <li>2. The connector's task (the task that issued IXLCONN) ended.</li> <li>3. The specified token is not the token that was returned from IXLCONN.</li> <li>4. The request was issued from an address space other than the address space in which IXLCONN was issued.</li> <li>5. The connect token was invalidated during rebuild.</li> <li>6. The connect token was invalidated by XES.</li> </ol> <p><b>Note:</b> The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Take the action with the corresponding meaning.</p> <ol style="list-style-type: none"> <li>1. Discontinue use of the structure. Perform recovery and cleanup for the structure and check the protocol for the use of the structure to determine why you did not know of the disconnection.</li> <li>2. Discontinue use of the structure. Perform recovery and cleanup for the structure and check the protocol for the use of the structure to determine why you did not know that the task ended.</li> <li>3. The contoken is returned in the CONACONTOKEN field of the answer area specified on the IXLCONN request.</li> <li>4. Issue your request from the same address space the IXLCONN was issued in.</li> <li>5. Wait for the rebuild to complete, and try again.</li> <li>6. Discontinue use of the structure. Perform recovery and cleanup for the structure.</li> </ol>

Table 70. Return and Reason Codes for IXLLIST REQUEST=READ\_LCONTROLS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0824	<p><b>Equate Symbol:</b> IXLRNCODEWRONGSTRTYPE</p> <p><b>Meaning:</b> Program error. The connection specified by CONTOKEN is not to a list structure.</p> <p><b>Action:</b> Specify the appropriate CONTOKEN value, and rerun your program. The CONTOKEN is returned in the answer area provided on the IXLCONN invocation. The type of structure built is passed on the TYPE parameter of the IXLCONN macro.</p>
8	xxxx0833	<p><b>Equate Symbol:</b> IXLRNCODEBADPGBLATTR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or the buffer in the BUFLIST list is specified as being pageable (PAGEABLE=YES), but is not.</p> <p><b>Action:</b> Change the buffer area to pageable storage or specify PAGEABLE=NO. See the PAGEABLE parameter description for specification instructions.</p>
8	xxxx0834	<p><b>Equate Symbol:</b> IXLRNCODEBADNONPGBLATTR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or the buffer in the BUFLIST list, is specified as being nonpageable (PAGEABLE=NO), but is either pageable or not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The type of storage in the buffer area corresponds to the value you specified on the PAGEABLE parameter. See the PAGEABLE parameter description for specification instructions.</li> <li>• The correct buffer address was used.</li> <li>• The buffer area was not previously freed.</li> <li>• If you are calling IXLLIST while disabled, the buffers must reside in either page-fixed or DREF storage.</li> <li>• The buffer areas were allocated in a storage key that matches the key specified by the BUFSTGKEY parameter, or, if BUFSTGKEY is omitted, the caller's PSW key.</li> <li>• If BUFLIST was specified and your program is running in AR-mode: <ul style="list-style-type: none"> <li>– If the BUFFER or BUFLIST parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>– The BUFALET specification is correct.</li> <li>– You specified SYSSTATE ASCENV=AR before issuing the IXLLIST macro.</li> </ul> </li> </ul>

Table 70. Return and Reason Codes for IXLLIST REQUEST=READ\_LCONTROLS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0835	<p><b>Equate Symbol:</b> IXLRNCODEBADDATAADDR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or the buffer in the BUFLIST list, is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The correct buffer address was used.</li> <li>• The buffer area was not previously freed.</li> <li>• The buffer area was allocated in a storage key that matches the key specified using the BUFSTGKEY parameter, or, if BUFSTGKEY is omitted, the caller's PSW key.</li> <li>• If BUFLIST was specified and your program is running in AR mode: <ul style="list-style-type: none"> <li>– The BUFALET specification is correct.</li> <li>– You specified SYSSTATE ASCENV=AR before issuing the macro.</li> </ul> </li> </ul>
8	xxxx0836	<p><b>Equate Symbol:</b> IXLRNCODEBADREALADDR</p> <p><b>Meaning:</b> Program error. A real storage address was provided in the BUFLIST list, but the buffer is not addressable in real storage.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that BUFADDRTYPE was specified as you intended.</li> <li>• Ensure that the buffer addresses specified by BUFLIST are valid.</li> </ul>
8	xxxx0838	<p><b>Equate Symbol:</b> IXLRNCODEBADANSAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by ANSAREA is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The answer area address specified by ANSAREA is valid.</li> <li>• If the caller is running in AR-mode and the ANSAREA parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• The address specified for ANSAREA is addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are calling IXLLIST while disabled, ANSAREA must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLLIST in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLIST macro.</li> </ul>



Table 70. Return and Reason Codes for IXLLIST REQUEST=READ\_LCONTROLS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0839	<p><b>Equate Symbol:</b> IXLRSNCODEBADREQTOKENAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by REQTOKEN is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The request token area specified by REQTOKEN is valid.</li> <li>• If the caller is running in AR-mode and the REQTOKEN parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If you are calling IXLLIST while disabled, REQTOKEN must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLLIST in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLIST macro.</li> </ul>
8	xxxx083D	<p><b>Equate Symbol:</b> IXLRSNCODEBADANSLEN</p> <p><b>Meaning:</b> Program error. The length of the answer area, as specified by ANSLEN, is not sufficient to contain answer area information. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Increase the size of the answer area, and rerun your program. The minimum size of the ANSAREA is in the LAA_LEN field of the IXLYLAA mapping macro.</p>
8	xxxx0847	<p><b>Equate Symbol:</b> IXLRSNCODEBADLISTNUMBER</p> <p><b>Meaning:</b> Program error. The specified LISTNUM value exceeds the number of lists for the structure.</p> <p><b>Action:</b> Correct LISTNUM to specify a list number that exists in the structure. The number of lists allocated for a structure is determined on the LISTHEADERS parameter of the IXLCONN macro. The list numbers go from 0 to n-1, where n is the number of lists specified.</p>
8	xxxx0851	<p><b>Equate Symbol:</b> IXLRSNCODENOSUSPENDISABLE</p> <p><b>Meaning:</b> Program error. The request failed because MODE=SYNCSUSPEND was specified, but the caller is disabled and cannot be suspended.</p> <p><b>Action:</b> Either specify another MODE value, or become enabled (release the CPU lock), then reissue the request.</p>
8	xxxx0865	<p><b>Equate Symbol:</b> IXLRSNCODEBADBUFSPEC</p> <p><b>Meaning:</b> Program error. There is an error in the buffer specification.</p> <p><b>Action:</b> Check the following:</p> <ul style="list-style-type: none"> <li>• The requirements for BUFLIST or BUFFER.</li> <li>• Buffer pointer(s) in the BUFLIST.</li> <li>• Buffer boundaries.</li> </ul>

Table 70. Return and Reason Codes for IXLLIST REQUEST=READ\_LCONTROLS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0866	<p><b>Equate Symbol:</b> IXLRSNCODEBADBUFKEY</p> <p><b>Meaning:</b> Program error. The buffer storage key specified by BUFSTGKEY is incorrect or BUFSTGKEY was not specified and the PSW key does not match the key of the buffers.</p> <p>The data cannot be stored in the specified buffer area.</p> <p><b>Action:</b> Check the following:</p> <ul style="list-style-type: none"> <li>• Determine if the key of the storage being used for the buffers is different from the PSW key.</li> <li>• If BUFSTGKEY was specified, verify that the key was put in the correct bits (see the explanation of this parameter for more information).</li> <li>• If you are calling IXLLIST in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLIST macro.</li> </ul>
8	xxxx0867	<p><b>Equate Symbol:</b> IXLRSNCODEBADBUFLIST</p> <p><b>Meaning:</b> Program error. The 128-byte storage area specified by BUFLIST is not addressable.</p> <p><b>Action:</b> Ensure that the address specified by BUFLIST is valid.</p>
C	xxxx0C06	<p><b>Equate Symbol:</b> IXLRSNCODENOCNN</p> <p><b>Meaning:</b> Environmental error. This system does not have connectivity to the coupling facility that contains the list structure. Possible reasons for this are:</p> <ul style="list-style-type: none"> <li>• The operator issued VARY PATH,OFFLINE.</li> <li>• The operator issued CONFIG CHP,OFFLINE.</li> <li>• Hardware errors to the coupling facility.</li> <li>• Facility or path failure to the coupling facility.</li> </ul> <p><b>Action:</b> Begin rebuilding the structure on a different coupling facility, or disconnect from the structure.</p>
C	xxxx0C13	<p><b>Equate Symbol:</b> IXLRSNCODEREQPURGED</p> <p><b>Meaning:</b> Environmental error. The request was purged prior to completion of the request. Possible reasons include:</p> <ul style="list-style-type: none"> <li>• The connector failed.</li> <li>• The connector disconnected.</li> <li>• The requestor failed.</li> <li>• The request was purged by IXLPURGE.</li> <li>• Requests were purged when the connector provided an IXLEERSP response for the Rebuild Stop or Rebuild Cleanup event.</li> <li>• The secondary address space was no longer valid.</li> </ul> <p><b>Action:</b> None if this is expected. Otherwise, determine why the connector failed.</p>

Table 70. Return and Reason Codes for IXLLIST REQUEST=READ\_LCONTROLS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C14	<b>Equate Symbol:</b> IXLRSNCODESTATUSUNKNOWN <b>Meaning:</b> Environmental error. The request has completed, but the final disposition of the request cannot be determined. The answer area (ANSAREA) fields do not contain valid information. <b>Action:</b> <ul style="list-style-type: none"> <li>• Verify the validity of your data by comparing the expected results with what is in the coupling facility.</li> <li>• Consider running rebuild processing to move the structure to attempt to protect it from further errors of this kind.</li> </ul>
C	xxxx0C25	<b>Equate Symbol:</b> IXLRSNCODESTRFAILURE <b>Meaning:</b> Environmental error. The list structure failed prior to completion of the request. <b>Action:</b> Either rebuild or disconnect from the structure.
C	xxxx0CA0	<b>Equate Symbol:</b> IXLRSNCODEQUIESCEDSUSPENDFAIL <b>Meaning:</b> Environmental error. The request is failed because the structure is quiesced for a system-managed process and SUSPEND=FAIL is specified on the IXLCONN. <b>Action:</b> None, if this is expected.
C	xxxxFFFF	<b>Equate Symbol:</b> IXLRSNCODENOTAVAILABLE <b>Meaning:</b> Environmental error. XES functions are not available. This could occur because the coupling facility hardware necessary to provide XES functions is not present. <b>Action:</b> Re-IPL the system, or follow your particular failure management protocol.
10	xxxx10xx	<b>Meaning:</b> System error. XES processing failure. The state of the involved structure and the disposition of the request are unpredictable. The answer area (ANSAREA) fields do not contain valid information. <b>Action:</b> Contact the IBM support center.



---

## Chapter 71. IXLLIST REQUEST=READ\_LIST

---

### Description

A READ\_LIST request allows you to read multiple entries from a single list into the storage areas specified by ANSAREA and/or BUFLIST or BUFFER and/or ADJAREA.

Entries are read in sequence from a starting list entry to the head or tail of the list on which the starting entry resides. The direction of processing (toward the head or toward the tail) depends on the direction you specify on the LISTDIR parameter. The starting entry must exist or the request fails.

For each processed entry, any combination of the following types of data can be read depending on what you specify on the TYPE parameter:

- List entry controls
- Entry data
- Adjunct data

Additionally, you can perform locking functions with a READ\_LIST request by specifying LOCKOPER. The lock designated by LOCKINDEX will be operated on as specified by LOCKOPER.

With a coupling facility of CFLEVEL=1 or higher, the following additional functions are supported for a READ\_LIST request:

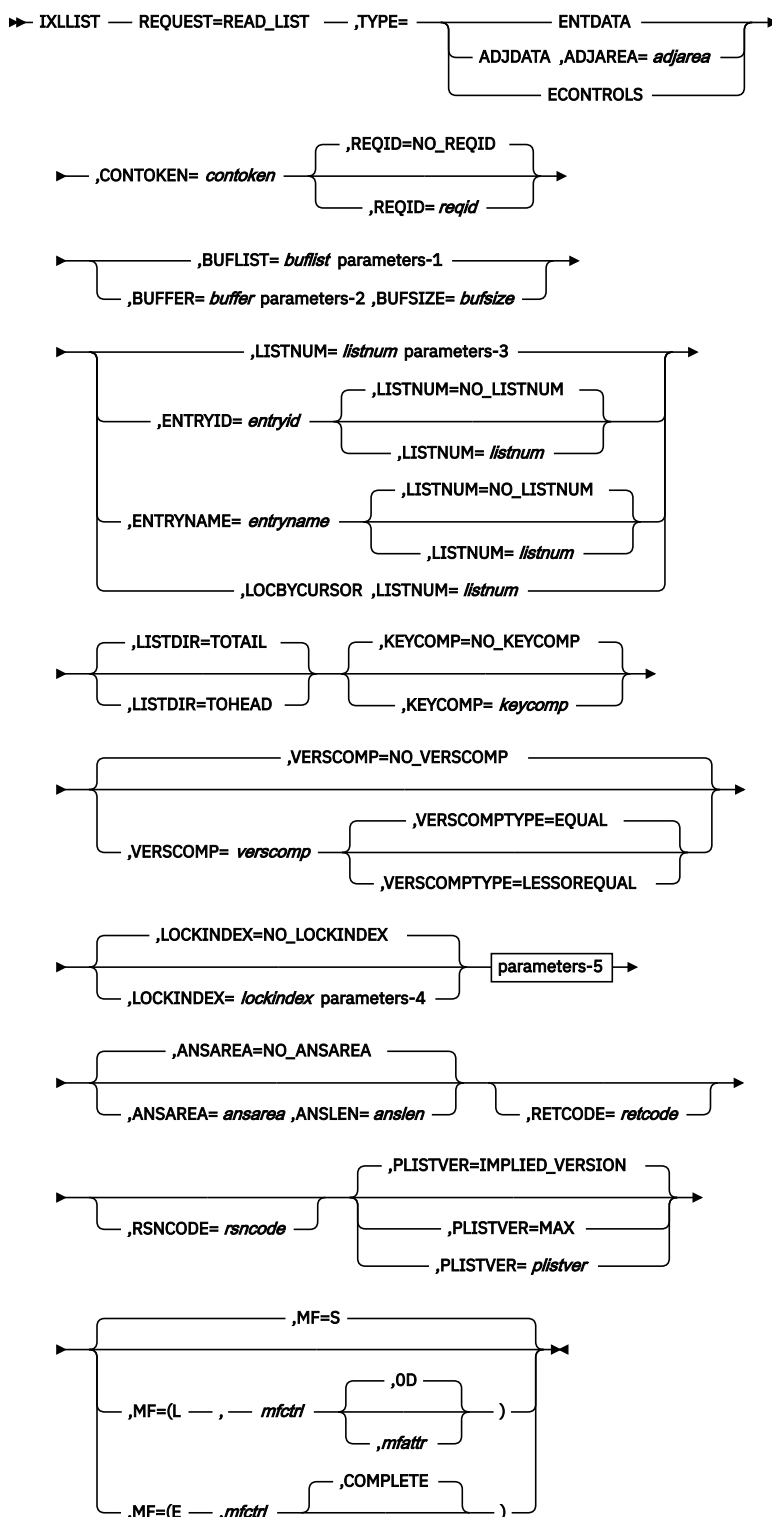
- You can specify that each list entry is to be read only after a successful comparison of the list authority value for the target list with a user-specified value. You specify the list authority comparison requirements using AUTHCOMP and AUTHCOMPTYPE. If the request is successful, you also can update the list authority value to a new value that you specify with NEWAUTH.
- You can specify that a list entry is to be read only after a successful comparison of the entry key value with a user-specified value (KEYCOMP). If the comparison is not successful, the current list entry is not read and processing continues with the next entry to be considered.
- Version number comparison also is enhanced to allow for an additional less-than-or-equal comparison operation.

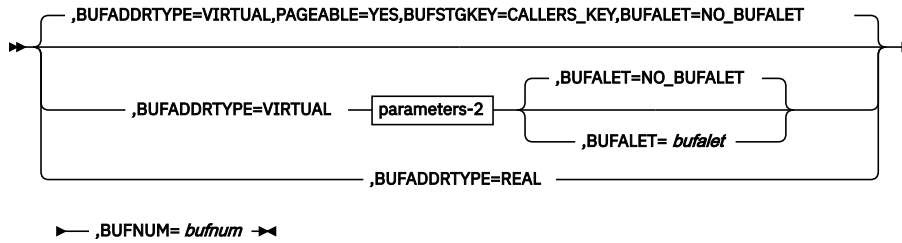
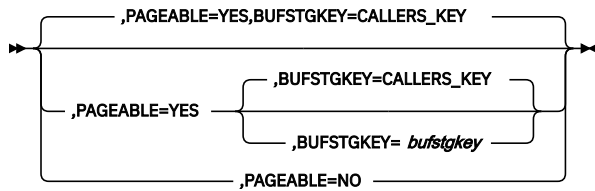
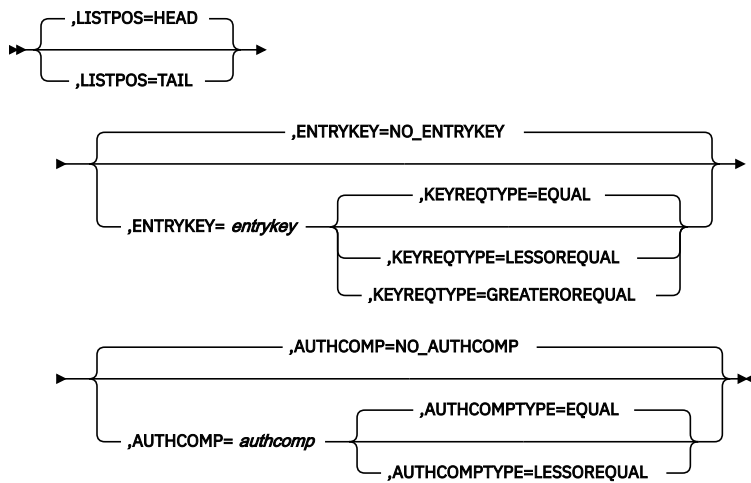
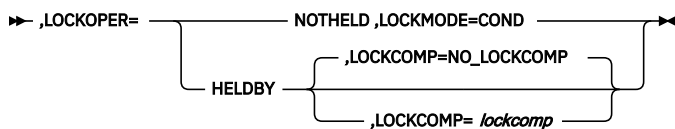
---

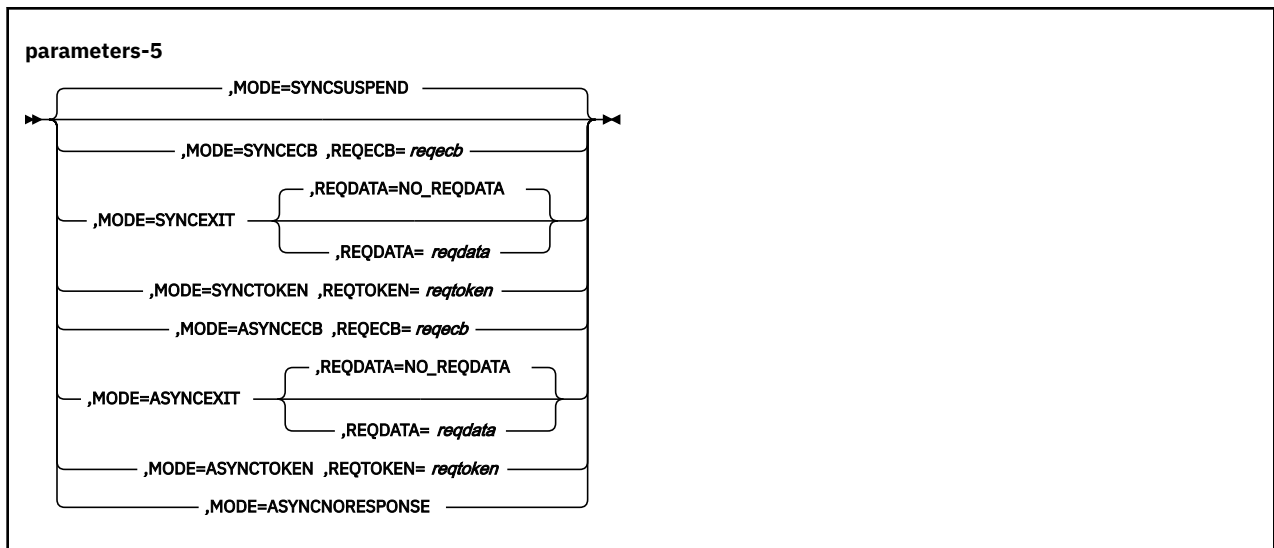
### Syntax Diagram

The syntax diagram for IXLLIST REQUEST=READ\_LIST is as follows:

## main diagram



**parameters-1****parameters-2****parameters-3****parameters-4**

**Note:**

1. If MODE=SYNCTOKEN or MODE=ASYNCTOKEN is specified, ANSAREA=*ansarea* and ANSLEN=*anslen* are required.
2. If MODE=ASYNCSUSPEND is specified, BUFFER, BUFLIST, and LOCKINDEX may not be specified.

## Parameter Descriptions

The parameter descriptions for REQUEST=READ\_LIST are listed in alphabetical order. Default values are underlined:

**REQUEST=READ\_LIST**

Use this input parameter to read multiple entries from a single list.

**,ADJAREA=*adjarea***

Use this output parameter to specify a storage area to contain the adjunct data that was read from the first processed entry. Adjunct data is returned only if ADJDATA is specified on the TYPE parameter.

Specify ADJAREA only for structures that support adjunct data. (Adjunct areas for a structure are established through the IXLCONN macro.)

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) to a 64-byte area (the adjunct area) to contain the adjunct data.

**,ANSAREA=NO\_ANSAREA****,ANSAREA=*ansarea***

Use this output parameter to specify an answer area to contain information returned from the request. The format of the answer area is described by mapping macro IXLYLAA.

On a successful completion of the request, the answer area contains:

- List entry controls (field LAARLRMLCTL mapped by IXLYLCTL) from the first processed list entry (if ECONTROLS is specified on the TYPE parameter).
- The number of processed entries (field LAAREADCNT).

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of an area (with a length of ANSLEN) where the information from the request will be put.

**,ANSLEN=*anslen***

Use this input parameter to specify the size of the storage area specified by ANSAREA.

Check the prologue of the IXLYLAA mapping macro for the minimum required size of the answer area, or use the length field (LAA\_LEN) in the LAA.



**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 2-byte field that contains the size, in bytes, of the answer area.

**,AUTHCOMP=NO\_AUTHCOMP**

**,AUTHCOMP=authcomp**

Use this input parameter to specify a value to be compared to the list authority value of the list specified by LISTNUM. You must supply the LISTNUM parameter.

If the comparison does not meet the condition specified by the AUTHCOMPTYPE parameter (EQUAL or LESSOREQUAL), the request fails.

**Note:** The AUTHCOMP parameter is valid only for list structures allocated in a coupling facility with CFLEVEL=1 or higher.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12 2 to ) of the 16-byte field that contains the list authority value.

**,AUTHCOMPTYPE=EQUAL**

**,AUTHCOMPTYPE=LESSOREQUAL**

Use this input parameter specify how the list audit comparison is to be performed.

**EQUAL**

The list authority for the list specified by LISTNUM must be equal to the value specified for AUTHCOMP.

**LESSOREQUAL**

The list authority for the list specified by LISTNUM must be less than or equal to the value specified for AUTHCOMP.

**Note:** The AUTHCOMPTYPE parameter is valid only for list structures allocated in a coupling facility with CFLEVEL=1 or higher.

**,BUFADDRTYPE=VIRTUAL**

**,BUFADDRTYPE=REAL**

Use this input parameter to specify whether the buffer addresses specified in the BUFLIST list are virtual storage or real storage addresses.

**VIRTUAL**

The buffer addresses are virtual storage addresses. The virtual storage can be pageable or nonpageable. See the PAGEABLE parameter for information about managing storage binds when specifying virtual storage addresses.

**REAL**

The buffer addresses are real storage addresses.

It is the caller's responsibility to manage the binds between the data buffer virtual storage and the real storage addresses provided. The caller must ensure that the data buffer virtual storage remains bound to the real storage addresses provided until the request completes.

**,BUFALET=NO\_BUFALET**

**,BUFALET=bufalet**

Use this input parameter to specify an access list entry token (ALET) to be used in referencing all of the buffers specified by BUFLIST.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 4-byte field that contains the ALET.

**,BUFFER=buffer**

Use this output parameter to specify a buffer area to hold data that is read from list entries.

You must ensure that the storage area specified by BUFFER:

- Is a multiple of 4096 bytes
- Is less than or equal to 65536 bytes
- Starts on a 4096-byte boundary

See the BUFSIZE parameter description for specifying the size of the buffer.

The type of data returned to the buffer depends on which types of data that you request on the TYPE parameter. See *z/OS MVS Programming: Sysplex Services Guide* for more detailed information about how data is returned from this request.

**Note:** You cannot code BUFFER with MODE=ASYNCRESPONSE.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of an area (with a length of BUFSIZE) where the data reads from list entries will be put.

#### **,BUFLIST=buflist**

Use this output parameter to specify a list of buffers to hold data that is read from list entries. BUFLIST specifies a 128-byte storage area that consists of a list of 0 to 16 buffer addresses.

##### **The 128-byte storage area must:**

- Consist of 0 to 16 elements.
- Each element must consist of an 8-byte field in which:
  - The left (high-order) 4 bytes are reserved.
  - The right (low-order) 4 bytes contain the address of a buffer.

##### **The BUFLIST buffers must:**

- Reside in the same address space or same data space.
- Be 4096 bytes.
- Start on a 4096-byte boundary.

**Note:** The buffers do not have to be contiguous in storage. List services treat BUFLIST buffers as a single buffer, even if the buffers are not contiguous.

See the BUFNUM parameter description to specify the number of buffers in the buffer list.

The type of data returned to the buffers depends on which types of data that you request on the TYPE parameter. See *z/OS MVS Programming: Sysplex Services Guide* for more detailed information about how data is returned from this request.

**Note:** You cannot code BUFLIST with MODE=ASYNCRESPONSE.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 128-byte area that contains a list of buffer addresses.

#### **,BUFNUM=bufnum**

Use this input parameter to specify the number of buffers in the BUFLIST list. Valid BUFNUM values are 0 - 16. A value of zero indicates that no data is to be read into the buffers.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 1-byte field that contains the number of buffers (0 - 16) in the list (BUFLIST).

#### **,BUFSIZE=bufsize**

Use this input parameter to specify the size of the BUFFER area. See the BUFFER parameter description for valid buffer sizes.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a fullword field that contains the size of the buffer (BUFFER) in bytes.

#### **,BUFSTGKEY=CALLERS\_KEY**

#### **,BUFSTGKEY=bufstgkey**

Use this input parameter to specify a storage key that you define and use when referencing the buffers specified by BUFLIST or the buffer that is specified by BUFFER.

If you do not specify BUFSTGKEY, or if you specify BUFSTGKEY=CALLERS\_KEY, all references to one or more buffers are performed by using the caller's PSW key.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of an 8-bit field that contains the storage key in the format B'kkkkxxxx', where kkkk is the key and xxxx is ignored.

**,CONTOKEN=contoken**

Use this input parameter to specify the connect token that was returned by the IXLCONN service. The connect token uniquely identifies your connection to the list structure, and must be specified on each IXLLIST invocation.

The connect token is available in the IXLCONN answer area that is mapped by IXLYCONA.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 16-byte field that contains the connect token.

**,ENTRYID=entryid**

Use this input parameter to designate the entry identifier of the starting entry to be read.

Each entry identifier, which is assigned by list services when the entry is created, is unique within the structure.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 12-byte field that contains the entry identifier.

**,ENTRYKEY=NO\_ENTRYKEY****,ENTRYKEY=entrykey**

Use this input parameter with LISTNUM to designate the entry key of the starting entry to be read.

If there is a sublist of entries on the list all with the same key as the ENTRYKEY key, the LISTPOS parameter determines whether the entry at the HEAD or TAIL of the sublist is designated.

Specify ENTRYKEY only for structures that support keyed entries.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-byte field that contains the entry key.

**,ENTRYNAME=entryname**

Use this input parameter to designate the name of the starting entry to be read.

Specify ENTRYNAME only for structures that support named entries. Each entry name is unique within the structure.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-byte field that contains the entry name.

**,KEYCOMP=NO\_KEYCOMP****,KEYCOMP=keycomp**

Use this input parameter to specify a value to be compared to the entry key of each list entry in the designated list.

If the list entry that is to be processed does not have an entry key value that is equal to the specified KEYCOMP value, the entry is not processed. Processing continues with the next entry.

**Note:** The KEYCOMP parameter is valid only for list structures allocated in a coupling facility with CFLEVEL=1 or higher.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the 16-byte field that contains the value to be compared the entry key of the designated list entry.

**,KEYREQTYPE=EQUAL****,KEYREQTYPE=LESSOREQUAL****,KEYREQTYPE=GREATEROREQUAL**

Use this input parameter to specify how, if at all, the entry key of the entry to be read can differ from the entry key that is specified by ENTRYKEY. KEYREQTYPE applies only to locating an existing entry; it does not determine the key of a new entry.

**EQUAL**

The entry must have a key that equals the ENTRYKEY key.

**LESSOREQUAL**

The entry must have a key that is less than or equal to the ENTRYKEY key.

**GREATEROREQUAL**

The entry must have a key that is greater than or equal to the ENTRYKEY key.

**Note:**

1. When no entries on the list meet the requirements of the KEYREQTYPE, the request is failed with a return code X'8' and reason code IXLRSNCOEENTRY.
2. When LESSOREQUAL or GREATEROREQUAL is specified, if no entries on the list have an entry key value equal to the specified value, but entries exist with an entry key value greater than (if GREATEROREQUAL was specified), or less than (if LESSOREQUAL was specified) the entry key value that is specified, the entry with an entry key value closest to the value specified will be selected. When multiple entries have the same entry key value, LISTPOS is used to resolve whether the first, or last entry with the entry key value is selected.

**,LISTDIR=TOHEAD****,LISTDIR=TOTAIL**

Use this input parameter to specify the direction of processing from the designated starting entry.

**TOHEAD**

The direction of processing is from the designated entry to the head of the list.

**TOTAIL**

The direction of processing is from the designated entry to the tail of the list.

**Note:** See the LISTPOS parameter description for a note about the valid combinations of LISTDIR and LISTPOS.

**,LISTNUM=NO LISTNUM****,LISTNUM=listnum**

Use this input parameter to specify the number of the list on which the entries to be read reside. Use LISTNUM in the following ways:

- With LISTPOS and/or ENTRYKEY to designate the starting entry to be read.
- With either ENTRYID or ENTRYNAME to verify that the designated starting entry resides on the list specified by LISTNUM before proceeding with the request.
- With LOCBYCURSOR to designate the starting entry to be read.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 4-byte field that contains the number of the list.

**,LISTPOS=HEAD****,LISTPOS=TAIL**

Use this input parameter with LISTNUM to designate the starting entry to be read. The designated entry is at the HEAD or the TAIL of a list or sublist:

- If you specify ENTRYKEY and the list contains a sublist of one or more entries with the same key (or that satisfies the KEYREQTYPE criteria, if specified), then:
  - LISTPOS=HEAD designates the entry at the head of the sublist.
  - LISTPOS=TAIL designates the entry at the tail of the sublist.
- If you do not specify ENTRYKEY, then:
  - LISTPOS=HEAD designates the entry at the head of the list.
  - LISTPOS=TAIL designates the entry at the tail of the list.

**Note:** The direction of processing is always from an entry at one end of a list or sublist, through to the entry at the opposite end of the list. For instance, list services will process from the head of a sublist to the tail of the list, but not from the head of a sublist to the head of the list. As such, only the following combinations of LISTPOS and LISTDIR are valid:

- LISTPOS=HEAD and LISTDIR=TOTAIL
- LISTPOS=TAIL and LISTDIR=TOHEAD

**,LOCBYCURSOR**

Use this input parameter with LISTNUM to designate the starting entry to be read. The designated entry is the entry to which the LISTNUM list cursor is pointing.

Be aware that the list cursor could have been reset to zeros by a previous request that, for instance, deleted or moved the entry to which the list cursor points, or updated the list cursor after the last entry on the list had been processed. See *z/OS MVS Programming: Sysplex Services Guide* for more information about using the list cursor.

**,LOCKCOMP=NO LOCKCOMP****,LOCKCOMP=lockcomp**

This parameter has slightly different meanings based on the value that is specified for the LOCKOPER parameter. Generally, this input parameter specifies a connection identifier to be verified as the owner of the lock specified by LOCKINDEX. This verification is a prerequisite to the successful completion of the request.

When LOCKCOMP is specified, the completion of the request is conditional on there being no contention for the lock. If contention exists, the request fails .

The connection identifier is available from the IXLCONN answer area, which is mapped which is by IXLYCONA, in field CONACONID.

The effect of LOCKCOMP is based on the LOCKOPER value specified. See the description under LOCKOPER to see how each request is affected by this parameter.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 1-byte field that contains the connection identifier.

**,LOCKINDEX=NO LOCKINDEX****,LOCKINDEX=lockindex**

Use this input parameter to specify the index of the lock to be operated on as specified by LOCKOPER. The lock indexes begin with zero.

**Note:** You cannot code LOCKINDEX with MODE=ASYNCSNORESPONSE.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 4-byte field that contains the lock index.

**,LOCKMODE=COND**

Use this input parameter specify that the lock operation for the lock specified by LOCKINDEX performed conditionally; that is, only if there is no contention for the lock. If the specified lock is held by another user, the request is terminated with no change to the structure, and return and reason codes describing the termination are returned to the caller.

**,LOCKOPER=NOTHELD****,LOCKOPER=HELD BY**

Use this input parameter to specify the type of operation to be performed on the lock specified by LOCKINDEX.

**LOCKOPER=NOTHELD**

The request is performed only if the lock is not held by any connection. This option also ensures that the lock remains free for the duration of the request. If another connection holds the lock, your task is suspended until the lock is released, or your request fails depending on the LOCKMODE value you specify. See the LOCKMODE description for how to handle possible lock contention.

**LOCKOPER=HELD BY**

- When LOCKCOMP is not specified, the request is performed only if your connection holds the lock.
- When LOCKCOMP is specified, the request is performed only if the lock is held by the connection that is specified by LOCKCOMP.

**,MF=S**  
**,MF=(L,mfctrl)**  
**,MF=(L,mfctrl,mfattr)**  
**,MF=(L,mfctrl,0D)**  
**,MF=(M,mfctrl)**  
**,MF=(M,mfctrl,COMPLETE)**  
**,MF=(M,mfctrl,NOCHECK)**  
**,MF=(E,mfctrl)**  
**,MF=(E,mfctrl,COMPLETE)**  
**,MF=(E,mfctrl,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

#### **,mfctrl**

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

#### **,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

#### **,COMPLETE**

#### **,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

#### **COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if SMILE=*var* were an optional parameter and the default is SMILE=NO\_SMILE then it would not be documented. However, if the default was SMILE=:), then it would be documented because a value would be the default.

#### **NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

**,MODE=SYNCSUSPEND**  
**,MODE=SYNCECB**  
**,MODE=SYNCEXIT**  
**,MODE=SYNCTOKEN**  
**,MODE=ASYNCECB**  
**,MODE=ASYNCEXIT**  
**,MODE=ASYNCTOKEN**  
**,MODE=ASYNCSNORESPONSE**

Use this input parameter to specify:

- Whether the request is to be performed synchronously or asynchronously
- How you want to be notified of request completion

**MODE=SYNCSUSPEND**

The request is suspended until it can complete synchronously. To use this option, your program must be enabled for I/O and external interrupts.

**MODE=SYNCECB**

The request processes synchronously if possible. If the request processes asynchronously, the ECB specified by REQECB is posted when the request completes.

**MODE=SYNCEXIT**

The request processes synchronously if possible. If the request processes asynchronously, your complete exit is given control when the request completes.

**MODE=SYNCTOKEN**

The request processes synchronously if possible. If the request processes asynchronously, an asynchronous request token is returned in the area specified by REQTOKEN on invocation of the request. Use the returned request token on the IXLFComp macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=SYNCTOKEN is specified.

**MODE=ASYNCECB**

The request processes asynchronously. The ECB specified by REQECB is posted when the request completes.

**MODE=ASYNCEXIT**

The request processes asynchronously. Your complete exit is given control when the request completes.

**MODE=ASYNCTOKEN**

The request processes asynchronously. An asynchronous request token is returned in the area specified by REQTOKEN upon invocation of the request. Use the returned request token on the IXLFComp macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=ASYNCTOKEN is specified.

**MODE=ASYNCSNORESPONSE**

The request processes asynchronously. No notification of request completion is provided. The answer area (ANSAREA) fields will not contain valid information when this mode is specified.

**Note:** You cannot code MODE=ASYNCSNORESPONSE with LOCKINDEX, BUFFER, or BUFLIST.

**,PAGEABLE=YES**

**,PAGEABLE=NO**

Use this input parameter to identify whether the storage areas specified by BUFFER or BUFLIST are in pageable or potentially pageable storage.

**YES**

Specify this option to indicate that the BUFFER or BUFLIST buffers reside in pageable virtual storage. XES performs the required page fixing to fix the buffers in real storage while the cache or list request transfers data to or from the coupling facility.

This includes storage obtained from pageable subpools, disabled reference (DREF) storage, and may include storage that has the potential to become pageable during the processing of a request.

(An example is address space storage owned by any swappable address space, for which a PGSER FIX has been successfully processed, but for which the owning address space gets swapped during processing of a cache or list request.) This does not include implicitly non-pageable storage (for example, storage obtained from non-pageable subpools).

The system takes responsibility for managing binds to central storage for the duration of the cache or list request, regardless of what address space owns the storage or whether the storage-owning address space is swappable or nonswappable. The storage can be owned by any address space.

High shared virtual storage areas (above 2GB) may not be used.

## NO

Specify this option to indicate that the BUFFER or BUFLIST buffers reside in non-pageable virtual storage. XES does not page fix the buffers in real storage.

This includes implicitly non-pageable storage areas (for example, storage obtained from non-pageable subpools), and may include storage that has the potential to become pageable during the processing of a request (An example is address space storage owned by any swappable address space, for which a PGSER FIX has been successfully processed, but for which the owning address space gets swapped-out during processing of a cache or list request.)

The system takes responsibility for managing binds to central storage for the duration of the cache or list request, if and only if the non-pageable storage is owned by either the requester's address space or the connector's address space. If the storage is owned by any other address space, then the invoker is responsible for ensuring that the virtual storage remains non-pageable for the duration of the request (including the case in which the storage is owned by a swappable address space that is swapped during processing of an IXLCACHE or IXLLIST request). Subject to this consideration, the storage can be owned by any address space. See *z/OS MVS Programming: Sysplex Services Guide*.

## **,PLISTVER=IMPLIED\_VERSION**

## **,PLISTVER=MAX**

## **,PLISTVER=plistver**

Use this input parameter to specify the version of the macro. See “[Understanding IXLLIST Version Support](#)” on page 965 for a description of the options available with PLISTVR.

## **,REQDATA=NO\_REQDATA**

## **,REQDATA=reqdata**

Use this input parameter with MODE=SYNCEXIT or MODE=ASYNCEXIT to pass any data you choose to the complete exit. The exit will get control only if the request is processed asynchronously.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the data to be passed to the complete exit.

## **,REQECB=reqecb**

Use this output parameter with either MODE=SYNCECB or MODE=ASYNCECB to specify the address of an ECB, which is to be posted when the request completes if the request was processed asynchronously.

Before coding REQECB, you must ensure that:

- You initialize the ECB before you issue the request.
- The ECB resides in either common storage or the home address space where IXLCONN was issued.
- Any tasks that wait for the ECB to be posted reside in the home address space where IXLCONN was issued.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that contains the address of the ECB to be posted when the request completes. The ECB must be aligned on a fullword boundary.



**,REQID=NO\_REQID****,REQID=reqid**

Use this input parameter to specify a user-defined request identifier to be associated with the request. You can specify this request identifier on the IXPURGE macro to cancel a request that has not yet been processed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the user-defined request identifier.

**,REQTOKEN=reqtoken**

Use this output parameter with either MODE=SYNCTOKEN or MODE=ASYNCTOKEN to specify the address of a storage area to receive the request token that is returned when the request will be processed asynchronously. This token, which uniquely identifies the request, must be used as input to the IXLFCOMP macro, which you use to determine if the request has completed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field where the system will put the request token.

**,RETCODE=retcode**

Use this output parameter to specify a field to contain the return code. (The return code is also returned in register 15.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the return code when the request has completed.

**,RSNCODE=rsncode**

Use this output parameter to specify a field to contain the reason code returned, if applicable. (The reason code is also returned in register 0.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the reason code (if any) when the request has completed.

**,TYPE=ENTDATA****,TYPE=ADJDATA****,TYPE=ECONTROLS**

Use this input parameter to specify the type of data to be read from each processed entry.

**ENTDATA**

Entry data is read.

**ADJDATA**

Adjunct data is read.

**ECONTROLS**

List entry control data is read.

You can specify one, two, or all three of these data types, and you can code them in any order on the TYPE parameter. When more than one data type is specified, the format of the parameter would be: TYPE=(ENTDATA,ADJDATA,ECONTROLS). The order in which you code the values does not affect the order in which the system returns the data in the BUFFER area or BUFLIST buffers. See [z/OS MVS Programming: Sysplex Services Guide](#) for more information about how the data is arranged in the.

**,VERSCOMP=NO\_VERSCOMP****,VERSCOMP=verscomp**

Use this input parameter to specify a version number to be compared to the version number of each entry from the starting entry to the head or tail of the list (as specified by LISTDIR). Only those entries with a version that meets the condition specified by the VERSCOMPTYPE parameter are read.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-byte field that contains the version number.

**,VERSCOMPTYPE=EQUAL****,VERSCOMPTYPE=LESSOREQUAL**

Use this input parameter to specify how a list entry version comparison as specified by VERSCOMP is to be performed.

**Note:** The VERSCOMPTYPE parameter is valid only for list structures allocated in a coupling facility with CFLEVEL=1 or higher.

**VERSCOMPTYPE=EQUAL**

The version number for the list entry must be equal to the value specified for VERSCOMP.

**VERSCOMPTYPE=LESSOREQUAL**

The version number for the list entry must be less than or equal to the value specified for VERSCOMP.

## ABEND Codes

---

Abend X'026' (See *z/OS MVS System Codes* for more information on this abend.)

## Return and Reason Codes

---

When the system returns control to your program:

- GPR 15 (and RETCODE, if specified) contains a return code.
- If the return code is not zero, GPR 0 (and RSNCODE, if specified) contains a reason code.

**Note:** The return and reason codes will also be put into the answer area (mapped by IXLYLAA), if there is one, when the request completes.

The IXLYCON macro provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

**0**

IXLRETCODEOK

**4**

IXLRETCODEWARNING

**8**

IXLRETCODEPARMERROR

**C**

IXLRETCODEENVERROR

**10**

IXLRETCODECOMPERROR

The following table contains hexadecimal return and reason codes, the equate symbols associated with each reason code, and the meaning and suggested action for each return and reason code.

Table 71. Return and Reason Codes for IXLLIST REQUEST=READ\_LIST Macro

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0	None.	<p><b>Meaning:</b> If you specified a MODE value of ASYNCECB, ASYNCEXIT, ASYNCTOKEN, or ASYNCNORESPONSE, the request has been successfully initiated. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed. For any other MODE value, the request has successfully completed.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=ASYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=ASYNCEXIT, the connection's complete exit will be given control when the request has completed.</li> <li>• If you specified MODE=ASYNCTOKEN, use REQTOKEN when calling IXLFComp to determine when the request has completed.</li> </ul>
4	xxxx0402	<p><b>Equate Symbol:</b> IXLRNCodeASYNCH</p> <p><b>Meaning:</b> The request could not be processed synchronously. It will be processed asynchronously. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed.</p> <ul style="list-style-type: none"> <li>• If MODE=SYNCECB was specified, the ECB specified for REQECB will be posted when the request has finished.</li> <li>• If MODE=SYNCEXIT was specified, the connection's complete exit will be given control when the request has finished.</li> <li>• If MODE=SYNCTOKEN was specified, an asynchronous request token was returned in the area specified by REQTOKEN. This request token may be specified on the IXLFComp macro to determine when the request has finished.</li> </ul> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=SYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=SYNCEXIT, the connection's complete exit will be given control when the request has completed.</li> <li>• If you specified MODE=SYNCTOKEN, use REQTOKEN when calling IXLFComp to determine when the request has completed.</li> </ul>

Table 71. Return and Reason Codes for IXLLIST REQUEST=READ\_LIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	xxxx0409	<p><b>Equate Symbol:</b> IXLRSNCODETIMEOUT</p> <p><b>Meaning:</b> The request has completed prematurely because it exceeded the coupling facility model-dependent time-out criteria. The following information has been returned in the answer area:</p> <ul style="list-style-type: none"> <li>• The number of successfully processed entries (field LAAREADCNT).</li> <li>• The list entry controls for the first UNPROCESSED entry in the list sequence (field LAALCTL).</li> <li>• The list entry controls for the first PROCESSED entry in the list sequence (field LAARLRMLCTL).</li> </ul> <p><b>Action:</b> Reissue the request beginning with the first unprocessed entry specified as the starting entry. The list entry controls (LAALCTL) will contain the ENTRYID, ENTRYNAME, or ENTRYKEY of the first unprocessed entry. Before you reissue the request, be sure to process all the returned data. For more information about premature completion, see <i>z/OS MVS Programming: Sysplex Services Guide</i>.</p>

Table 71. Return and Reason Codes for IXLLIST REQUEST=READ\_LIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	xxxx040D	<p><b>Equate Symbol:</b> IXLRSNCODEBADREADADJDATA</p> <p><b>Meaning:</b> Program error. The request specified that adjunct data was to be read, but the storage area specified by ADJAREA is not addressable. All other requested data was retrieved and the following fields in the answer area have been filled in:</p> <ul style="list-style-type: none"> <li>• LAARLRLCTL - The first PROCESSED entry in the list sequence.</li> <li>• LAAREADCNT - The number of entries processed successfully</li> <li>• LAALCTL (if the request completed prematurely as well) - The list entry controls for the first UNPROCESSED entry in the list sequence.</li> </ul> <p><b>Note:</b> Only the adjunct data for the first entry in the list is placed in the ADJAREA. The buffers should contain the adjunct data for the other entries in the list.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the ADJAREA address.</li> <li>• ADJAREA must be addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are calling IXLLIST while disabled, ADJAREA must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLLIST in AR-mode and you specified the ADJAREA address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are calling IXLLIST in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLIST macro.</li> <li>• If LAALCTL is not zeros, the request completed prematurely. See the action suggested for return code X'4' reason code X'xxxx0409'.</li> </ul> <p>Correct the address specified by ADJAREA, and rerun the request asking for adjunct data only.</p>

Table 71. Return and Reason Codes for IXLLIST REQUEST=READ\_LIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	xxxx040F	<p><b>Equate Symbol:</b> IXLRSNCODEBUFFERFULL</p> <p><b>Meaning:</b> The request has completed prematurely because the buffer specified by BUFFER or the buffers specified by BUFLIST are full. The following information has been returned in the answer area:</p> <ul style="list-style-type: none"> <li>• The number of successfully processed entries (field LAAREADCNT).</li> <li>• The list entry controls for the first unprocessed entry in the list sequence (field LAALCTL).</li> <li>• The list entry controls for the first processed entry in the list sequence (field LAARLRLCTL).</li> </ul> <p><b>Action:</b> Reissue the request beginning with the first unprocessed entry. If the same buffers are used, the data returned from the original request will be overwritten, so be sure to process the returned data first. For more information about premature completion, see <i>z/OS MVS Programming: Sysplex Services Guide</i>.</p>
4	xxxx0410	<p><b>Equate Symbol:</b> IXLRSNCODELOCKCOND</p> <p><b>Meaning:</b> For a LOCKOPER=HELDDBY request, or a LOCKMODE=COND request, or a request that specified LOCKCOMP, the request could not be completed successfully because the specified lock is not currently held as required. The connection identifier of the lock owner is returned in the answer area (LAACONID field).</p> <p><b>Action:</b> Retry the request, or obtain the lock as required, and retry the request. If you are unable to get the lock, check the ID in the LAACONID field and determine if some recovery is necessary.</p>
4	xxxx0412	<p><b>Equate Symbol:</b> IXLRSNCODELOCKHELDDBYSYS</p> <p><b>Meaning:</b> The lock is not held by any connection, but instead is held by the system. For a request that specified any of the following, the request could not be completed successfully because the specified lock is not currently held as required:</p> <ul style="list-style-type: none"> <li>• LOCKOPER=NOTHELD with either of: <ul style="list-style-type: none"> <li>– LOCKMODE=COND</li> <li>– LOCKCOMP</li> </ul> </li> <li>• LOCKOPER=HELDDBY</li> </ul> <p><b>Action:</b> Retry the request, or obtain the lock as required, and retry the request.</p>

Table 71. Return and Reason Codes for IXLLIST REQUEST=READ\_LIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0801	<p><b>Equate Symbol:</b> IXLRSNCODEBADPARMLIST</p> <p><b>Meaning:</b> Program error. The parameter list for this request is not addressable. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Verify that:</p> <ul style="list-style-type: none"> <li>• The parameter list address is uncorrupted.</li> <li>• The parameter list is addressable in the caller's primary address space.</li> <li>• If you are issuing this macro while disabled, the parameter list resides in either page-fixed or DREF storage.</li> <li>• If you are issuing this macro in AR-mode and you specified the parameter list address using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If you are issuing this macro in AR-mode, SYSSTATE ASCENV=AR must be issued before issuing this macro.</li> </ul>
8	xxxx0804	<p><b>Equate Symbol:</b> IXLRSNCODEBADVERSIONNUM (or IXLRSNCODEBADVERSION#)</p> <p><b>Meaning:</b> The version number in the macro parameter list is not compatible with the level of XES currently being used. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that your program did not overlay the parameter list storage.</li> <li>• Verify that your program was assembled with the correct macro library for the release of MVS on which your program is running.</li> </ul>

Table 71. Return and Reason Codes for IXLLIST REQUEST=READ\_LIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx080A	<p><b>Equate Symbol:</b> IXLRSNCODEBADCONTOKEN</p> <p><b>Meaning:</b> Program error. The specified connect token (CONTOKEN) was not valid for one of the following reasons:</p> <ol style="list-style-type: none"> <li>1. The user with the connection identifier represented by the token has disconnected from the structure.</li> <li>2. The connector's task (the task that issued IXLCONN) ended.</li> <li>3. The specified token is not the token that was returned from IXLCONN.</li> <li>4. The request was issued from an address space other than the address space in which IXLCONN was issued.</li> <li>5. The connect token was invalidated during rebuild.</li> <li>6. The connect token was invalidated by XES.</li> </ol> <p><b>Note:</b> The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Take the action with the corresponding meaning.</p> <ol style="list-style-type: none"> <li>1. Discontinue use of the structure. Perform recovery and cleanup for the structure and check the protocol for the use of the structure to determine why you did not know of the disconnection.</li> <li>2. Discontinue use of the structure. Perform recovery and cleanup for the structure and check the protocol for the use of the structure to determine why you did not know that the task ended.</li> <li>3. The contoken is returned in the CONACONTOKEN field of the answer area specified on the IXLCONN request.</li> <li>4. Issue your request from the same address space the IXLCONN was issued in.</li> <li>5. Wait for the rebuild to complete, and try again.</li> <li>6. Discontinue use of the structure. Perform recovery and cleanup for the structure.</li> </ol>
8	xxxx0822	<p><b>Equate Symbol:</b> IXLRSNCODEBADREADTYPE</p> <p><b>Meaning:</b> Program error. You specified that either entry data or adjunct data should be returned, but the list structure does not contain the type of data you requested. No data is returned.</p> <p><b>Action:</b> Ensure that you are requesting the type of data you need from the structure and that the structure supports that type of data. Issue IXLMG to get more information about the structure.</p>



Table 71. Return and Reason Codes for IXLLIST REQUEST=READ\_LIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0824	<p><b>Equate Symbol:</b> IXLRSNCODEWRONGSTRTYPE</p> <p><b>Meaning:</b> Program error. The connection specified by CONTOKEN is not to a list structure.</p> <p><b>Action:</b> Specify the appropriate CONTOKEN value, and rerun your program. The CONTOKEN is returned in the answer area provided on the IXLCONN invocation. The type of structure built is passed on the TYPE parameter of the IXLCONN macro.</p>
8	xxxx0825	<p><b>Equate Symbol:</b> IXLRSNCODENOENTRY</p> <p><b>Meaning:</b> Program error. The designated list entry does not exist; therefore, no entries were processed.</p> <p><b>Action:</b> None necessary. However, if this return code and reason code are unexpected, you should check the way the list entry was created. Examine the parameters specified for locating the list entry on the invocation of this macro.</p>
8	xxxx0833	<p><b>Equate Symbol:</b> IXLRSNCODEBADPGBLATTR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list is specified as being pageable (PAGEABLE=YES), but is not.</p> <p><b>Action:</b> Change the buffer area(s) to pageable storage, or specify PAGEABLE=NO. See the PAGEABLE parameter description for specification instructions.</p>

Table 71. Return and Reason Codes for IXLLIST REQUEST=READ\_LIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0834	<p><b>Equate Symbol:</b> IXLRSNCODEBADNONPGBLATTR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list, is specified as being nonpageable (PAGEABLE=NO), but is either pageable or not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The type of storage in the buffer area corresponds to the value you specified on the PAGEABLE parameter. See the PAGEABLE parameter description for specification instructions.</li> <li>• The correct buffer address was used.</li> <li>• The buffer area was not previously freed.</li> <li>• If you are calling IXLLIST while disabled, the buffers must reside in either page-fixed or DREF storage.</li> <li>• The buffer areas were allocated in a storage key that matches the key specified by the BUFSTGKEY parameter, or, if BUFSTGKEY is omitted, the caller's PSW key.</li> <li>• If BUFLIST was specified and your program is running in AR-mode: <ul style="list-style-type: none"> <li>– If the BUFFER or BUFLIST parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>– The BUFALET specification is correct.</li> <li>– You specified SYSSTATE ASCENV=AR before issuing the IXLLIST macro.</li> </ul> </li> </ul>
8	xxxx0835	<p><b>Equate Symbol:</b> IXLRSNCODEBADDATAADDR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list, is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The correct buffer address was used.</li> <li>• The buffer area was not previously freed.</li> <li>• The buffer area was allocated in a storage key that matches the key specified using the BUFSTGKEY parameter, or, if BUFSTGKEY is omitted, the caller's PSW key.</li> <li>• If BUFLIST was specified and your program is running in AR mode: <ul style="list-style-type: none"> <li>– The BUFALET specification is correct.</li> <li>– You specified SYSSTATE ASCENV=AR before issuing the macro.</li> </ul> </li> </ul>

Table 71. Return and Reason Codes for IXLLIST REQUEST=READ\_LIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0836	<p><b>Equate Symbol:</b> IXLRSNCODEBADREALADDR</p> <p><b>Meaning:</b> Program error. Real storage addresses were provided in the BUFLIST list, but one of the buffers is not addressable in central storage.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that BUFADDRTYPE was specified as you intended.</li> <li>• Ensure that the buffer addresses specified by BUFLIST are valid.</li> </ul>
8	xxxx0838	<p><b>Equate Symbol:</b> IXLRSNCODEBADANSAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by ANSAREA is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The answer area address specified by ANSAREA is valid.</li> <li>• If the caller is running in AR-mode and the ANSAREA parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• The address specified for ANSAREA is addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are calling IXLLIST while disabled, ANSAREA must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLLIST in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLIST macro.</li> </ul>
8	xxxx0839	<p><b>Equate Symbol:</b> IXLRSNCODEBADREQTOKENAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by REQTOKEN is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The request token area specified by REQTOKEN is valid.</li> <li>• If the caller is running in AR-mode and the REQTOKEN parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If you are calling IXLLIST while disabled, REQTOKEN must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLLIST in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLIST macro.</li> </ul>

Table 71. Return and Reason Codes for IXLLIST REQUEST=READ\_LIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx083D	<p><b>Equate Symbol:</b> IXLRSNCODEBADANSLEN</p> <p><b>Meaning:</b> Program error. The length of the answer area, as specified by ANSLEN, is not sufficient to contain answer area information. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Increase the size of the answer area, and rerun your program. The minimum size of the ANSAREA is in the LAA_LEN field of the IXLYLAA mapping macro.</p>
8	xxxx0840	<p><b>Equate Symbol:</b> IXLRSNCODEBADENTRYLIST</p> <p><b>Meaning:</b> The entry designated by ENTRYID or ENTRYNAME exists in the structure, but does not reside on the list specified by LISTNUM. The list entry controls for the entry are returned in the answer area (field LAALCTL).</p> <p><b>Action:</b> None necessary. However, if you did not expect this return and reason code, you might want to verify what list this entry is on. Maybe the entry was moved, or you designated the wrong list in LISTNUM. Check the protocol for using this list.</p> <p>The information returned in the LAALCTL field of the answer area contains the number of the list that this list entry is on as well as other control information.</p>
8	xxxx0845	<p><b>Equate Symbol:</b> IXLRSNCODENONAMES</p> <p><b>Meaning:</b> Program error. A list entry was designated by entry name, but the structure does not support entry names.</p> <p><b>Action:</b> Ensure that you are connected to the intended structure. Ensure that the correct specification was made for REFOPTION on the IXLCONN macro. You may want to issue IXLMG to get more information about the structure.</p>
8	xxxx0846	<p><b>Equate Symbol:</b> IXLRSNCODEBADLOCKINDEX</p> <p><b>Meaning:</b> Program error. The specified LOCKINDEX exceeds the size of the lock table for the structure.</p> <p><b>Action:</b> Correct LOCKINDEX to specify an index that is contained within the lock table. The maximum value for the LOCKINDEX is one less than the value specified by the LOCKENTRIES parameter on the IXLCONN macro.</p>
8	xxxx0847	<p><b>Equate Symbol:</b> IXLRSNCODEBADLISTNUMBER</p> <p><b>Meaning:</b> Program error. The specified LISTNUM value exceeds the number of lists for the structure.</p> <p><b>Action:</b> Correct LISTNUM to specify a list number that exists in the structure. The number of lists allocated for a structure is determined on the LISTHEADERS parameter of the IXLCONN macro. The list numbers go from 0 to n-1, where n is the number of lists specified.</p>

Table 71. Return and Reason Codes for IXLLIST REQUEST=READ\_LIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx084A	<p><b>Equate Symbol:</b> IXLRNOCODENOKEYS</p> <p><b>Meaning:</b> Program error. The structure does not support the use of entry keys. The IXLLIST request type either required the structure to support entry keys or designated a list entry or list position by list number and entry key.</p> <p><b>Action:</b> Ensure that you are connected to the intended structure. The use of keys for a structure is determined by the REFOPTION keyword on the IXLCONN macro.</p>
8	xxxx084B	<p><b>Equate Symbol:</b> IXLRNOCODENOLOCKS</p> <p><b>Meaning:</b> Program error. A locking operation was requested, but the structure does not contain a lock table.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• You are connected to the intended structure.</li> <li>• You intended to perform a locking operation.</li> </ul> <p>The use of locks is determined by the LOCKENTRIES keyword on the IXLCONN macro.</p>
8	xxxx0851	<p><b>Equate Symbol:</b> IXLRNOCODENOSUSPENDISABLE</p> <p><b>Meaning:</b> Program error. The request failed because MODE=SYNCSUSPEND was specified, but the caller is disabled and cannot be suspended.</p> <p><b>Action:</b> Either specify another MODE value, or become enabled (release the CPU lock), then reissue the request.</p>
8	xxxx0854	<p><b>Equate Symbol:</b> IXLRNOCODEBADLOCKCOMP</p> <p><b>Meaning:</b> Program error. The connection identifier specified for LOCKCOMP is not valid.</p> <p><b>Action:</b> The connection identifier is returned in the answer area (mapped by IXLYCONA) when the IXLCONN macro was issued.</p>
8	xxxx0859	<p><b>Equate Symbol:</b> IXLRNOCODEBADLISTAUTH</p> <p><b>Meaning:</b> The list authority specified for AUTHCOMP failed the comparison specified by AUTHCOMPTYPE for the list authority of the specified list. The current list authority (field LAALISTAUTH) and description (field LAALISTDESC) are returned in the answer area.</p> <p><b>Action:</b> None required; however you might want to take some action depending on your application. The list authority is set to binary zeros when the structure is allocated. When IXLLIST is issued with the NEWAUTH keyword, the list authority is changed if the correct comparison list authority value is specified. Check the answer area to determine the list authority value for the list specified. Verify that the correct list number was specified.</p>

Table 71. Return and Reason Codes for IXLLIST REQUEST=READ\_LIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0864	<p><b>Equate Symbol:</b> IXLRNCODEBADBUFSIZE</p> <p><b>Meaning:</b> Program error. The size of the BUFFER area or the buffer areas specified by BUFLIST is not large enough to contain the data being read. No data is returned. The answer area (field LAALCTL) contains the list entry controls for the first entry selected for processing.</p> <p><b>Action:</b> If more space is available, specify a larger buffer size, and reissue the request. Consider using the BUFLIST parameter instead of the BUFFER parameter.</p>
8	xxxx0865	<p><b>Equate Symbol:</b> IXLRNCODEBADBUFSPEC</p> <p><b>Meaning:</b> Program error. There is an error in the buffer specification</p> <p><b>Action:</b> Check the following:</p> <ul style="list-style-type: none"> <li>• If BUFLIST was specified, check the requirements for BUFLIST and BUFNUM.</li> <li>• If BUFFER was specified, check the requirements for BUFFER and BUFSIZE.</li> <li>• Buffer pointer(s) in BUFLIST.</li> <li>• Buffer boundaries.</li> </ul>
8	xxxx0866	<p><b>Equate Symbol:</b> IXLRNCODEBADBUFKEY</p> <p><b>Meaning:</b> Program error. The buffer storage key specified by BUFSTGKEY is incorrect or BUFSTGKEY was not specified and the PSW key does not match the key of the buffers.</p> <p>The data cannot be stored in the specified buffer area.</p> <p><b>Action:</b> Check the following:</p> <ul style="list-style-type: none"> <li>• Determine if the key of the storage being used for the buffers is different from the PSW key.</li> <li>• If BUFSTGKEY was specified, verify that the key was put in the correct bits (see the explanation of this parameter for more information).</li> <li>• If you are calling IXLLIST in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLIST macro.</li> </ul>
8	xxxx0867	<p><b>Equate Symbol:</b> IXLRNCODEBADBUFLIST</p> <p><b>Meaning:</b> Program error. The 128-byte storage area specified by BUFLIST is not addressable.</p> <p><b>Action:</b> Ensure that the address specified by BUFLIST is valid.</p>

Table 71. Return and Reason Codes for IXLLIST REQUEST=READ\_LIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C06	<p><b>Equate Symbol:</b> IXLRSNCODENOCNN</p> <p><b>Meaning:</b> Environmental error. This system does not have connectivity to the coupling facility that contains the list structure. Possible reasons for this are:</p> <ul style="list-style-type: none"> <li>• The operator issued VARY PATH,OFFLINE.</li> <li>• The operator issued CONFIG CHP,OFFLINE.</li> <li>• Hardware errors to the coupling facility.</li> <li>• Facility or path failure to the coupling facility.</li> </ul> <p><b>Action:</b> Begin rebuilding the structure on a different coupling facility, or disconnect from the structure.</p>
C	xxxx0C13	<p><b>Equate Symbol:</b> IXLRSNCODEREQPURGED</p> <p><b>Meaning:</b> Environmental error. The request was purged prior to completion of the request. Possible reasons include:</p> <ul style="list-style-type: none"> <li>• The connector failed.</li> <li>• The connector disconnected.</li> <li>• The requestor failed.</li> <li>• The request was purged by IXLPURGE.</li> <li>• Requests were purged when the connector provided an IXLEERSP response for the Rebuild Stop or Rebuild Cleanup event.</li> <li>• The secondary address space was no longer valid.</li> </ul> <p><b>Action:</b> None if this is expected. Otherwise, determine why the connector failed.</p>
C	xxxx0C14	<p><b>Equate Symbol:</b> IXLRSNCODESTATUSUNKNOWN</p> <p><b>Meaning:</b> Environmental error. The request has completed, but the final disposition of the request cannot be determined. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the validity of your data by comparing the expected results with what is in the coupling facility.</li> <li>• Consider running rebuild processing to move the structure to attempt to protect it from further errors of this kind.</li> </ul>
C	xxxx0C25	<p><b>Equate Symbol:</b> IXLRSNCODESTRFAILURE</p> <p><b>Meaning:</b> Environmental error. The list structure failed prior to completion of the request.</p> <p><b>Action:</b> Either rebuild or disconnect from the structure.</p>

Table 71. Return and Reason Codes for IXLLIST REQUEST=READ\_LIST Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0CA0	<b>Equate Symbol:</b> IXLRSNCODEQUIESCEDSUSPENDFAIL <b>Meaning:</b> Environmental error. The request is failed because the structure is quiesced for a system-managed process and SUSPEND=FAIL is specified on the IXLCONN. <b>Action:</b> None, if this is expected.
C	xxxxFFFF	<b>Equate Symbol:</b> IXLRSNCODENOTAVAILABLE <b>Meaning:</b> Environmental error. XES functions are not available. This could occur because the coupling facility hardware necessary to provide XES functions is not present. <b>Action:</b> Re-IPL the system, or follow your particular failure management protocol.
10	xxxx10xx	<b>Meaning:</b> System error. XES processing failure. The state of the involved structure and the disposition of the request are unpredictable. The answer area (ANSAREA) fields do not contain valid information. <b>Action:</b> Contact the IBM support center.



---

## Chapter 72. IXLLIST REQUEST=READ\_MULT

### Description

---

A READ\_MULT request allows you to read multiple entries from the list structure into the storage areas specified by ANSAREA and/or BUFLIST or BUFFER and/or ADJAREA. You can restrict processing to only those entries that satisfy either one or both of the following criteria:

- Those residing on a certain list (which you specify using LISTNUM)
- Those containing data that satisfies a version number comparison (which you specify using VERSCOMP).

For structures allocated in a CFLEVEL=0 coupling facility, if you specify neither LISTNUM nor VERSCOMP, every entry in the structure is read. The order in which the entries are read is unpredictable.

With a coupling facility of CFLEVEL=1 or higher, you can restrict processing to only those entries that satisfy one or more of the following criteria:

- A successful comparison of the list authority value for the target list with a user-specified value. You specify the list authority requirements using AUTHCOMP and AUTHCOMPTYPE.
- A successful comparison of the list entry key value that is being processed with a user-specified value (KEYCOMP).
- A successful comparison of the version number, which is enhanced to include a less-than-or-equal comparison operation.

For each processed entry, any combination of the following types of data can be read depending on what you specify on the TYPE parameter:

- List entry controls
- Entry data
- Adjunct data

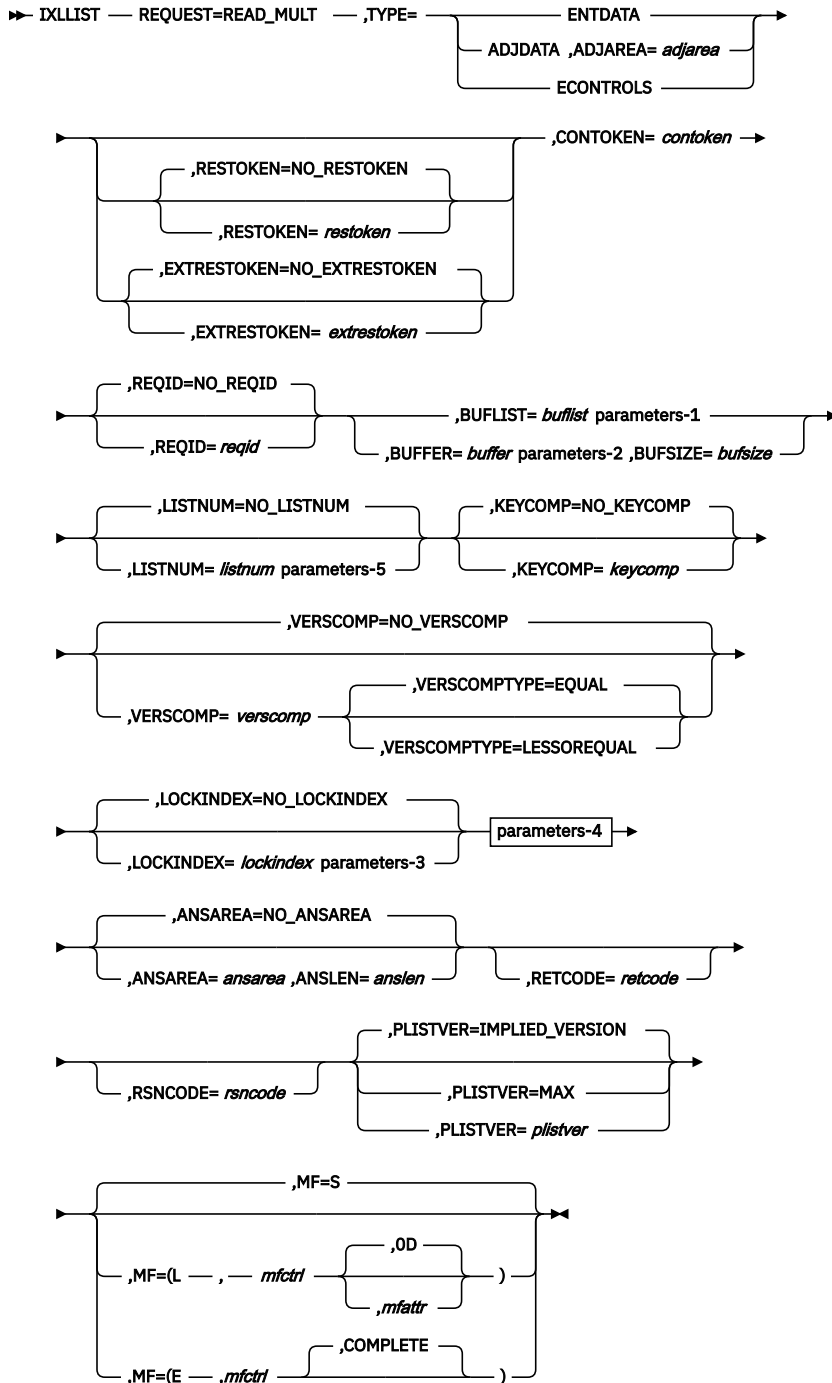
If the request completes prematurely because it exceeds the coupling facility model-dependent time-out criteria or the specified buffer area (BUFFER or BUFLIST) is full, either a restart token (RESTOKEN) or an extended restart token (EXTRESTOKEN) is returned in the answer area (field LAARESTOKEN or LAAEXTRESTOKEN). The token can be specified on the next READ\_MULT request to resume processing with the next data item to be processed. Resumed requests are processed identically whether using the RESTOKEN or EXTRESTOKEN to specify the starting location.

### Syntax Diagram

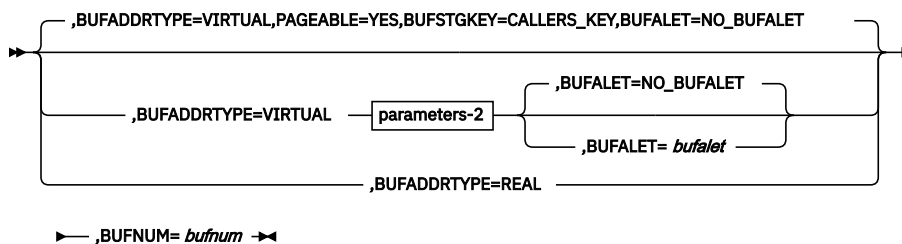
---

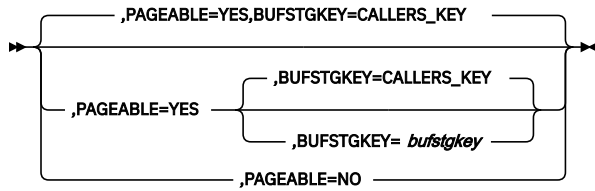
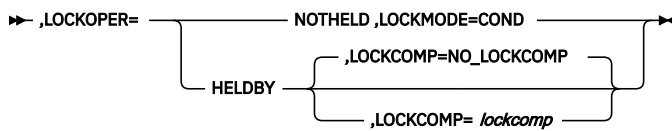
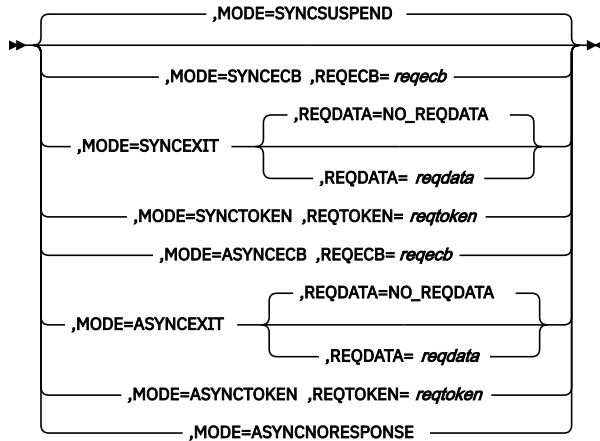
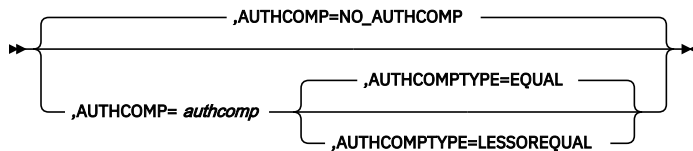
The syntax diagram for IXLLIST REQUEST=READ\_MULT is as follows:

## main diagram



## parameters-1



**parameters-2****parameters-3****parameters-4****parameters-5****Note:**

1. If MODE=SYNCTOKEN or MODE=ASYNCTOKEN is specified, ANSAREA=*ansarea* and ANSLLEN=*anslen* are required.
2. If MODE=ASYNCNORESPONSE is specified, BUFFER, BUFLIST, and LOCKINDEX may not be specified.

## Parameter Descriptions

The parameter descriptions for REQUEST=READ\_MULT are listed in alphabetical order. Default values are underlined:

**REQUEST=READ\_MULT**

Use this input parameter to read multiple entries from within the entire structure.

**,ADJAREA=*adjarea***

Use this output parameter to specify a storage area to contain the adjunct data that was read from the first processed entry. Adjunct data is returned only if ADJDATA is specified on the TYPE parameter.

Specify ADJAREA only for structures that support adjunct data. (Adjunct areas for a structure are established through the IXLCONN macro.)

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) to a 64-byte area (the adjunct area) to contain the adjunct data.

**,ANSAREA=NO\_ANSAREA**

**,ANSAREA=ansarea**

Use this output parameter to specify an answer area to contain information returned from the request. The format of the answer area is described by mapping macro IXLYLAA.

On a successful completion of the request, the answer area contains:

- List entry controls (field LAARLRMLCTLS mapped by IXLYLCTL) from the first processed list entry (if ECONTROLS is specified on the TYPE parameter).
- The number of processed entries (field LAAREADCNT).

If the request completes prematurely, the answer area contains:

- A restart token (LAARESTOKEN) or extended restart token (LAAEXTRESTOKEN) that can be specified on a subsequent READ\_MULT request. (See the RESTOKEN and EXTRESTOKEN parameter descriptions.)
- The number of processed entries (LAAREADCNT).

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an area (with a length of ANSLEN) where the information from the request will be put.

**,ANSLEN=anslen**

Use this input parameter to specify the size of the storage area specified by ANSAREA.

Check the prologue of the IXLYLAA mapping macro for the minimum required size of the answer area, or use the length field (LAA\_LEN) in the LAA.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 2-byte field that contains the size, in bytes, of the answer area.

**,AUTHCOMP=NO\_AUTHCOMP**

**,AUTHCOMP=authcomp**

Use this input parameter to specify a value to be compared to the list authority value of the list specified by LISTNUM. You must supply the LISTNUM parameter.

If the comparison does not meet the condition specified by the AUTHCOMPTYPE parameter (EQUAL or LESSOREQUAL), the request fails.

**Note:** The AUTHCOMP parameter is valid only for list structures allocated in a coupling facility with CFLEVEL=1 or higher.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the 16-byte field that contains the list authority value.

**,AUTHCOMPTYPE=EQUAL**

**,AUTHCOMPTYPE=LESSOREQUAL**

Use this input parameter specify how the list audit comparison is to be performed.

#### **EQUAL**

The list authority for the list specified by LISTNUM must be equal to the value specified for AUTHCOMP.

#### **LESSOREQUAL**

The list authority for the list specified by LISTNUM must be less than or equal to the value specified for AUTHCOMP.

**Note:** The AUTHCOMPTYPE parameter is valid only for list structures allocated in a coupling facility with CFLEVEL=1 or higher.

**,BUFADDRTYPE=VIRTUAL****,BUFADDRTYPE=REAL**

Use this input parameter to specify whether the buffer addresses specified in the BUFLIST list are virtual storage or real storage addresses.

**VIRTUAL**

The buffer addresses are virtual storage addresses. The virtual storage can be pageable or nonpageable. See the PAGEABLE parameter for information about managing storage binds when specifying virtual storage addresses.

**REAL**

The buffer addresses are real storage addresses.

It is the caller's responsibility to manage the binds between the data buffer virtual storage and the real storage addresses provided. The caller must ensure that the data buffer virtual storage remains bound to the real storage addresses provided until the request completes.

**,BUFALET=NO\_BUFALET****,BUFALET=bufalet**

Use this input parameter to specify an access list entry token (ALET) to be used in referencing all of the buffers specified by BUFLIST.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 4-byte field that contains the ALET.

**,BUFFER=buffer**

Use this output parameter to specify a buffer area to hold data that is read from list entries.

You must ensure that the storage area specified by BUFFER:

- Is a multiple of 4096 bytes
- Is less than or equal to 65536 bytes
- Starts on a 4096-byte boundary

See the BUFSIZE parameter description for specifying the size of the buffer.

The type of data returned to the buffer depends on which types of data that you request on the TYPE parameter. See *z/OS MVS Programming: Sysplex Services Guide* for more detailed information about how data is returned from this request.

**Note:** You cannot code BUFFER with MODE=ASYNCRESPONSE.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of an area (with a length of BUFSIZE) where the data reads from list entries will be put.

**,BUFLIST=buflist**

Use this output parameter to specify a list of buffers to hold data that is read from list entries. BUFLIST specifies a 128-byte storage area that consists of a list of 0 to 16 buffer addresses.

**The 128-byte storage area must:**

- Consist of 0 to 16 elements.
- Each element must consist of an 8-byte field in which:
  - The left (high-order) 4 bytes are reserved.
  - The right (low-order) 4 bytes contain the address of a buffer.

**The BUFLIST buffers must:**

- Reside in the same address space or same data space.
- Be 4096 bytes.
- Start on a 4096-byte boundary.

**Note:** The buffers do not have to be contiguous in storage. List services treat BUFLIST buffers as a single buffer, even if the buffers are not contiguous.

See the BUFNUM parameter description to specify the number of buffers in the buffer list.

The type of data returned to the buffers depends on which types of data that you request on the TYPE parameter. See *z/OS MVS Programming: Sysplex Services Guide* for more detailed information about how data is returned from this request.

**Note:** You cannot code BUFLIST with MODE=ASYNCRESPONSE.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 128-byte area that contains a list of buffer addresses.

**,BUFNUM=*bufnum***

Use this input parameter to specify the number of buffers in the BUFLIST list. Valid BUFNUM values are 0 - 16. A value of zero indicates that no data is to be read into the buffers.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 1-byte field that contains the number of buffers (0 - 16) in the list (BUFLIST).

**,BUFSIZE=*bufsize***

Use this input parameter to specify the size of the BUFFER area. See the BUFFER parameter description for valid buffer sizes.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a fullword field that contains the size of the buffer (BUFFER) in bytes.

**,BUFSTGKEY=CALLERS\_KEY**

**,BUFSTGKEY=*bufstgkey***

Use this input parameter to specify a storage key that you define and use when referencing the buffers specified by BUFLIST or the buffer that is specified by BUFFER.

If you do not specify BUFSTGKEY, or if you specify BUFSTGKEY=CALLERS\_KEY, all references to one or more buffers are performed by using the caller's PSW key.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of an 8-bit field that contains the storage key in the format B'kkkkxxxx', where kkkk is the key and xxxx is ignored.

**,CONTOKEN=*contoken***

Use this input parameter to specify the connect token that was returned by the IXLCONN service. The connect token uniquely identifies your connection to the list structure, and must be specified on each IXLLIST invocation.

The connect token is available in the IXLCONN answer area that is mapped by IXLYCONA.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 16-byte field that contains the connect token.

**,EXTRESTOKEN=NO\_EXTRESTOKEN**

**,EXTRESTOKEN=*extrestoken***

Use this input parameter to specify an extended restart token that can be used to resume processing of a READ\_MULT request that completed prematurely. The extended restart token is returned in the answer area (field LAAEXTRESTOKEN), and should be specified on the next READ\_MULT request to resume processing.

If the request does not complete prematurely, the extended restart token will not be provided.

**Note:**

1. Specifying an extended restart token of all zeros causes list services to treat all of the entries as unprocessed.
2. Do not specify an extended restart token other than the one returned in the answer area or one set to all zeros, because results will be unpredictable.

Requestors that specify IXLCONN ALLOWAUTO=YES must use the extended restart token. Requestors that specify or default to IXLCONN ALLOWAUTO=NO must use the standard restart token (RESTOKEN).

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-byte field that contains the extended restart token.

**,KEYCOMP=NO\_KEYCOMP**

**,KEYCOMP=keycomp**

Use this input parameter to specify a value to be compared to the entry key of each list entry in the designated list.

If the list entry that is to be processed does not have an entry key value that is equal to the specified KEYCOMP value, the entry is not processed. Processing continues with the next entry.

**Note:** The KEYCOMP parameter is valid only for list structures allocated in a coupling facility with CFLEVEL=1 or higher.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the 16-byte field that contains the value to be compared the entry key of the designated list entry.

**,LISTNUM=NO\_LISTNUM**

**,LISTNUM=listnum**

Use this input parameter to specify the number of the list on which the entries to be read must reside.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 4-byte field that contains the number of the list.

**,LOCKCOMP=NO\_LOCKCOMP**

**,LOCKCOMP=lockcomp**

This parameter has slightly different meanings based on the value that is specified for the LOCKOPER parameter. Generally, this input parameter specifies a connection identifier to be verified as the owner of the lock specified by LOCKINDEX. This verification is a prerequisite to the successful completion of the request.

When LOCKCOMP is specified, the completion of the request is conditional on there being no contention for the lock. If contention exists, the request fails.

The connection identifier is available from the IXLCONN answer area, which is mapped which is by IXLYCONA, in field CONACONID.

The effect of LOCKCOMP is based on the LOCKOPER value specified. See the description under LOCKOPER to see how each request is affected by this parameter.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 1-byte field that contains the connection identifier.

**,LOCKINDEX=NO\_LOCKINDEX**

**,LOCKINDEX=lockindex**

Use this input parameter to specify the index of the lock to be operated on as specified by LOCKOPER. The lock indexes begin with zero.

**Note:** You cannot code LOCKINDEX with MODE=ASYNCHNORESPONSE.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 4-byte field that contains the lock index.

**,LOCKMODE=COND**

Use this input parameter specify that the lock operation for the lock specified by LOCKINDEX is performed conditionally; that is, only if there is no contention for the lock. If the specified lock is held by another user, the request is terminated with no change to the structure, and return and reason codes describing the termination are returned to the caller.

**,LOCKOPER=NOTHELD**

**,LOCKOPER=HELD BY**

Use this input parameter to specify the type of operation to be performed on the lock specified by LOCKINDEX.

**LOCKOPER=NOTHELD**

The request is performed only if the lock is not held by any connection. This option also ensures that the lock remains free for the duration of the request. If another connection holds the lock, your task is suspended until the lock is released, or your request fails depending on the LOCKMODE value you specify. See the LOCKMODE description for how to handle possible lock contention.

**LOCKOPER=HELD**

- When LOCKCOMP is not specified, the request is performed only if your connection holds the lock.
- When LOCKCOMP is specified, the request is performed only if the lock is held by the connection that is specified by LOCKCOMP.

**,MF=S****,MF=(L,*mfctrl*)****,MF=(L,*mfctrl*,*mfattr*)****,MF=(L,*mfctrl*,0D)****,MF=(M,*mfctrl*)****,MF=(M,*mfctrl*,COMPLETE)****,MF=(M,*mfctrl*,NOCHECK)****,MF=(E,*mfctrl*)****,MF=(E,*mfctrl*,COMPLETE)****,MF=(E,*mfctrl*,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

**,*mfctrl***

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

**,*mfattr***

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

**,COMPLETE****,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

**COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if SMILE=*var* were an optional parameter and the default is SMILE=NO\_SMILE



then it would not be documented. However, if the default was SMILE=:-), then it would be documented because a value would be the default.

### **NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

**,MODE=SYNCSUSPEND**

**,MODE=SYNCECB**

**,MODE=SYNCEXIT**

**,MODE=SYNCTOKEN**

**,MODE=ASYNCECB**

**,MODE=ASYNCEXIT**

**,MODE=ASYNCTOKEN**

**,MODE=ASYNCSUSPEND**

Use this input parameter to specify:

- Whether the request is to be performed synchronously or asynchronously
- How you want to be notified of request completion

### **MODE=SYNCSUSPEND**

The request is suspended until it can complete synchronously. To use this option, your program must be enabled for I/O and external interrupts.

### **MODE=SYNCECB**

The request processes synchronously if possible. If the request processes asynchronously, the ECB specified by REQECB is posted when the request completes.

### **MODE=SYNCEXIT**

The request processes synchronously if possible. If the request processes asynchronously, your complete exit is given control when the request completes.

### **MODE=SYNCTOKEN**

The request processes synchronously if possible. If the request processes asynchronously, an asynchronous request token is returned in the area specified by REQTOKEN on invocation of the request. Use the returned request token on the IXLFComp macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=SYNCTOKEN is specified.

### **MODE=ASYNCECB**

The request processes asynchronously. The ECB specified by REQECB is posted when the request completes.

### **MODE=ASYNCEXIT**

The request processes asynchronously. Your complete exit is given control when the request completes.

### **MODE=ASYNCTOKEN**

The request processes asynchronously. An asynchronous request token is returned in the area specified by REQTOKEN upon invocation of the request. Use the returned request token on the IXLFComp macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=ASYNCTOKEN is specified.

### **MODE=ASYNCSUSPEND**

The request processes asynchronously. No notification of request completion is provided. The answer area (ANSAREA) fields will not contain valid information when this mode is specified.

**Note:** You cannot code MODE=ASYNCSUSPEND with LOCKINDEX, BUFFER, or BUFLIST.

**,PAGEABLE=YES**

**,PAGEABLE=NO**

Use this input parameter to identify whether the storage areas specified by BUFFER or BUFLIST are in pageable or potentially pageable storage.

**YES**

Specify this option to indicate that the BUFFER or BUFLIST buffers reside in pageable virtual storage. XES performs the required page fixing to fix the buffers in real storage while the cache or list request transfers data to or from the coupling facility.

This includes storage obtained from pageable subpools, disabled reference (DREF) storage, and may include storage that has the potential to become pageable during the processing of a request. (An example is address space storage owned by any swappable address space, for which a PGSER FIX has been successfully processed, but for which the owning address space gets swapped during processing of a cache or list request.) This does not include implicitly non-pageable storage (for example, storage obtained from non-pageable subpools).

The system takes responsibility for managing binds to central storage for the duration of the cache or list request, regardless of what address space owns the storage or whether the storage-owning address space is swappable or nonswappable. The storage can be owned by any address space.

High shared virtual storage areas (above 2GB) may not be used.

**NO**

Specify this option to indicate that the BUFFER or BUFLIST buffers reside in non-pageable virtual storage. XES does not page fix the buffers in real storage.

This includes implicitly non-pageable storage areas (for example, storage obtained from non-pageable subpools), and may include storage that has the potential to become pageable during the processing of a request (An example is address space storage owned by any swappable address space, for which a PGSER FIX has been successfully processed, but for which the owning address space gets swapped-out during processing of a cache or list request.)

The system takes responsibility for managing binds to central storage for the duration of the cache or list request, if and only if the non-pageable storage is owned by either the requester's address space or the connector's address space. If the storage is owned by any other address space, then the invoker is responsible for ensuring that the virtual storage remains non-pageable for the duration of the request (including the case in which the storage is owned by a swappable address space that is swapped during processing of an IXLCACHE or IXLLIST request). Subject to this consideration, the storage can be owned by any address space. See *z/OS MVS Programming: Sysplex Services Guide*.

**,PLISTVER=IMPLIED\_VERSION****,PLISTVER=MAX****,PLISTVER=plistver**

Use this input parameter to specify the version of the macro. See “[Understanding IXLLIST Version Support](#)” on page 965 for a description of the options available with PLISTVER.

**,REQDATA=NO\_REQDATA****,REQDATA=reqdata**

Use this input parameter with MODE=SYNCEXIT or MODE=ASYNCEXIT to pass any data you choose to the complete exit. The exit will get control only if the request is processed asynchronously.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the data to be passed to the complete exit.

**,REQECB=reqecb**

Use this output parameter with either MODE=SYNCECB or MODE=ASYNCECB to specify the address of an ECB, which is to be posted when the request completes if the request was processed asynchronously.

Before coding REQECB, you must ensure that:

- You initialize the ECB before you issue the request.
- The ECB resides in either common storage or the home address space where IXLCONN was issued.
- Any tasks that wait for the ECB to be posted reside in the home address space where IXLCONN was issued.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that contains the address of the ECB to be posted when the request completes. The ECB must be aligned on a fullword boundary.

**,REQID=NO\_REQID**

**,REQID=reqid**

Use this input parameter to specify a user-defined request identifier to be associated with the request. You can specify this request identifier on the IXPURGE macro to cancel a request that has not yet been processed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the user-defined request identifier.

**,REQTOKEN=reqtoken**

Use this output parameter with either MODE=SYNCTOKEN or MODE=ASYNCTOKEN to specify the address of a storage area to receive the request token that is returned when the request will be processed asynchronously. This token, which uniquely identifies the request, must be used as input to the IXLFCOMP macro, which you use to determine if the request has completed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field where the system will put the request token.

**,RESTOKEN=NO\_RESTOKEN**

**,RESTOKEN=restoken**

Use this input parameter to specify a restart token that can be used to resume processing of a READ\_MULT request that completed prematurely because it exceeded the coupling facility model-dependent time-out criteria. The restart token is returned in the answer area (field LAARESTOKEN), and should be specified on the next READ\_MULT request to resume processing with the next entry to be read.

If the request does not exceed the model-dependent time-out criteria, the returned token will not be provided.

**Note:**

1. Specifying an extended restart token of all zeros causes list services to treat all of the entries as unprocessed.
2. Do not specify an extended restart token other than the one returned in the answer area or one set to all zeros, because results will be unpredictable.

Requestors that specify or default to IXLCONN ALLOWAUTO=NO must use the standard restart token. Requestors that specify IXLCONN ALLOWAUTO=YES must use the extended restart token (EXTRESTOKEN).

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-byte field that contains the restart token.

**,RETCODE=retcode**

Use this output parameter to specify a field to contain the return code. (The return code is also returned in register 15.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the return code when the request has completed.

**,RSNCODE=rsncode**

Use this output parameter to specify a field to contain the reason code returned, if applicable. (The reason code is also returned in register 0.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the reason code (if any) when the request has completed.

**,TYPE=ENTDATA**

**,TYPE=ADJDATA**

**,TYPE=ECONTROLS**

Use this input parameter to specify the type of data to be read from each processed entry.

**ENTDATA**

Entry data is read.

**ADJDATA**

Adjunct data is read.

**ECONTROLS**

List entry control data is read.

You can specify one, two, or all three of these data types, and you can code them in any order on the TYPE parameter. When more than one data type is specified, the format of the parameter would be: TYPE=(ENTDATA,ADJDATA,ECONTROLS). The order in which you code the values does not affect the order in which the system returns the data in the BUFFER area or BUFLIST buffers. See [z/OS MVS Programming: Sysplex Services Guide](#) for more information about how the data is arranged in the.

**,VERSCOMP=NO\_VERSCOMP****,VERSCOMP=verscomp**

Use this input parameter to specify a version number to be compared to the version number of each entry to be read. Only those entries with a version that meets the condition specified by the VERSCOMPTYPE parameter are read.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-byte field that contains the version number.

**,VERSCOMPTYPE=EQUAL****,VERSCOMPTYPE=LESSOREQUAL**

Use this input parameter to specify how a list entry version comparison as specified by VERSCOMP is to be performed.

**Note:** The VERSCOMPTYPE parameter is valid only for list structures allocated in a coupling facility with CFLEVEL=1 or higher.

**VERSCOMPTYPE=EQUAL**

The version number for the list entry must be equal to the value specified for VERSCOMP.

**VERSCOMPTYPE=LESSOREQUAL**

The version number for the list entry must be less than or equal to the value specified for VERSCOMP.

## ABEND Codes

---

Abend X'026' (See [z/OS MVS System Codes](#) for more information on this abend.)

## Return and Reason Codes

---

When the system returns control to your program:

- GPR 15 (and RETCODE, if specified) contains a return code.
- If the return code is not zero, GPR 0 (and RSNCODE, if specified) contains a reason code.

**Note:** The return and reason codes will also be put into the answer area (mapped by IXLYLAA), if there is one, when the request completes.

The IXLYCON macro provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

**0**

IXLRETCODEOK

**4**

IXLRETCODEWARNING

**8**

IXLRETCODEPARMERROR

## C

IXLRETCODEENVEERROR

## 10

IXLRETCODECOMPERROR

The following table contains hexadecimal return and reason codes, the equate symbols associated with each reason code, and the meaning and suggested action for each return and reason code.

Table 72. Return and Reason Codes for IXLLIST REQUEST=READ_MULT Macro		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0	None.	<p><b>Meaning:</b> If you specified a MODE value of ASYNCECB, ASYNCEXIT, ASYNCTOKEN, or ASYNCNORESPONSE, the request has been successfully initiated. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed. For any other MODE value, the request has successfully completed.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=ASYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=ASYNCEXIT, the connection's complete exit will be given control when the request has completed.</li> <li>• If you specified MODE=ASYNCTOKEN, use REQTOKEN when calling IXLFCOMP to determine when the request has completed.</li> </ul>
4	xxxx0402	<p><b>Equate Symbol:</b> IXLRSNCODEASYNCH</p> <p><b>Meaning:</b> The request could not be processed synchronously. It will be processed asynchronously. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed.</p> <ul style="list-style-type: none"> <li>• If MODE=SYNCECB was specified, the ECB specified for REQECB will be posted when the request has finished.</li> <li>• If MODE=SYNCEXIT was specified, the connection's complete exit will be given control when the request has finished.</li> <li>• If MODE=SYNCTOKEN was specified, an asynchronous request token was returned in the area specified by REQTOKEN. This request token may be specified on the IXLFCOMP macro to determine when the request has finished.</li> </ul> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=SYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=SYNCEXIT, the connection's complete exit will be given control when the request has completed.</li> <li>• If you specified MODE=SYNCTOKEN, use REQTOKEN when calling IXLFCOMP to determine when the request has completed.</li> </ul>

Table 72. Return and Reason Codes for IXLLIST REQUEST=READ\_MULT Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	xxxx0409	<p><b>Equate Symbol:</b> IXLRNCODETIMEOUT</p> <p><b>Meaning:</b> The request has completed prematurely because it exceeded the coupling facility model-dependent time-out criteria. The following information has been returned in the answer area:</p> <ul style="list-style-type: none"> <li>• The number of successfully processed entries (field LAAREADCNT)</li> <li>• The list entry controls for the first processed entry (field LAARLRMLCTL).</li> <li>• A token for restarting the request (field LAARESTOKEN or LAAESTRESTOKEN)</li> </ul> <p><b>Action:</b> Reissue the request using the restart token (RESTOKEN or EXTRESTOKEN). For more information about premature completion, see <i>z/OS MVS Programming: Sysplex Services Guide</i>.</p>

Table 72. Return and Reason Codes for IXLLIST REQUEST=READ\_MULT Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	xxxx040D	<p><b>Equate Symbol:</b> IXLRSNCODEBADREADADJDATA</p> <p><b>Meaning:</b> Program error. The request specified that adjunct data was to be read, but the storage area specified by ADJAREA is not addressable. All other requested data was retrieved and the following fields in the answer area have been filled in:</p> <ul style="list-style-type: none"> <li>• LAARLRLCTL - The first PROCESSED entry in the list sequence.</li> <li>• LAAREADCNT - The number of entries processed successfully</li> <li>• LAALCTL (if the request completed prematurely as well) - The list entry controls for the first UNPROCESSED entry in the list sequence.</li> </ul> <p><b>Note:</b> Only the adjunct data for the first entry in the list is placed in the ADJAREA. The buffers should contain the adjunct data for the other entries in the list.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the ADJAREA address.</li> <li>• ADJAREA must be addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are calling IXLLIST while disabled, ADJAREA must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLLIST in AR-mode and you specified the ADJAREA address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are calling IXLLIST in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLIST macro.</li> <li>• If LAALCTL is not zeros, the request completed prematurely. See the action suggested for return code X'4' reason code X'xxxx0409'.</li> </ul> <p>Correct the address specified by ADJAREA, and rerun the request asking for adjunct data only.</p>

Table 72. Return and Reason Codes for IXLLIST REQUEST=READ\_MULT Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	xxxx040F	<p><b>Equate Symbol:</b> IXLRSNCODEBUFFERFULL</p> <p><b>Meaning:</b> Program error. The request has completed prematurely because the buffer specified by BUFFER or the buffers specified by BUFLIST are full. The following information has been returned in the answer area:</p> <ul style="list-style-type: none"> <li>• The number of successfully processed entries (field LAAREADCNT).</li> <li>• A restart token (field LAARESTOKEN) or extended restart token (field LAAEXTRESTOKEN).</li> <li>• The list entry controls for the first processed entry in the list sequence (field LAARLRLCTL).</li> </ul> <p><b>Action:</b> Reissue the request, or increase the size of the buffer(s), and rerun your program. You can reissue the request using the restart token (RESTOKEN) or extended restart token (EXTRESTOKEN). If the same buffers are used, the data returned from the original request will be overwritten so be sure to process the returned data first. For more information about premature request completion, see <i>z/OS MVS Programming: Sysplex Services Guide</i>.</p>
4	xxxx0410	<p><b>Equate Symbol:</b> IXLRSNCODELOCKCOND</p> <p><b>Meaning:</b> For a LOCKOPER=HELDDBY request, or a LOCKMODE=COND request, or a request that specified LOCKCOMP, the request could not be completed successfully because the specified lock is not currently held as required. The connection identifier of the lock owner is returned in the answer area (LAACONID field).</p> <p><b>Action:</b> Retry the request, or obtain the lock as required, and retry the request. If you are unable to get the lock, check the ID in the LAACONID field and determine if some recovery is necessary.</p>
4	xxxx0412	<p><b>Equate Symbol:</b> IXLRSNCODELOCKHELDDBYSYS</p> <p><b>Meaning:</b> The lock is not held by any connection, but instead is held by the system. For a request that specified any of the following, the request could not be completed successfully because the specified lock is not currently held as required:</p> <ul style="list-style-type: none"> <li>• LOCKOPER=NOTHELD with either of: <ul style="list-style-type: none"> <li>– LOCKMODE=COND</li> <li>– LOCKCOMP</li> </ul> </li> <li>• LOCKOPER=HELDDBY</li> </ul> <p><b>Action:</b> Retry the request, or obtain the lock as required, and retry the request.</p>



Table 72. Return and Reason Codes for IXLLIST REQUEST=READ\_MULT Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0801	<p><b>Equate Symbol:</b> IXLRSNCODEBADPARMLIST</p> <p><b>Meaning:</b> Program error. The parameter list for this request is not addressable. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Verify that:</p> <ul style="list-style-type: none"> <li>• The parameter list address is uncorrupted.</li> <li>• The parameter list is addressable in the caller's primary address space.</li> <li>• If you are issuing this macro while disabled, the parameter list resides in either page-fixed or DREF storage.</li> <li>• If you are issuing this macro in AR-mode and you specified the parameter list address using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If you are issuing this macro in AR-mode, SYSSTATE ASCENV=AR must be issued before issuing this macro.</li> </ul>
8	xxxx0804	<p><b>Equate Symbol:</b> IXLRSNCODEBADVERSIONNUM (or IXLRSNCODEBADVERSION#)</p> <p><b>Meaning:</b> The version number in the macro parameter list is not compatible with the level of XES currently being used. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that your program did not overlay the parameter list storage.</li> <li>• Verify that your program was assembled with the correct macro library for the release of MVS on which your program is running.</li> </ul>

Table 72. Return and Reason Codes for IXLLIST REQUEST=READ\_MULT Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx080A	<p><b>Equate Symbol:</b> IXLRSNCODEBADCONTOKEN</p> <p><b>Meaning:</b> Program error. The specified connect token (CONTOKEN) was not valid for one of the following reasons:</p> <ol style="list-style-type: none"> <li>1. The user with the connection identifier represented by the token has disconnected from the structure.</li> <li>2. The connector's task (the task that issued IXLCONN) ended.</li> <li>3. The specified token is not the token that was returned from IXLCONN.</li> <li>4. The request was issued from an address space other than the address space in which IXLCONN was issued.</li> <li>5. The connect token was invalidated during rebuild.</li> <li>6. The connect token was invalidated by XES.</li> </ol> <p><b>Note:</b> The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Take the action with the corresponding meaning.</p> <ol style="list-style-type: none"> <li>1. Discontinue use of the structure. Perform recovery and cleanup for the structure and check the protocol for the use of the structure to determine why you did not know of the disconnection.</li> <li>2. Discontinue use of the structure. Perform recovery and cleanup for the structure and check the protocol for the use of the structure to determine why you did not know that the task ended.</li> <li>3. The contoken is returned in the CONACONTOKEN field of the answer area specified on the IXLCONN request.</li> <li>4. Issue your request from the same address space the IXLCONN was issued in.</li> <li>5. Wait for the rebuild to complete, and try again.</li> <li>6. Discontinue use of the structure. Perform recovery and cleanup for the structure.</li> </ol>
8	xxxx0822	<p><b>Equate Symbol:</b> IXLRSNCODEBADREADTYPE</p> <p><b>Meaning:</b> Program error. You specified that either entry data or adjunct data should be returned, but the list structure does not contain the type of data you requested. No data is returned.</p> <p><b>Action:</b> Ensure that you are requesting the type of data you need from the structure and that the structure supports that type of data. Issue IXLMG to get more information about the structure.</p>

Table 72. Return and Reason Codes for IXLLIST REQUEST=READ\_MULT Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0824	<p><b>Equate Symbol:</b> IXLRSNCODEWRONGSTRTYPE</p> <p><b>Meaning:</b> Program error. The connection specified by CONTOKEN is not to a list structure.</p> <p><b>Action:</b> Specify the appropriate CONTOKEN value, and rerun your program. The CONTOKEN is returned in the answer area provided on the IXLCONN invocation. The type of structure built is passed on the TYPE parameter of the IXLCONN macro.</p>
8	xxxx0833	<p><b>Equate Symbol:</b> IXLRSNCODEBADPGBLATTR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list is specified as being pageable (PAGEABLE=YES), but is not.</p> <p><b>Action:</b> Change the buffer area(s) to pageable storage, or specify PAGEABLE=NO. See the PAGEABLE parameter description for specification instructions.</p>
8	xxxx0834	<p><b>Equate Symbol:</b> IXLRSNCODEBADNONPGBLATTR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list, is specified as being nonpageable (PAGEABLE=NO), but is either pageable or not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The type of storage in the buffer area corresponds to the value you specified on the PAGEABLE parameter. See the PAGEABLE parameter description for specification instructions.</li> <li>• The correct buffer address was used.</li> <li>• The buffer area was not previously freed.</li> <li>• If you are calling IXLLIST while disabled, the buffers must reside in either page-fixed or DREF storage.</li> <li>• The buffer areas were allocated in a storage key that matches the key specified by the BUFSTGKEY parameter, or, if BUFSTGKEY is omitted, the caller's PSW key.</li> <li>• If BUFLIST was specified and your program is running in AR-mode: <ul style="list-style-type: none"> <li>– If the BUFFER or BUFLIST parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>– The BUFALET specification is correct.</li> <li>– You specified SYSSTATE ASCENV=AR before issuing the IXLLIST macro.</li> </ul> </li> </ul>

Table 72. Return and Reason Codes for IXLLIST REQUEST=READ\_MULT Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0835	<p><b>Equate Symbol:</b> IXLRSNCODEBADDATAADDR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list, is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The correct buffer address was used.</li> <li>• The buffer area was not previously freed.</li> <li>• The buffer area was allocated in a storage key that matches the key specified using the BUFSTGKEY parameter, or, if BUFSTGKEY is omitted, the caller's PSW key.</li> <li>• If BUFLIST was specified and your program is running in AR mode: <ul style="list-style-type: none"> <li>– The BUFALET specification is correct.</li> <li>– You specified SYSSTATE ASCENV=AR before issuing the macro.</li> </ul> </li> </ul>
8	xxxx0836	<p><b>Equate Symbol:</b> IXLRSNCODEBADREALADDR</p> <p><b>Meaning:</b> Program error. Real storage addresses were provided in the BUFLIST list, but one of the buffers is not addressable in central storage.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that BUFADDRTYPE was specified as you intended.</li> <li>• Ensure that the buffer addresses specified by BUFLIST are valid.</li> </ul>
8	xxxx0838	<p><b>Equate Symbol:</b> IXLRSNCODEBADANSAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by ANSAREA is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The answer area address specified by ANSAREA is valid.</li> <li>• If the caller is running in AR-mode and the ANSAREA parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• The address specified for ANSAREA is addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are calling IXLLIST while disabled, ANSAREA must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLLIST in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLIST macro.</li> </ul>

Table 72. Return and Reason Codes for IXLLIST REQUEST=READ\_MULT Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0839	<p><b>Equate Symbol:</b> IXLRSNCODEBADREQTOKENAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by REQTOKEN is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The request token area specified by REQTOKEN is valid.</li> <li>• If the caller is running in AR-mode and the REQTOKEN parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If you are calling IXLLIST while disabled, REQTOKEN must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLLIST in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLIST macro.</li> </ul>
8	xxxx083D	<p><b>Equate Symbol:</b> IXLRSNCODEBADANSLEN</p> <p><b>Meaning:</b> Program error. The length of the answer area, as specified by ANSLEN, is not sufficient to contain answer area information. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Increase the size of the answer area, and rerun your program. The minimum size of the ANSAREA is in the LAA_LEN field of the IXLYLAA mapping macro.</p>
8	xxxx0846	<p><b>Equate Symbol:</b> IXLRSNCODEBADLOCKINDEX</p> <p><b>Meaning:</b> Program error. The specified LOCKINDEX exceeds the size of the lock table for the structure.</p> <p><b>Action:</b> Correct LOCKINDEX to specify an index that is contained within the lock table. The maximum value for the LOCKINDEX is one less than the value specified by the LOCKENTRIES parameter on the IXLCONN macro.</p>
8	xxxx0847	<p><b>Equate Symbol:</b> IXLRSNCODEBADLISTNUMBER</p> <p><b>Meaning:</b> Program error. The specified LISTNUM value exceeds the number of lists for the structure.</p> <p><b>Action:</b> Correct LISTNUM to specify a list number that exists in the structure. The number of lists allocated for a structure is determined on the LISTHEADERS parameter of the IXLCONN macro. The list numbers go from 0 to n-1, where n is the number of lists specified.</p>

Table 72. Return and Reason Codes for IXLLIST REQUEST=READ\_MULT Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0849	<p><b>Equate Symbol:</b> IXLRSNCODEBADRESTOKEN</p> <p><b>Meaning:</b> Program error. The restart token specified by RESTOKEN is not valid.</p> <p><b>Action:</b> Determine why the restart token cannot be used to restart the request. Possible causes are:</p> <ul style="list-style-type: none"> <li>• The specified token does not correspond to the restart token returned in the answer area of the previous request (field LAARESTOKEN).</li> <li>• The user specified RESTOKEN when EXTRESTOKEN was required.</li> <li>• The user specified EXTRESTOKEN when RESTOKEN was required.</li> </ul> <p>For more information about premature request completion, see <i>z/OS MVS Programming: Sysplex Services Guide</i>.</p>
8	xxxx084B	<p><b>Equate Symbol:</b> IXLRSNCODENOLOCKS</p> <p><b>Meaning:</b> Program error. A locking operation was requested, but the structure does not contain a lock table.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• You are connected to the intended structure.</li> <li>• You intended to perform a locking operation.</li> </ul> <p>The use of locks is determined by the LOCKENTRIES keyword on the IXLCONN macro.</p>
8	xxxx0851	<p><b>Equate Symbol:</b> IXLRSNCODENOSUSPENDISABLE</p> <p><b>Meaning:</b> Program error. The request failed because MODE=SYNCSUSPEND was specified, but the caller is disabled and cannot be suspended.</p> <p><b>Action:</b> Either specify another MODE value, or become enabled (release the CPU lock), then reissue the request.</p>
8	xxxx0854	<p><b>Equate Symbol:</b> IXLRSNCODEBADLOCKCOMP</p> <p><b>Meaning:</b> Program error. The connection identifier specified for LOCKCOMP is not valid.</p> <p><b>Action:</b> The connection identifier is returned in the answer area (mapped by IXLYCONA) when the IXLCONN macro was issued.</p>

Table 72. Return and Reason Codes for IXLLIST REQUEST=READ\_MULT Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0859	<p><b>Equate Symbol:</b> IXLRSNCODEBADLISTAUTH</p> <p><b>Meaning:</b> The list authority specified for AUTHCOMP failed the comparison specified by AUTHCOMPTYPE for the list authority of the specified list. The current list authority (field LAALISTAUTH) and description (field LAALISTDESC) are returned in the answer area.</p> <p><b>Action:</b> None required; however you might want to take some action depending on your application. The list authority is set to binary zeros when the structure is allocated. When IXLLIST is issued with the NEWAUTH keyword, the list authority is changed if the correct comparison list authority value is specified. Check the answer area to determine the list authority value for the list specified. Verify that the correct list number was specified.</p>
8	xxxx0864	<p><b>Equate Symbol:</b> IXLRSNCODEBADBUFSIZE</p> <p><b>Meaning:</b> Program error. The size of the BUFFER area or the buffer areas specified by BUFLIST is not large enough to contain the data being read. No data is returned. The answer area (field LAALCTL) contains the list entry controls for the first entry selected for processing.</p> <p><b>Action:</b> If more space is available, specify a larger buffer size, and reissue the request. Consider using the BUFLIST parameter instead of the BUFFER parameter.</p>
8	xxxx0865	<p><b>Equate Symbol:</b> IXLRSNCODEBADBUFSPEC</p> <p><b>Meaning:</b> Program error. There is an error in the buffer specification</p> <p><b>Action:</b> Check the following:</p> <ul style="list-style-type: none"> <li>• If BUFLIST was specified, check the requirements for BUFLIST and BUFNUM.</li> <li>• If BUFFER was specified, check the requirements for BUFFER and BUFSIZE.</li> <li>• Buffer pointer(s) in BUFLIST.</li> <li>• Buffer boundaries.</li> </ul>

Table 72. Return and Reason Codes for IXLLIST REQUEST=READ\_MULT Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0866	<p><b>Equate Symbol:</b> IXLRSNCODEBADBUFKEY</p> <p><b>Meaning:</b> Program error. The buffer storage key specified by BUFSTGKEY is incorrect or BUFSTGKEY was not specified and the PSW key does not match the key of the buffers.</p> <p>The data cannot be stored in the specified buffer area.</p> <p><b>Action:</b> Check the following:</p> <ul style="list-style-type: none"> <li>• Determine if the key of the storage being used for the buffers is different from the PSW key.</li> <li>• If BUFSTGKEY was specified, verify that the key was put in the correct bits (see the explanation of this parameter for more information).</li> <li>• If you are calling IXLLIST in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLIST macro.</li> </ul>
8	xxxx0867	<p><b>Equate Symbol:</b> IXLRSNCODEBADBUFLIST</p> <p><b>Meaning:</b> Program error. The 128-byte storage area specified by BUFLIST is not addressable.</p> <p><b>Action:</b> Ensure that the address specified by BUFLIST is valid.</p>
8	xxxx0887	<p><b>Equate Symbol:</b> IXLRSNCODEBADEXTRESTOKEN</p> <p><b>Meaning:</b> Program error. The extended restart token specified by EXTRESTOKEN is not valid. The specified token refers to an older instance of the target structure.</p> <p><b>Action:</b> Reinitiate the request with an EXTRESTOKEN value of zero. A system-managed process occurred between the time a request returned the extended restart token and the time the connector tried to continue the request using that token. Discard the results of the initial request and reissue the request. For more information about restarting requests, see <a href="#">z/OS MVS Programming: Sysplex Services Guide</a>.</p>
C	xxxx0C06	<p><b>Equate Symbol:</b> IXLRSNCODENOCNN</p> <p><b>Meaning:</b> Environmental error. This system does not have connectivity to the coupling facility that contains the list structure. Possible reasons for this are:</p> <ul style="list-style-type: none"> <li>• The operator issued VARY PATH,OFFLINE.</li> <li>• The operator issued CONFIG CHP,OFFLINE.</li> <li>• Hardware errors to the coupling facility.</li> <li>• Facility or path failure to the coupling facility.</li> </ul> <p><b>Action:</b> Begin rebuilding the structure on a different coupling facility, or disconnect from the structure.</p>



Table 72. Return and Reason Codes for IXLLIST REQUEST=READ\_MULT Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C13	<p><b>Equate Symbol:</b> IXLRNCDEREQPURGED</p> <p><b>Meaning:</b> Environmental error. The request was purged prior to completion of the request. Possible reasons include:</p> <ul style="list-style-type: none"> <li>• The connector failed.</li> <li>• The connector disconnected.</li> <li>• The requestor failed.</li> <li>• The request was purged by IXLPURGE.</li> <li>• Requests were purged when the connector provided an IXLEERSP response for the Rebuild Stop or Rebuild Cleanup event.</li> <li>• The secondary address space was no longer valid.</li> </ul> <p><b>Action:</b> None if this is expected. Otherwise, determine why the connector failed.</p>
C	xxxx0C14	<p><b>Equate Symbol:</b> IXLRNCDERSTATUSUNKNOWN</p> <p><b>Meaning:</b> Environmental error. The request has completed, but the final disposition of the request cannot be determined. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the validity of your data by comparing the expected results with what is in the coupling facility.</li> <li>• Consider running rebuild processing to move the structure to attempt to protect it from further errors of this kind.</li> </ul>
C	xxxx0C25	<p><b>Equate Symbol:</b> IXLRNCDERSTRFAILURE</p> <p><b>Meaning:</b> Environmental error. The list structure failed prior to completion of the request.</p> <p><b>Action:</b> Either rebuild or disconnect from the structure.</p>
C	xxxx0CA0	<p><b>Equate Symbol:</b> IXLRNCDERQUIESCEDSUSPENDFAIL</p> <p><b>Meaning:</b> Environmental error. The request is failed because the structure is quiesced for a system-managed process and SUSPEND=FAIL is specified on the IXLCONN.</p> <p><b>Action:</b> None, if this is expected.</p>
C	xxxxFFFF	<p><b>Equate Symbol:</b> IXLRNCDERNOTAVAILABLE</p> <p><b>Meaning:</b> Environmental error. XES functions are not available. This could occur because the coupling facility hardware necessary to provide XES functions is not present.</p> <p><b>Action:</b> Re-IPL the system, or follow your particular failure management protocol.</p>

Table 72. Return and Reason Codes for IXLLIST REQUEST=READ\_MULT Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
10	xxxx10xx	<b>Meaning:</b> System error. XES processing failure. The state of the involved structure and the disposition of the request are unpredictable. The answer area (ANSAREA) fields do not contain valid information. <b>Action:</b> Contact the IBM support center.

## Chapter 73. IXLLIST REQUEST=WRITE

### Description

A WRITE request writes entry data and/or adjunct data to an existing or new list entry depending on the value specified for ENTRYTYPE. Entry data is written to the **data entry** segment of a list entry, while the adjunct data is written to the **adjunct data area** segment. Before issuing the request, the data to be written to the entry should be stored as follows:

- Use the BUFFER or BUFLIST parameter to identify the storage containing the data to be placed in the data entry.
- Use the ADJAREA parameter to identify the storage containing the data to be placed in the adjunct area.

**Note:** If the structure supports entry data and BUFFER and BUFLIST are not specified, then the created list entry does not contain any entry data. If the structure supports adjunct data and ADJAREA is not specified, then the created list entry contains binary zeros in the adjunct area.

If you are writing entry data to the structure, you can specify the number of data elements comprising the data entry segment of an existing or new entry by using ELEMNUM.

Additionally, you can perform locking functions with a WRITE request by specifying LOCKOPER. The lock designated by LOCKINDEX will be operated on as specified by LOCKOPER.

With a coupling facility of CFLEVEL=1 or higher, the following additional functions are supported for a WRITE request:

- You can specify that a list entry is to be written only after a successful comparison of the list authority value for the target list with a user-specified value. You specify the list authority comparison requirements using AUTHCOMP and AUTHCOMPTYPE. If the request is successful, you also can update the list authority value to a new value that you specify with NEWAUTH.
- Several options are available for updating the list cursor, based on request completion.
- Version number comparison is enhanced to allow for an additional equal-or-less-than-equal comparison operation.
- The system can assign an entry key value automatically from a list control list key value.

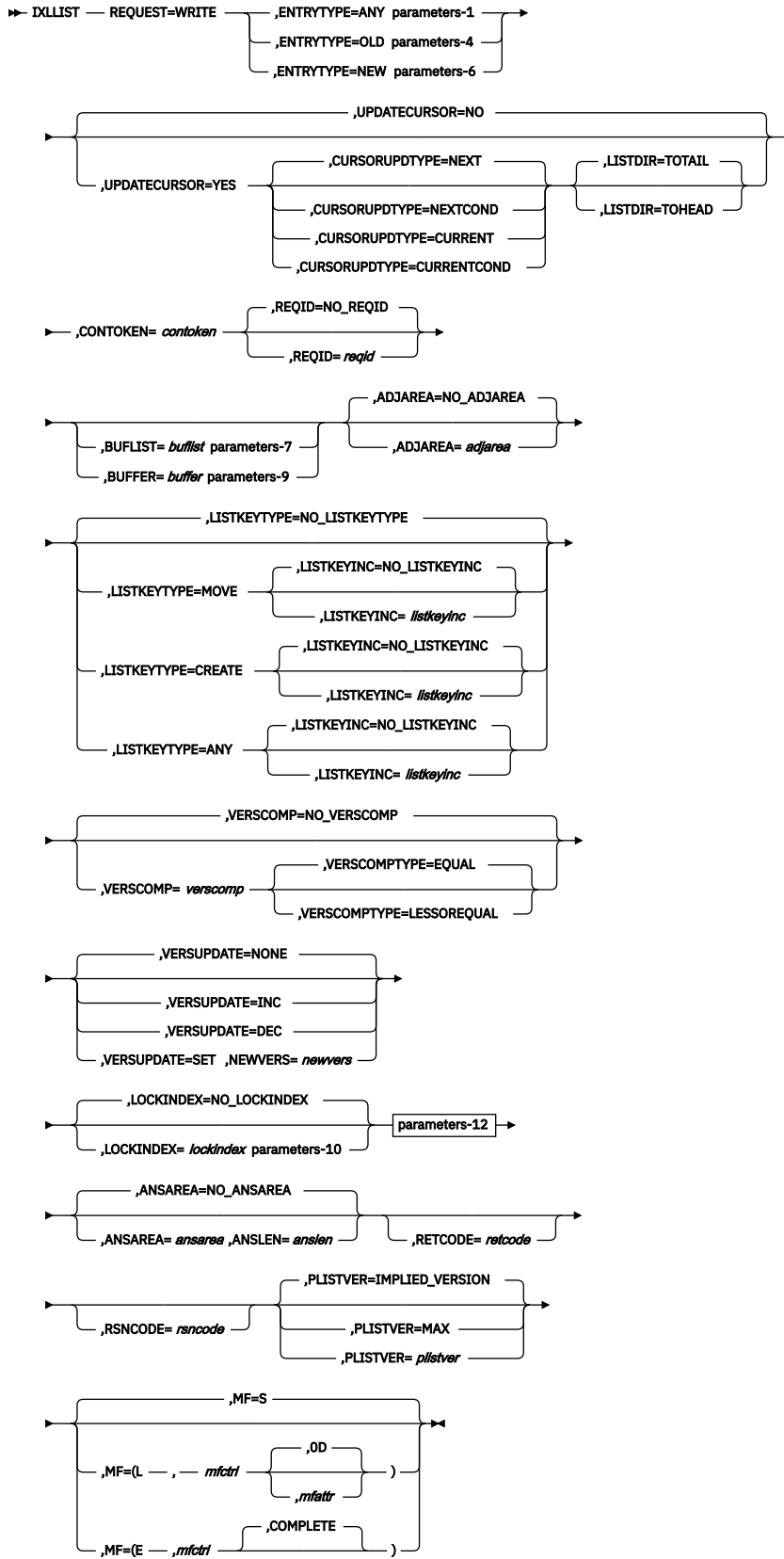
When the request successfully completes, list entry controls, the number of elements or entries residing on the list, and the total number of allocated entries in the structure are returned in the answer area (ANSAREA) if specified (mapped by IXLYLAA).

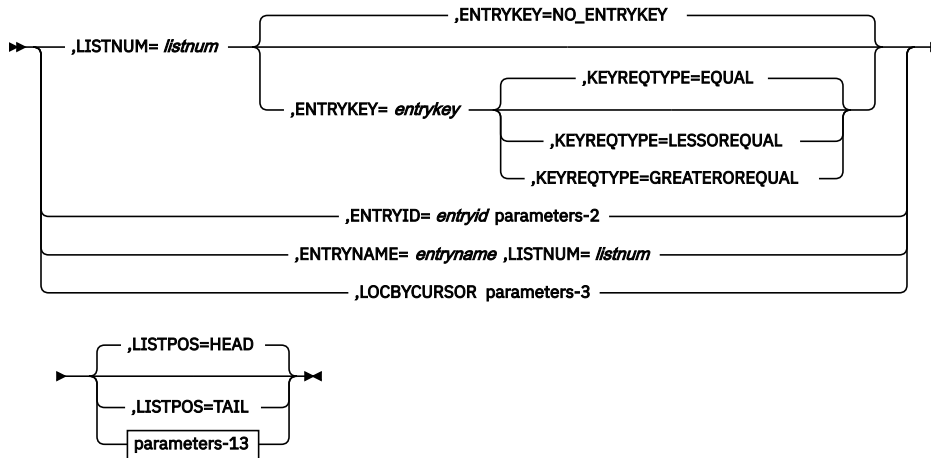
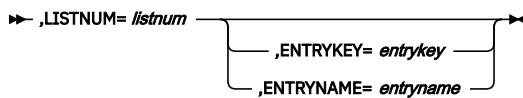
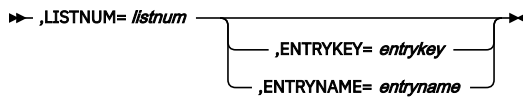
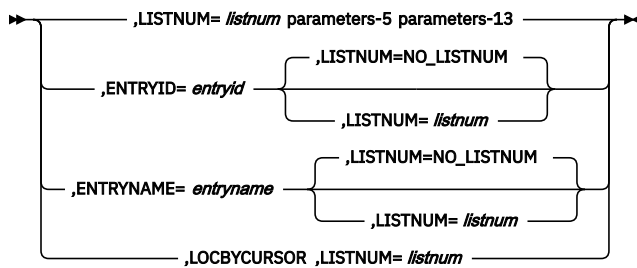
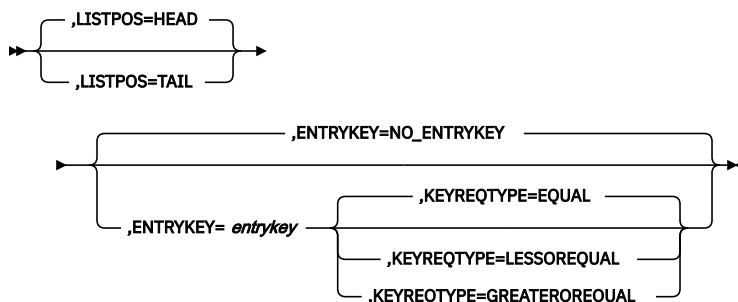
See *z/OS MVS Programming: Sysplex Services Guide* for more information on how to use the IXLLIST REQUEST=WRITE request.

### Syntax Diagram

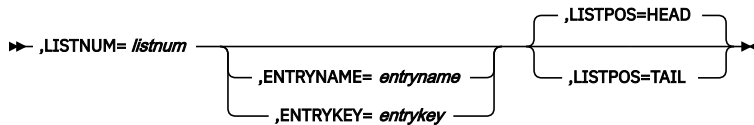
The syntax diagram for IXLLIST REQUEST=WRITE is as follows:

## main diagram



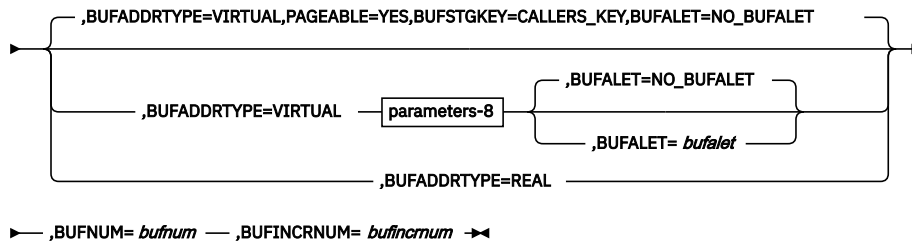
**parameters-1****parameters-2****parameters-3****parameters-4****parameters-5**

## parameters-6



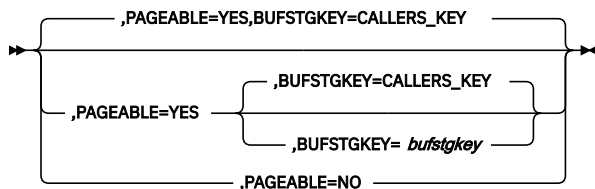
## parameters-7

►► ,ELEMNUM= elemnum →

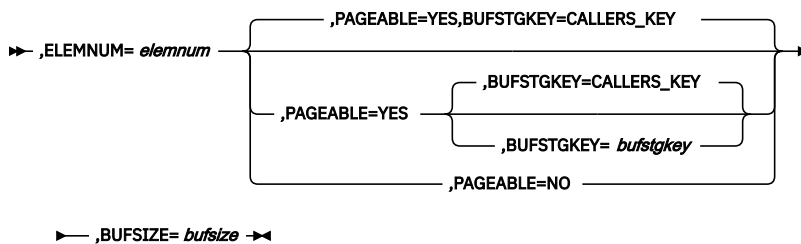


►► ,BUFNUM= bufnum — ,BUFINCRNUM= bufincrnum →◄

## parameters-8

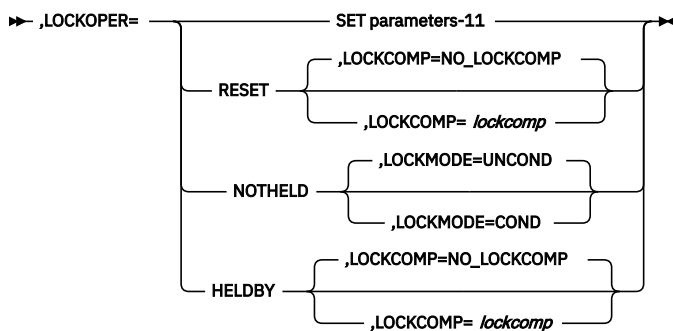


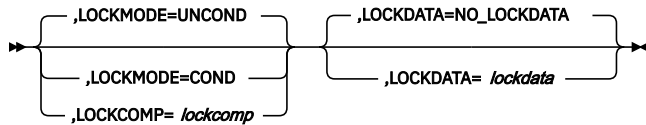
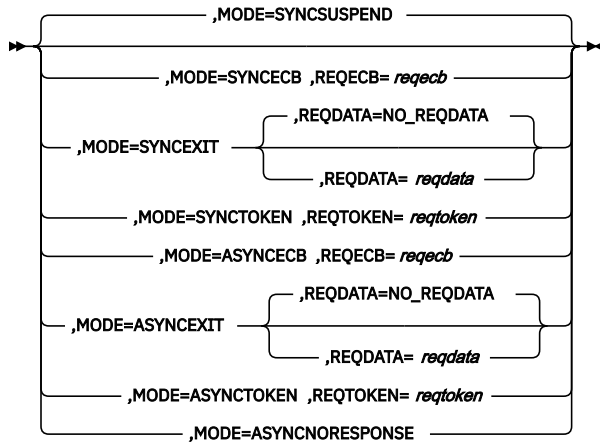
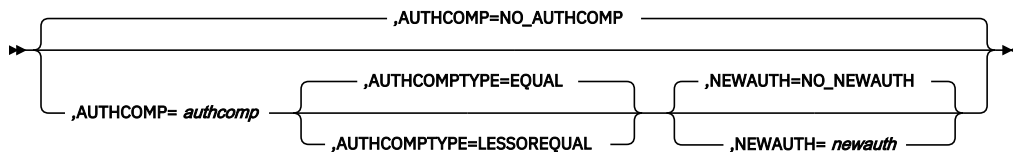
## parameters-9



►► ,BUFSIZE= bufsize →◄

## parameters-10



**parameters-11****parameters-12****parameters-13****Note:**

1. If ENTRYTYPE is not specified, ENTRYTYPE=ANY is the default and you must code the required parameters shown in the | parameters-1 | fragment.
2. If MODE=SYNCTOKEN or MODE=ASYNCTOKEN is specified, ANSAREA=*ansarea* and ANSLEN=*anslen* are required.
3. If MODE=ASYNCRESPONSE is specified, you may not code BUFFER, BUFLIST, or LOCKINDEX.

## Parameter Descriptions

The parameter descriptions for REQUEST=WRITE are listed in alphabetical order. Default values are underlined:

**REQUEST=WRITE**

Use this input parameter to write data to a list entry.

**,ADJAREA=NO\_ADJAREA****,ADJAREA=*adjarea***

Use this input parameter to specify a storage area to contain the adjunct data to be written to the existing or new entry.

Specify ADJAREA only for structures that support adjunct data. (Adjunct areas for a structure are established through the IXLCONN macro.)

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 64-byte area that contains the adjunct data.

**,ANSAREA=NO\_ANSAREA****,ANSAREA=ansarea**

Use this output parameter to specify an answer area to contain information returned from the request. The format of the answer area is described by mapping macro IXLYLAA.

On a successful completion (for a synchronous request: RC=0, for an asynchronous request: ECB is posted, IXLFCOMP indicates completion, or your complete exit ran) of the request, the following information is returned to the answer area:

- List entry controls of the existing or new entry (field LAALCTL).
- The number of list entries or elements residing on the list (field LAALISTCNT).
- The total number of allocated entries in the structure (field LAATOTALCNT).
- The total number of allocated elements in the structure (field LAATOTALELECNT).

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of an area (with a length of ANSLEN) that will contain the information that is returned by the request.

**,ANSLEN=anslen**

Use this input parameter to specify the size of the storage area specified by ANSAREA.

Check the prologue of the IXLYLAA mapping macro for the minimum required size of the answer area, or use the length field (LAA\_LEN) in the LAA.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 2-byte field that contains the size, in bytes, of the answer area.

**,AUTHCOMP=NO\_AUTHCOMP****,AUTHCOMP=authcomp**

Use this input parameter to specify a value to be compared to the list authority value of the list specified by LISTNUM. You must supply the LISTNUM parameter.

If the comparison does not meet the condition specified by the AUTHCOMPTYPE parameter (EQUAL or LESSOREQUAL), the request fails.

**Note:** The AUTHCOMP parameter is valid only for list structures allocated in a coupling facility with CFLEVEL=1 or higher.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12 2 to ) of the 16-byte field that contains the list authority value.

**,AUTHCOMPTYPE=EQUAL****,AUTHCOMPTYPE=LESSOREQUAL**

Use this input parameter specify how the list audit comparison is to be performed.

**EQUAL**

The list authority for the list specified by LISTNUM must be equal to the value specified for AUTHCOMP.

**LESSOREQUAL**

The list authority for the list specified by LISTNUM must be less than or equal to the value specified for AUTHCOMP.

**Note:** The AUTHCOMPTYPE parameter is valid only for list structures allocated in a coupling facility with CFLEVEL=1 or higher.

**,BUFADDRTYPE=VIRTUAL****,BUFADDRTYPE=REAL**

Use this input parameter to specify whether the buffer addresses specified in the BUFLIST list are virtual storage or real storage addresses.

**VIRTUAL**

The buffer addresses are virtual storage addresses. The virtual storage can be pageable or nonpageable. See the PAGEABLE parameter for information about managing storage binds when specifying virtual storage addresses.



**REAL**

The buffer addresses are real storage addresses.

It is the caller's responsibility to manage the binds between the data buffer virtual storage and the real storage addresses provided. The caller must ensure that the data buffer virtual storage remains bound to the real storage addresses provided until the request completes.

**,BUFALET=NO\_BUFALET****,BUFALET=*bufalet***

Use this input parameter to specify an access list entry token (ALET) to be used in referencing all of the buffers specified by BUFLIST.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 4-byte field that contains the ALET.

**,BUFFER=*buffer***

Use this input parameter to specify a buffer area to hold the entry data to be written to the existing or new entry. Only 31-bit addressable (below 2GB) virtual storage areas are supported for the BUFFER specification.

You can define the buffer size to be a total of up to 65536 bytes. Depending on the size you select, the following restrictions apply:

- If you specify a buffer size of less than or equal to 4096 bytes, you must ensure that the buffer:
  - Is 256, 512, 1024, 2048, or 4096 bytes.
  - Starts on a 256-byte boundary.
  - Does not cross a 4096-byte boundary.
- If you specify a buffer size of greater than 4096 bytes, you must ensure that the buffer:
  - Is a multiple of 4096 bytes.
  - Is less than or equal to 65536 bytes.
  - Starts on a 4096-byte boundary.

See the BUFSIZE parameter description for defining the size of the buffer.

**Note:** You cannot code BUFFER with MODE=ASYNCRESPONSE.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an area (with a length of BUFSIZE) that contains the entry data.

**,BUFINCRNUM=*bufincrnum***

Use this input parameter to specify the number of 256-byte segments comprising each buffer in the BUFLIST list.

Valid BUFINCRNUM values are 1, 2, 4, 8, or 16, which correspond to BUFLIST buffer sizes of 256, 512, 1024, 2048, and 4096 bytes respectively.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 1-byte field that contains 1, 2, 4, 8, or 16.

**,BUFLIST=*buflist***

Use this input parameter to specify a list of buffers to hold the entry data to be written to the existing or new entry. BUFLIST specifies a 128-byte storage area that consists of a list of 0 to 16 buffer addresses.

**The 128-byte storage area must:**

- Consist of 0 to 16 elements.
- Each element consists of an 8-byte field in which:
  - The left (high-order) four bytes are reserved and
  - The right (low-order) four bytes contain the address of a buffer.

**The BUFLIST buffers must:**

- Reside in the same address space or data space.
- Be the same size: either 256, 512, 1024, 2048, or 4096 bytes.
- Start on a 256-byte boundary and not cross a 4096-byte boundary.

**Note:** The buffers do not have to be contiguous in storage. List services treats BUFLIST buffers as a single, contiguous buffer, even if the buffers are not contiguous.

See BUFNUM and BUFINCRNUM parameter descriptions for defining the number and size of buffers.

**Note:** You cannot code BUFLIST with MODE=ASYNCRESPONSE.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 128-byte area that contains a list of buffer addresses.

#### **,BUFNUM=*bufnum***

Use this input parameter to specify the number of buffers in the BUFLIST list. Valid BUFNUM values are from 0 to 16. A value of zero indicates that no data is to be written to the list entry.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 1-byte field that contains the number of buffers in the buffer list.

#### **,BUFSIZE=*bufsize***

Use this input parameter to specify the size of the BUFFER area. See the BUFFER parameter description for valid buffer sizes.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a fullword field that contains the size of the buffer (BUFFER) in bytes.

#### **,BUFSTGKEY=CALLERS\_KEY**

#### **,BUFSTGKEY=*bufstgkey***

Use this input parameter to specify a storage key that you define and use when referencing the buffers specified by BUFLIST or the buffer that is specified by BUFFER.

If you do not specify BUFSTGKEY, or if you specify BUFSTGKEY=CALLERS\_KEY, all references to one or more buffers are performed by using the caller's PSW key.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of an 8-bit field that contains the storage key in the format B'kkkkxxxx', where kkkk is the key and xxxx is ignored.

#### **,CONTOKEN=*contoken***

Use this input parameter to specify the connect token that was returned by the IXLCONN service. The connect token uniquely identifies your connection to the list structure, and must be specified on each IXLLIST invocation.

The connect token is available in the IXLCONN answer area that is mapped by IXLYCONA.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 16-byte field that contains the connect token.

#### **,CURSORUPDTYPE=NEXT**

#### **,CURSORUPDTYPE=NEXTCOND**

#### **,CURSORUPDTYPE=CURRENT**

#### **,CURSORUPDTYPE=CURRENTCOND**

Use this input parameter to specify how the list cursor is to be updated when UPDATECURSOR=YES is specified.

**Note:** The NEXTCOND, CURRENT, and CURRENTCOND values are valid only for list structures allocated in a coupling facility with CFLEVEL=1 or higher.

#### **CURSORUPDTYPE=NEXT**

Update the list cursor to point to the list entry before or after the target entry. The direction of the cursor update depends on the value specified for LISTPOS or LISTDIR.

#### **CURSORUPDTYPE=NEXTCOND**

Update the list cursor to point to the list entry before or after the target entry only if the target list entry is moved or deleted.

You set the list cursor direction (so that it will point to either before or after the target entry) with SETCURSOR on a WRITE\_LCONTROLS request.

The list cursor will be reset to binary zeros if either:

- The entry is the last entry on the list and the list cursor direction is set to a head-to-tail direction.
- The entry is the first entry on the list and the list cursor is set to a tail-to-head direction.

#### **CURSORUPDTYPE=CURRENT**

Set the list cursor to point to the list entry processed for the request.

If the list entry is deleted or moved to another list, then the list cursor will be reset to binary zeros.

#### **CURSORUPDTYPE=CURRENTCOND**

Set the list cursor to point to the list entry processed only if the list cursor value currently is zero and the target list entry is not deleted or moved to another list.

If the list entry is deleted or moved to another list, then the list cursor will remain binary zeros.

#### **,ELEMNUM=*elemnum***

Use this input parameter to specify the number of elements to be allocated to the existing or new data entry. Valid ELEMNUM values are from zero to the MAXELEMNUM value that was specified on the IXLCONN macro.

Considerations for specifying ELEMNUM are:

- If this request creates a new entry, the number of elements is set to the ELEMNUM value.
- If the entry exists, the number of elements is updated to the ELEMNUM value specified.

**Note:** If the data from the buffers exceeds the amount of space in the elements, the data is truncated. If the data from the buffers is less than the amount of space in the elements, the remaining space is padded with binary zeros.

If you specify an ELEMNUM value of zero:

- No entry data is written to the new entry.
- Any entry data in the existing entry will be deleted.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 1-byte field that contains the number of elements.

#### **,ENTRYID=*entryid***

Use this input parameter to designate the entry identifier of an existing entry to be updated. ENTRYID applies only to locating an existing entry; it does not determine the identifier of a new entry should one be created.

If an entry with this identifier does not exist (and ENTRYTYPE=ANY or ENTRYTYPE=NEW is specified) a new entry is created on the LISTNUM list with a unique entry identifier assigned by list services. The new entry identifier is returned in the answer area in the field LCTLENTYID in LAALCTL. (The entry identifier you specify will be ignored.)

When ENTRYID is specified with ENTRYNAME, ENTRYKEY, or LISTPOS, list services uses only ENTRYID to determine if the entry already exists.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 12-byte field that contains the entry identifier.

#### **,ENTRYKEY=NO ENTRYKEY**

#### **,ENTRYKEY=*entrykey***

Use this input parameter to either:

- Designate the entry key of an existing entry to be updated.
- Assign an entry key of your choice to a new entry.

The existing entry must reside on the LISTNUM list; the new entry will be created on the LISTNUM list.

Whether the specified entry key designates an existing entry or is assigned to a new entry depends on what ENTRYTYPE value you specify and whether the entry already exists.

If there is a sublist of entries on the list all with the same key as the ENTRYKEY key (or that satisfies the KEYREQTYPE criteria, if specified), the LISTPOS parameter determines which entry in the sublist (the HEAD or TAIL entry) is updated, or, if an entry is created, at which end of the sublist (the HEAD or TAIL) the new entry is positioned.

If this request creates a new entry, but ENTRYKEY is not specified (and the structure supports keyed entries), list services assigns a default key value to the new entry as follows:

- If LISTPOS=HEAD, the new list entry key is set to all binary zeros.
- If LISTPOS=TAIL, the new list entry key is set to all binary ones.

Specify ENTRYKEY only for structures that support keyed entries.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-byte field that contains the entry key.

#### **,ENTRYNAME=entryname**

Use this input parameter to either:

- Designate the entry name of an existing entry to be updated.
- Assign an entry name to a new entry to be created.

The new entry will be created on the LISTNUM list. The existing entry can reside on any list.

Specify ENTRYNAME only for structures that support named entries. Each entry name is unique within the structure.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-byte field that contains the entry name.

#### **,ENTRYTYPE=ANY**

#### **,ENTRYTYPE=OLD**

#### **,ENTRYTYPE=NEW**

Use this input parameter to specify whether you want to create a new entry, update an existing entry, or either.

##### **ANY**

Either update an existing entry or create a new entry. If the designated entry exists, its data is replaced with the new data. If the designated entry does not exist, one will be created with the new data.

##### **OLD**

Update an existing entry. The designated entry must already exist, or the request fails.

##### **NEW**

Create a new entry.

- If you specify ENTRYNAME, a new entry is created only if the entry does not already exist. If the entry exists, the request fails.
- If you specify ENTRYKEY, a new entry will be created regardless of whether another entry with the same key already exists on the designated list.
- If you specify LISTPOS without ENTRYNAME or ENTRYKEY, a new entry is created at the specified HEAD or TAIL of the list.

**Note:** When a new entry is created, the ENTRYID is returned in the answer area in the LCTLENTYID field of LAALCTL.

**,KEYREQTYPE=EQUAL****,KEYREQTYPE=LESSOREQUAL****,KEYREQTYPE=GREATEROREQUAL**

Use this input parameter to specify how, if at all, the entry key of the existing entry to be updated can differ from the entry key specified by ENTRYKEY. KEYREQTYPE applies only to locating an existing entry; it does not determine the key of a new entry should one be created.

**EQUAL**

The entry must have a key that equals the ENTRYKEY key.

**LESSOREQUAL**

The entry must have a key that is less than or equal to the ENTRYKEY key.

**GREATEROREQUAL**

The entry must have a key that is greater than or equal to the ENTRYKEY key.

**Note:**

1. When no entries on the list meet the requirements of the KEYREQTYPE, and a new entry cannot be created (ENTRYTYPE=OLD was specified), the request will be failed with a return code X'8' and reason code IXLRSCODENOENTRY, or a new list entry will be allocated, depending on the other options specified (ENTRYTYPE=NEW or ENTRYTYPE=ANY).
2. When LESSOREQUAL or GREATEROREQUAL are specified, if no entries on the list have an entry key value equal to the specified value, but entries exist with an entry key value greater than (if GREATEROREQUAL was specified), or less than (if LESSOREQUAL was specified) the entry key value specified, the entry with an entry key value closest to the value specified will be selected. When multiple entries have the same entry key value, LISTPOS is used to resolve whether the first or last entry with the entry key value is selected.

**,LISTDIR=TOTAIL****,LISTDIR=TOHEAD**

Use this input parameter with UPDATECURSOR to specify the direction in which the list cursor is moved as a result of this request. See the UPDATECURSOR parameter for a full description of LISTDIR.

**,LISTKEYINC=NO LISTKEYINC****,LISTKEYINC=listkeyinc**

Use this input parameter to specify a value to be added to the list key after the entry key is set to the list key value. If the entry key is not set to the list key value, the list key value will not be changed. If the result of adding the value that is specified by LISTKEYINC to the list key value is greater than the maximum list key value, the system fails the request.

**Note:** The LISTKEYINC parameter is valid only for list structures allocated in a coupling facility with CFLEVEL=1 or higher.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12 2 to ) of the fullword field that contains the value to be added to the list key after the entry key is set the list key value.

**LISTKEYTYPE=NOLISTKEY****LISTKEYTYPE=CREATE****LISTKEYTYPE=ANY**

Use thinut parameter to specify when the designated entry key is to be set to the current list key value. You can set the entry key to the current list key value when you create the entry.

(Set the list key and maximum list key values with a WRITE\_LCONTROLS request.)

**Note:** The LISTKEYTYPE parameter is valid only for list structures allocated in a coupling facility with CFLEVEL=1 or higher.

**LISTKEYTYPE=NOLISTKEY**

Do not set the entry key for the target list entry to the current list key value. If the list entry is created and the list structure supports entries with keys, then the other parameters specified (such as ENTRYKEY) will determine the resultant entry key value.

**LISTKEYTYPE=CREATE**

Set the entry key for the designated list entry to the current list key value if the entry is created by this request.

**LISTKEYTYPE=ANY**

Set the entry key for the designated list entry to the current list key value if the entry is created by this request. (This parameter also applies when you MOVE an entry.)

**,LISTNUM=NO\_LISTNUM****,LISTNUM=listnum**

Use this input parameter to specify the number of the list to which the data will be written. Use LISTNUM in the following ways:

- With LISTPOS and/or ENTRYKEY to designate an existing entry to be updated, or a position on the list for a new entry.
- With ENTRYID or ENTRYNAME to:
  - Check if the entry exists on the list (when ENTRYTYPE=ANY) before proceeding with the request.
  - Confirm that the entry exists on the list (when ENTRYTYPE=OLD) before proceeding with the request.
  - Confirm that the entry does not exist on the list (when ENTRYTYPE=NEW) before proceeding with the request.
- With LOCBYCURSOR to designate an existing entry to be updated.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 4-byte field that contains the number of the list.

**,LISTPOS=HEAD****,LISTPOS=TAIL**

Use this input parameter to designate either:

- The position on the LISTNUM list where the new entry is placed (if this request creates a new entry).
- The existing entry that is to be updated (if this request updates an existing entry).

LISTPOS designates the position or entry at the HEAD or the TAIL of a list or sublist:

- If you specify ENTRYKEY to designate the position or entry and the list contains a sublist of one or more entries with the same key (or that satisfies the KEYREQTYPE criteria, if specified), then:
  - LISTPOS=HEAD designates the position or entry at the head of the sublist.
  - LISTPOS=TAIL designates the position or entry at the tail of the sublist.
- If you do not specify ENTRYKEY to designate the position or entry, then:
  - LISTPOS=HEAD designates the position or entry at the head of the list.
  - LISTPOS=TAIL designates the position or entry at the tail of the list.

**,LOCBYCURSOR**

Use this input parameter to designate an existing entry to be updated. The designated entry is the entry which is identified by the list cursor for a particular list (LISTNUM).

Be aware that the list cursor could have been reset to zeros by a previous request that, for instance, deleted or moved the entry to which the list cursor points, or updated the list cursor after the last entry on the list had been processed. See [z/OS MVS Programming: Sysplex Services Guide](#) for more information about using the list cursor.

**,LOCKCOMP=NO\_LOCKCOMP****,LOCKCOMP=lockcomp**

This parameter has slightly different meanings based on the value that is specified for the LOCKOPER parameter. Generally, this input parameter specifies a connection identifier to be verified as the owner of the lock specified by LOCKINDEX. This verification is a prerequisite to the successful completion of the request.

When LOCKCOMP is specified, the completion of the request is conditional on there being no contention for the lock. If contention exists, the request fails .

The connection identifier is available from the IXLCONN answer area, which is mapped which is by IXLYCONA, in field CONACONID.

The effect of LOCKCOMP is based on the LOCKOPER value specified. See the description under LOCKOPER to see how each request is affected by this parameter.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 1-byte field that contains the connection identifier.

**,LOCKDATA=NO LOCKDATA**

**,LOCKDATA=lockdata**

Use this input parameter to specify information that is to be passed to your notify exit when another user requests the LOCKINDEX lock after you have obtained the lock by using LOCKOPER=SET. This user-defined information will be passed to your notify exit when the other user issues a request specifying either one of the following:

- LOCKOPER=SET
- LOCKOPER=NOTHELD with LOCKMODE=UNCOND

**To Code:** Specify the RS-type name or address (by using a register 2 - 12 2 to ) of an 8-byte field that contains the user-defined information.

**,LOCKINDEX=NO LOCKINDEX**

**,LOCKINDEX=lockindex**

Use this input parameter to specify the index of the lock to be operated on as specified by LOCKOPER. The lock indexes begin with zero.

**Note:** You cannot code LOCKINDEX with MODE=ASYNCSNORESPONSE.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 4-byte field that contains the lock index.

**,LOCKMODE=UNCOND**

**,LOCKMODE=COND**

Use this input parameter to specify how contention for the lock that is specified by LOCKINDEX should be handled if your request causes lock contention.

**UNCOND**

If the specified lock is held by another user, your request is either:

- Suspended until the lock becomes available (if MODE=SYNCSUSPEND is specified).
- Performed asynchronously with notification to you when the request completes (if any MODE value other than SYNCSUSPEND is specified).

**COND**

The lock operation is performed conditionally; that is, only if there is no contention for the lock. If the specified lock is held by another user, the request is terminated with no change to the structure, and return and reason codes describing the termination are returned to the caller.

**,LOCKOPER=SET**

**,LOCKOPER=RESET**

**,LOCKOPER=NOTHELD**

**,LOCKOPER=HELDDBY**

Use this input parameter to specify the type of operation to be performed on the lock specified by LOCKINDEX.

**SET**

Set the lock.

- When LOCKCOMP is not specified, your connection gets the lock, providing no other connection holds the lock. If another connection holds the lock, the request either continues once the lock is released or fails, depending on the LOCKMODE value you specify.

- When LOCKCOMP is specified, if the connection specified by LOCKCOMP holds the lock, the lock is transferred to your connection.

**RESET**

Reset the lock.

- When LOCKCOMP is not specified, the lock is released if it is held by your connection.
- When LOCKCOMP is specified, the lock is released if it is held by the connection that is specified by LOCKCOMP.

**NOTHELD**

The request is performed only if the lock is not held by any connection. This option also ensures that the lock remains free for the duration of the request. If another connection holds the lock, your task is suspended until the lock is released or your request fails, depending on the LOCKMODE value you specify. See the LOCKMODE description for how to handle possible lock contention.

**HELDDBY**

Processing is determined by which connector is holding the lock.

- When LOCKCOMP is not specified, the request is performed only if your connection holds the lock.
- When LOCKCOMP is specified, the request is performed only if the lock is held by the connection that is specified by LOCKCOMP.

**,MF=S**

**,MF=(L,mfctrl)**

**,MF=(L,mfctrl,mfattr)**

**,MF=(L,mfctrl,0D)**

**,MF=(M,mfctrl)**

**,MF=(M,mfctrl,COMPLETE)**

**,MF=(M,mfctrl,NOCHECK)**

**,MF=(E,mfctrl)**

**,MF=(E,mfctrl,COMPLETE)**

**,MF=(E,mfctrl,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

**,mfctrl**

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

**,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.



**,COMPLETE****,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

**COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if `SMILE=var` were an optional parameter and the default is `SMILE=NO_SMILE` then it would not be documented. However, if the default was `SMILE=-)`, then it would be documented because a value would be the default.

**NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

**,MODE=SYNCSUSPEND****,MODE=SYNCECB****,MODE=SYNCEXIT****,MODE=SYNCTOKEN****,MODE=ASYNCECB****,MODE=ASYNCEXIT****,MODE=ASYNCTOKEN****,MODE=ASYNCSUSPEND**

Use this input parameter to specify:

- Whether the request is to be performed synchronously or asynchronously
- How you want to be notified of request completion

**MODE=SYNCSUSPEND**

The request is suspended until it can complete synchronously. To use this option, your program must be enabled for I/O and external interrupts.

**MODE=SYNCECB**

The request processes synchronously if possible. If the request processes asynchronously, the ECB specified by REQECB is posted when the request completes.

**MODE=SYNCEXIT**

The request processes synchronously if possible. If the request processes asynchronously, your complete exit is given control when the request completes.

**MODE=SYNCTOKEN**

The request processes synchronously if possible. If the request processes asynchronously, an asynchronous request token is returned in the area specified by REQTOKEN on invocation of the request. Use the returned request token on the IXLFCOMP macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=SYNCTOKEN is specified.

**MODE=ASYNCECB**

The request processes asynchronously. The ECB specified by REQECB is posted when the request completes.

**MODE=ASYNCEXIT**

The request processes asynchronously. Your complete exit is given control when the request completes.

**MODE=ASYNCTOKEN**

The request processes asynchronously. An asynchronous request token is returned in the area specified by REQTOKEN upon invocation of the request. Use the returned request token on the IXLFCOMP macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=ASYNCTOKEN is specified.

#### **MODE=ASYNCRESPONSE**

The request processes asynchronously. No notification of request completion is provided. The answer area (ANSAREA) fields will not contain valid information when this mode is specified.

**Note:** You cannot code MODE=ASYNCRESPONSE with LOCKINDEX, BUFFER, or BUFLIST.

#### **,NEWAUTH=NO\_NEWAUTH**

##### ***,NEWAUTH=newauth***

Use this input parameter to specify a list authority value for the list specified by LISTNUM. If NEWAUTH is omitted, the list authority for the designated list is unchanged.

When the structure is allocated, the authority value for each list is initialized to binary zeros.

**Note:** The NEWAUTH parameter is valid only for list structures allocated in a coupling facility with CFLEVEL=1 or higher, except on WRITE\_LCONTROLS requests.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12 to ) of the 16-byte field that contains the list authority value.

#### **,NEWVERS=newvers**

Use this input parameter with VERSUPDATE=SET to specify a version number to either replace the version number of the existing entry, or initialize the version number of a new entry.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-byte field that contains the entry version number.

#### **,PAGEABLE=YES**

##### **,PAGEABLE=NO**

Use this input parameter to identify whether the storage areas specified by BUFFER or BUFLIST are in pageable or potentially pageable storage.

##### **YES**

Specify this option to indicate that the BUFFER or BUFLIST buffers reside in pageable virtual storage. XES performs the required page fixing to fix the buffers in real storage while the cache or list request transfers data to or from the coupling facility.

This includes storage obtained from pageable subpools, disabled reference (DREF) storage, and may include storage that has the potential to become pageable during the processing of a request. (An example is address space storage owned by any swappable address space, for which a PGSER FIX has been successfully processed, but for which the owning address space gets swapped during processing of a cache or list request.) This does not include implicitly non-pageable storage (for example, storage obtained from non-pageable subpools).

The system takes responsibility for managing binds to central storage for the duration of the cache or list request, regardless of what address space owns the storage or whether the storage-owning address space is swappable or nonswappable. The storage can be owned by any address space.

High shared virtual storage areas (above 2GB) may not be used.

##### **NO**

Specify this option to indicate that the BUFFER or BUFLIST buffers reside in non-pageable virtual storage. XES does not page fix the buffers in real storage.

This includes implicitly non-pageable storage areas (for example, storage obtained from non-pageable subpools), and may include storage that has the potential to become pageable during the processing of a request (An example is address space storage owned by any swappable address space, for which a PGSER FIX has been successfully processed, but for which the owning address space gets swapped-out during processing of a cache or list request.)

The system takes responsibility for managing binds to central storage for the duration of the cache or list request, if and only if the non-pageable storage is owned by either the requester's address space or the connector's address space. If the storage is owned by any other address space, then the invoker is responsible for ensuring that the virtual storage remains non-pageable for the duration of the request (including the case in which the storage is owned by a swappable

address space that is swapped during processing of an IXLCACHE or IXLLIST request). Subject to this consideration, the storage can be owned by any address space. See *z/OS MVS Programming: Sysplex Services Guide*.

**,PLISTVER=IMPLIED\_VERSION**

**,PLISTVER=MAX**

**,PLISTVER=plistver**

Use this input parameter to specify the version of the macro. See “Understanding IXLLIST Version Support” on page 965 for a description of the options available with PLISTVER.

**,REQDATA=NO\_REQDATA**

**,REQDATA=reqdata**

Use this input parameter with MODE=SYNCEXIT or MODE=ASYNCEXIT to pass any data you choose to the complete exit. The exit will get control only if the request is processed asynchronously.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the data to be passed to the complete exit.

**,REQECB=reqecb**

Use this output parameter with either MODE=SYNCECB or MODE=ASYNCECB to specify the address of an ECB, which is to be posted when the request completes if the request was processed asynchronously.

Before coding REQECB, you must ensure that:

- You initialize the ECB before you issue the request.
- The ECB resides in either common storage or the home address space where IXLCONN was issued.
- Any tasks that wait for the ECB to be posted reside in the home address space where IXLCONN was issued.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that contains the address of the ECB to be posted when the request completes. The ECB must be aligned on a fullword boundary.

**,REQID=NO\_REQID**

**,REQID=reqid**

Use this input parameter to specify a user-defined request identifier to be associated with the request. You can specify this request identifier on the IXPURGE macro to cancel a request that has not yet been processed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the user-defined request identifier.

**,REQTOKEN=reqtoken**

Use this output parameter with either MODE=SYNCTOKEN or MODE=ASYNCTOKEN to specify the address of a storage area to receive the request token that is returned when the request will be processed asynchronously. This token, which uniquely identifies the request, must be used as input to the IXLFComp macro, which you use to determine if the request has completed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field where the system will put the request token.

**,RETCODE=retcode**

Use this output parameter to specify a field to contain the return code. (The return code is also returned in register 15.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the return code when the request has completed.

**,RSNCODE=rsncode**

Use this output parameter to specify a field to contain the reason code returned, if applicable. (The reason code is also returned in register 0.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the reason code (if any) when the request has completed.

**,UPDATECURSOR=NO****,UPDATECURSOR=YES**

Use this input parameter to specify whether the list cursor, for the list specified by LISTNUM, is to be updated to point to another entry on the list.

**NO**

The list cursor is not updated.

**YES**

The list cursor is updated according to the specification of the CURSORUPDATE keyword.

The list cursor is set to binary zeros if its new position would be before the first entry or after the last entry in the list.

**,VERSCOMP=NO\_VERSCOMP****,VERSCOMP=verscomp**

Use this input parameter to specify a version number to be compared to the version number of the existing entry to be updated. If this request creates a new entry, the VERSCOMP specification is ignored.

The entry is updated only if its version number meets the condition specified by the VERSCOMPTYPE parameter.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-byte field that contains the version number.

**,VERSCOMPTYPE=EQUAL****,VERSCOMPTYPE=LESSOREQUAL**

Use this input parameter to specify how a list entry version comparison as specified by VERSCOMP is to be performed.

**Note:** The VERSCOMPTYPE parameter is valid only for list structures allocated in a coupling facility with CFLEVEL=1 or higher.

**VERSCOMPTYPE=EQUAL**

The version number for the list entry must be equal to the value specified for VERSCOMP.

**VERSCOMPTYPE=LESSOREQUAL**

The version number for the list entry must be less than or equal to the value specified for VERSCOMP.

**,VERSUPDATE=NONE****,VERSUPDATE=INC****,VERSUPDATE=DEC****,VERSUPDATE=SET**

Use this input parameter to specify how the version number of the existing entry will be updated, or, for those cases where a new entry is created, initialized.

**VERSUPDATE=NONE**

The existing entry's version number is not updated. The new entry's version number is initialized to binary zeros.

**VERSUPDATE=INC**

The existing entry's version number is increased by one. The new entry's version number is initialized to binary zeros, except for the low-order bit, which is set to one.

**VERSUPDATE=DEC**

The existing entry's version number is decreased by one. The new entry's version number is initialized to all binary ones.

**VERSUPDATE=SET**

Both the existing and the new entry's version number is set to the value specified by NEWVERS.

## ABEND Codes

---

Abend X'026' (See [z/OS MVS System Codes](#) for more information on this abend.)

## Return and Reason Codes

When the system returns control to your program:

- GPR 15 (and RETCODE, if specified) contains a return code.
- If the return code is not zero, GPR 0 (and RSNCODE, if specified) contains a reason code.

**Note:** The return and reason codes will also be put into the answer area (mapped by IXLYLAA) if there is one, when the request completes.

The IXLYCON macro provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

<b>0</b>	IXLRETCODEOK
<b>4</b>	IXLRETCODEWARNING
<b>8</b>	IXLRETCODEPARMERROR
<b>C</b>	IXLRETCODEENVERROR
<b>10</b>	IXLRETCODECOMPERROR

The following table contains hexadecimal return and reason codes, the equate symbols associated with each reason code, and the meaning and suggested action for each return and reason code.

Table 73. Return and Reason Codes for IXLLIST REQUEST=WRITE Macro		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0	None.	<p><b>Meaning:</b> If you specified a MODE value of ASYNCECB, ASYNCEXIT, ASYNCTOKEN, or ASYNCNORESPONSE, the request has been successfully initiated. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed. For any other MODE value, the request has successfully completed.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=ASYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=ASYNCEXIT, the connection's complete exit will be given control when the request has completed.</li> <li>• If you specified MODE=ASYNCTOKEN, use REQTOKEN when calling IXLFCOMP to determine when the request has completed.</li> </ul>

Table 73. Return and Reason Codes for IXLLIST REQUEST=WRITE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	xxxx0402	<p><b>Equate Symbol:</b> IXLRSNCODEASYNCH</p> <p><b>Meaning:</b> The request could not be processed synchronously. It will be processed asynchronously. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed.</p> <ul style="list-style-type: none"> <li>• If MODE=SYNCECB was specified, the ECB specified for REQECB will be posted when the request has finished.</li> <li>• If MODE=SYNCEXIT was specified, the connection's complete exit will be given control when the request has finished.</li> <li>• If MODE=SYNCTOKEN was specified, an asynchronous request token was returned in the area specified by REQTOKEN. This request token may be specified on the IXLFCOMP macro to determine when the request has finished.</li> </ul> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=SYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=SYNCEXIT, the connection's complete exit will be given control when the request has completed.</li> <li>• If you specified MODE=SYNCTOKEN, use REQTOKEN when calling IXLFCOMP to determine when the request has completed.</li> </ul>
4	xxxx0410	<p><b>Equate Symbol:</b> IXLRSNCODELOCKCOND</p> <p><b>Meaning:</b> For a LOCKOPER=HELDDBY request, or a LOCKMODE=COND request, or a request that specified LOCKCOMP, the request could not be completed successfully because the specified lock is not currently held as required. The connection identifier of the lock owner is returned in the answer area (LAACONID field).</p> <p><b>Action:</b> Retry the request, or obtain the lock as required, and retry the request. If you are unable to get the lock, check the ID in the LAACONID field and determine if some recovery is necessary.</p>
4	xxxx0412	<p><b>Equate Symbol:</b> IXLRSNCODELOCKHELDDBYSYS</p> <p><b>Meaning:</b> The lock is not held by any connection, but instead is held by the system. For a request that specified any of the following, the request could not be completed successfully because the specified lock is not currently held as required:</p> <ul style="list-style-type: none"> <li>• LOCKOPER=SET or LOCKOPER=NOTHELD with either of: <ul style="list-style-type: none"> <li>– LOCKMODE=COND</li> <li>– LOCKCOMP</li> </ul> </li> <li>• LOCKOPER=HELDDBY</li> </ul> <p><b>Action:</b> Retry the request, or obtain the lock as required, and retry the request.</p>

Table 73. Return and Reason Codes for IXLLIST REQUEST=WRITE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0801	<p><b>Equate Symbol:</b> IXLRSNCODEBADPARMLIST</p> <p><b>Meaning:</b> Program error. The parameter list for this request is not addressable. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Verify that:</p> <ul style="list-style-type: none"> <li>• The parameter list address is uncorrupted.</li> <li>• The parameter list is addressable in the caller's primary address space.</li> <li>• If you are issuing this macro while disabled, the parameter list resides in either page-fixed or DREF storage.</li> <li>• If you are issuing this macro in AR-mode and you specified the parameter list address using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If you are issuing this macro in AR-mode, SYSSTATE ASCENV=AR must be issued before issuing this macro.</li> </ul>
8	xxxx0804	<p><b>Equate Symbol:</b> IXLRSNCODEBADVERSIONNUM (or IXLRSNCODEBADVERSION#)</p> <p><b>Meaning:</b> The version number in the macro parameter list is not compatible with the level of XES currently being used. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that your program did not overlay the parameter list storage.</li> <li>• Verify that your program was assembled with the correct macro library for the release of MVS on which your program is running.</li> </ul>

Table 73. Return and Reason Codes for IXLLIST REQUEST=WRITE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx080A	<p><b>Equate Symbol:</b> IXLRSNCODEBADCONTOKEN</p> <p><b>Meaning:</b> Program error. The specified connect token (CONTOKEN) was not valid for one of the following reasons:</p> <ol style="list-style-type: none"> <li>1. The user with the connection identifier represented by the token has disconnected from the structure.</li> <li>2. The connector's task (the task that issued IXLCONN) ended.</li> <li>3. The specified token is not the token that was returned from IXLCONN.</li> <li>4. The request was issued from an address space other than the address space in which IXLCONN was issued.</li> <li>5. The connect token was invalidated during rebuild.</li> <li>6. The connect token was invalidated by XES.</li> </ol> <p><b>Note:</b> The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Take the action with the corresponding meaning.</p> <ol style="list-style-type: none"> <li>1. Discontinue use of the structure. Perform recovery and cleanup for the structure and check the protocol for the use of the structure to determine why you did not know of the disconnection.</li> <li>2. Discontinue use of the structure. Perform recovery and cleanup for the structure and check the protocol for the use of the structure to determine why you did not know that the task ended.</li> <li>3. The contoken is returned in the CONACONTOKEN field of the answer area specified on the IXLCONN request.</li> <li>4. Issue your request from the same address space the IXLCONN was issued in.</li> <li>5. Wait for the rebuild to complete, and try again.</li> <li>6. Discontinue use of the structure. Perform recovery and cleanup for the structure.</li> </ol>
8	xxxx0824	<p><b>Equate Symbol:</b> IXLRSNCODEWRONGSTRTYPE</p> <p><b>Meaning:</b> Program error. The connection specified by CONTOKEN is not to a list structure.</p> <p><b>Action:</b> Specify the appropriate CONTOKEN value, and rerun your program. The CONTOKEN is returned in the answer area provided on the IXLCONN invocation. The type of structure built is passed on the TYPE parameter of the IXLCONN macro.</p>



Table 73. Return and Reason Codes for IXLLIST REQUEST=WRITE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0825	<p><b>Equate Symbol:</b> IXLRSNCODENOENTRY</p> <p><b>Meaning:</b> Program error. The designated list entry does not exist; therefore, no entries were processed.</p> <p><b>Action:</b> None necessary. However, if this return code and reason code are unexpected, you should check the way the list entry was created. Examine the parameters specified for locating the list entry on the invocation of this macro.</p>
8	xxxx0833	<p><b>Equate Symbol:</b> IXLRSNCODEBADPGBLATTR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list is specified as being pageable (PAGEABLE=YES), but is not.</p> <p><b>Action:</b> Change the buffer area(s) to pageable storage, or specify PAGEABLE=NO. See the PAGEABLE parameter description for specification instructions.</p>
8	xxxx0834	<p><b>Equate Symbol:</b> IXLRSNCODEBADNONPGBLATTR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list, is specified as being nonpageable (PAGEABLE=NO), but is either pageable or not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The type of storage in the buffer area corresponds to the value you specified on the PAGEABLE parameter. See the PAGEABLE parameter description for specification instructions.</li> <li>• The correct buffer address was used.</li> <li>• The buffer area was not previously freed.</li> <li>• If you are calling IXLLIST while disabled, the buffers must reside in either page-fixed or DREF storage.</li> <li>• The buffer areas were allocated in a storage key that matches the key specified by the BUFSTGKEY parameter, or, if BUFSTGKEY is omitted, the caller's PSW key.</li> <li>• If BUFLIST was specified and your program is running in AR-mode: <ul style="list-style-type: none"> <li>– If the BUFFER or BUFLIST parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>– The BUFALET specification is correct.</li> <li>– You specified SYSSTATE ASCENV=AR before issuing the IXLLIST macro.</li> </ul> </li> </ul>

Table 73. Return and Reason Codes for IXLLIST REQUEST=WRITE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0835	<p><b>Equate Symbol:</b> IXLRSNCODEBADDATAADDR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list, is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The correct buffer address was used.</li> <li>• The buffer area was not previously freed.</li> <li>• The buffer area was allocated in a storage key that matches the key specified using the BUFSTGKEY parameter, or, if BUFSTGKEY is omitted, the caller's PSW key.</li> <li>• If BUFLIST was specified and your program is running in AR mode: <ul style="list-style-type: none"> <li>– The BUFALET specification is correct.</li> <li>– You specified SYSSTATE ASCENV=AR before issuing the macro.</li> </ul> </li> </ul>
8	xxxx0836	<p><b>Equate Symbol:</b> IXLRSNCODEBADREALADDR</p> <p><b>Meaning:</b> Program error. Real storage addresses were provided in the BUFLIST list, but one of the buffers is not addressable in central storage.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that BUFADDRTYPE was specified as you intended.</li> <li>• Ensure that the buffer addresses specified by BUFLIST are valid.</li> </ul>
8	xxxx0837	<p><b>Equate Symbol:</b> IXLRSNCODEBADWRITEADJDATA</p> <p><b>Meaning:</b> Program error. The request specified that adjunct data was to be written, but the area specified by ADJAREA is not addressable. There was no change to the structure.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the ADJAREA address.</li> <li>• ADJAREA must be addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are calling IXLLIST while disabled, ADJAREA must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLLIST in AR-mode and you specified the ADJAREA address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are calling IXLLIST in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLIST macro.</li> </ul>

Table 73. Return and Reason Codes for IXLLIST REQUEST=WRITE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0838	<p><b>Equate Symbol:</b> IXLRSNCODEBADANSAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by ANSAREA is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The answer area address specified by ANSAREA is valid.</li> <li>• If the caller is running in AR-mode and the ANSAREA parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• The address specified for ANSAREA is addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are calling IXLLIST while disabled, ANSAREA must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLLIST in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLIST macro.</li> </ul>
8	xxxx0839	<p><b>Equate Symbol:</b> IXLRSNCODEBADREQTOKENAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by REQTOKEN is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The request token area specified by REQTOKEN is valid.</li> <li>• If the caller is running in AR-mode and the REQTOKEN parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If you are calling IXLLIST while disabled, REQTOKEN must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLLIST in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLIST macro.</li> </ul>
8	xxxx083D	<p><b>Equate Symbol:</b> IXLRSNCODEBADANSLEN</p> <p><b>Meaning:</b> Program error. The length of the answer area, as specified by ANSLEN, is not sufficient to contain answer area information. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Increase the size of the answer area, and rerun your program. The minimum size of the ANSAREA is in the LAA_LEN field of the IXLYLAA mapping macro.</p>

Table 73. Return and Reason Codes for IXLLIST REQUEST=WRITE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx083E	<p><b>Equate Symbol:</b> IXLRNCODEMAXLISTKEY</p> <p><b>Meaning:</b> Program error. The list key to be assigned to an entry that was being created or moved was not valid. Either the list key or the list key plus the list key increment value was greater than the maximum list key.</p> <p><b>Action:</b> Ensure that you specified a correct list key increment. Depending on the protocol you are using for assigning list key values, you might want to issue a WRITE_LCONTROLS request to update either the list key value to a lower value or the maximum list key value to a higher value.</p>
8	xxxx083F	<p><b>Equate Symbol:</b> IXLRNCODEBADENTRYVERSION</p> <p><b>Meaning:</b> The designated entry has a version number that failed the comparison specified by VERSCOMPTYPE. The list entry controls for the entry are returned in the answer area (field LAALCTL).</p> <p><b>Action:</b> None necessary. However, if this return code and reason code are unexpected, you might want to verify the value specified in VERSCOMP and look at the version returned in the answer area.</p>
8	xxxx0840	<p><b>Equate Symbol:</b> IXLRNCODEBADENTRYLIST</p> <p><b>Meaning:</b> The entry designated by ENTRYID or ENTRYNAME exists in the structure, but does not reside on the list specified by LISTNUM. The list entry controls for the entry are returned in the answer area (field LAALCTL).</p> <p><b>Action:</b> None necessary. However, if you did not expect this return and reason code, you might want to verify what list this entry is on. Maybe the entry was moved, or you designated the wrong list in LISTNUM. Check the protocol for using this list.</p> <p>The information returned in the LAALCTL field of the answer area contains the number of the list that this list entry is on as well as other control information.</p>
8	xxxx0841	<p><b>Equate Symbol:</b> IXLRNCODEBADENTRYNAME</p> <p><b>Meaning:</b> Program error. Entry creation was attempted, but no entry was created for one of the following reasons:</p> <ul style="list-style-type: none"> <li>• No entry name was specified (and the structure supports names).</li> <li>• The specified entry name is not unique within the structure.</li> </ul> <p>The list entry controls for the first entry on the list are returned in the answer area (field LAALCTL).</p> <p><b>Action:</b> Be sure to provide a unique entry name when creating entries to be written to structures that support entry names.</p>

Table 73. Return and Reason Codes for IXLLIST REQUEST=WRITE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0842	<p><b>Equate Symbol:</b> IXLRNCODEPERSISTENTLOCK</p> <p><b>Meaning:</b> Program error. The request specifying LOCKOPER=SET with LOCKMODE=UNCOND, or LOCKOPER=NOTHELD with LOCKMODE=UNCOND failed because the lock is held by a connection that is in the failed-persistent state. The connection identifier of the lock owner is returned in the answer area (field LAACONID).</p> <p><b>Action:</b> Either perform recovery for the connection, or wait until recovery for the connection is performed.</p>
8	xxxx0845	<p><b>Equate Symbol:</b> IXLRNCODENONAMES</p> <p><b>Meaning:</b> Program error. A list entry was designated by entry name, but the structure does not support entry names.</p> <p><b>Action:</b> Ensure that you are connected to the intended structure. Ensure that the correct specification was made for REFOPTION on the IXLCONN macro. You may want to issue IXLMG to get more information about the structure.</p>
8	xxxx0846	<p><b>Equate Symbol:</b> IXLRNCODEBADLOCKINDEX</p> <p><b>Meaning:</b> Program error. The specified LOCKINDEX exceeds the size of the lock table for the structure.</p> <p><b>Action:</b> Correct LOCKINDEX to specify an index that is contained within the lock table. The maximum value for the LOCKINDEX is one less than the value specified by the LOCKENTRIES parameter on the IXLCONN macro.</p>
8	xxxx0847	<p><b>Equate Symbol:</b> IXLRNCODEBADLISTNUMBER</p> <p><b>Meaning:</b> Program error. The specified LISTNUM value exceeds the number of lists for the structure.</p> <p><b>Action:</b> Correct LISTNUM to specify a list number that exists in the structure. The number of lists allocated for a structure is determined on the LISTHEADERS parameter of the IXLCONN macro. The list numbers go from 0 to n-1, where n is the number of lists specified.</p>
8	xxxx0848	<p><b>Equate Symbol:</b> IXLRNCODEBADRESET</p> <p><b>Meaning:</b> Program error. LOCKOPER=RESET was specified for a lock not currently held by the invoker. The value of the connection ID holding the lock is returned in the answer area (field LAACONID).</p> <p><b>Action:</b> Check your code to ensure that your connection did not already reset the lock.</p>

Table 73. Return and Reason Codes for IXLLIST REQUEST=WRITE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx084A	<p><b>Equate Symbol:</b> IXLRSNCODENOKEYS</p> <p><b>Meaning:</b> Program error. The structure does not support the use of entry keys. The IXLLIST request type either required the structure to support entry keys or designated a list entry or list position by list number and entry key.</p> <p><b>Action:</b> Ensure that you are connected to the intended structure. The use of keys for a structure is determined by the REFOPTION keyword on the IXLCONN macro.</p>
8	xxxx084B	<p><b>Equate Symbol:</b> IXLRSNCODENOLOCKS</p> <p><b>Meaning:</b> Program error. A locking operation was requested, but the structure does not contain a lock table.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• You are connected to the intended structure.</li> <li>• You intended to perform a locking operation.</li> </ul> <p>The use of locks is determined by the LOCKENTRIES keyword on the IXLCONN macro.</p>
8	xxxx0851	<p><b>Equate Symbol:</b> IXLRSNCODENOSUSPENDISABLE</p> <p><b>Meaning:</b> Program error. The request failed because MODE=SYNCSUSPEND was specified, but the caller is disabled and cannot be suspended.</p> <p><b>Action:</b> Either specify another MODE value, or become enabled (release the CPU lock), then reissue the request.</p>
8	xxxx0854	<p><b>Equate Symbol:</b> IXLRSNCODEBADLOCKCOMP</p> <p><b>Meaning:</b> Program error. The connection identifier specified for LOCKCOMP is not valid.</p> <p><b>Action:</b> The connection identifier is returned in the answer area (mapped by IXLYCONA) when the IXLCONN macro was issued.</p>
8	xxxx0859	<p><b>Equate Symbol:</b> IXLRSNCODEBADLISTAUTH</p> <p><b>Meaning:</b> The list authority specified for AUTHCOMP failed the comparison specified by AUTHCOMPTYPE for the list authority of the specified list. The current list authority (field LAALISTAUTH) and description (field LAALISTDESC) are returned in the answer area.</p> <p><b>Action:</b> None required; however you might want to take some action depending on your application. The list authority is set to binary zeros when the structure is allocated. When IXLLIST is issued with the NEWAUTH keyword, the list authority is changed if the correct comparison list authority value is specified. Check the answer area to determine the list authority value for the list specified. Verify that the correct list number was specified.</p>

Table 73. Return and Reason Codes for IXLLIST REQUEST=WRITE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0865	<b>Equate Symbol:</b> IXLRSNCODEBADBUFSPEC <b>Meaning:</b> Program error. There is an error in the buffer specification. <b>Action:</b> Check the following: <ul style="list-style-type: none"> <li>• If BUFLIST was specified, check the requirements for BUFLIST, BUFNUM, and BUFINCRNUM.</li> <li>• If BUFFER was specified, check the requirements for BUFFER and BUFSIZE.</li> <li>• Buffer pointer(s) in BUFLIST</li> <li>• Buffer boundaries.</li> </ul>
8	xxxx0866	<b>Equate Symbol:</b> IXLRSNCODEBADBUFKEY <b>Meaning:</b> Program error. The buffer storage key specified by BUFSTGKEY is incorrect or BUFSTGKEY was not specified and the PSW key does not match the key of the buffers. The data cannot be fetched from the specified buffer area. <b>Action:</b> Check the following: <ul style="list-style-type: none"> <li>• Determine if the key of the storage being used for the buffers is different from the PSW key.</li> <li>• If BUFSTGKEY was specified, verify that the key was put in the correct bits (see the explanation of this parameter for more information).</li> <li>• If you are calling IXLLIST in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLIST macro.</li> </ul>
8	xxxx0867	<b>Equate Symbol:</b> IXLRSNCODEBADBUFLIST <b>Meaning:</b> Program error. The 128-byte storage area specified by BUFLIST is not addressable. <b>Action:</b> Ensure that the address specified by BUFLIST is valid.
8	xxxx086A	<b>Equate Symbol:</b> IXLRSNCODEBADELEMNUM <b>Meaning:</b> Program error. The value specified for ELEMNUM is not valid. <b>Action:</b> Correct the ELEMNUM specification to be within the allowable range: from zero to whatever MAXELEMNUM value was returned to the connector in the connect answer area.
8	xxxx08AD	<b>Equate Symbol:</b> IXLRSNCODEBADHIGHSHAREDVIRT <b>Meaning:</b> Program error. The request specified a high shared virtual storage area (above 2GB). <b>Action:</b> None required.

Table 73. Return and Reason Codes for IXLLIST REQUEST=WRITE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C06	<p><b>Equate Symbol:</b> IXLRSNCODENOCNN</p> <p><b>Meaning:</b> Environmental error. This system does not have connectivity to the coupling facility that contains the list structure. Possible reasons for this are:</p> <ul style="list-style-type: none"> <li>• The operator issued VARY PATH,OFFLINE.</li> <li>• The operator issued CONFIG CHP,OFFLINE.</li> <li>• Hardware errors to the coupling facility.</li> <li>• Facility or path failure to the coupling facility.</li> </ul> <p><b>Action:</b> Begin rebuilding the structure on a different coupling facility, or disconnect from the structure.</p>
C	xxxx0C13	<p><b>Equate Symbol:</b> IXLRSNCODEREQPURGED</p> <p><b>Meaning:</b> Environmental error. The request was purged prior to completion of the request. Possible reasons include:</p> <ul style="list-style-type: none"> <li>• The connector failed.</li> <li>• The connector disconnected.</li> <li>• The requestor failed.</li> <li>• The request was purged by IXLPURGE.</li> <li>• Requests were purged when the connector provided an IXLEERSP response for the Rebuild Stop or Rebuild Cleanup event.</li> <li>• The secondary address space was no longer valid.</li> </ul> <p><b>Action:</b> None if this is expected. Otherwise, determine why the connector failed.</p>
C	xxxx0C14	<p><b>Equate Symbol:</b> IXLRSNCODESTATUSUNKNOWN</p> <p><b>Meaning:</b> Environmental error. The request has completed, but the final disposition of the request cannot be determined. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the validity of your data by comparing the expected results with what is in the coupling facility.</li> <li>• Consider running rebuild processing to move the structure to attempt to protect it from further errors of this kind.</li> </ul>



Table 73. Return and Reason Codes for IXLLIST REQUEST=WRITE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C17	<p><b>Equate Symbol:</b> IXLRNCODESTRFULL</p> <p><b>Meaning:</b> Environmental error. The request attempted to create a new entry or overwrite an existing entry, but the structure is full and cannot accommodate any more entries.</p> <p><b>Action:</b> Determine why the structure is full. You should be monitoring the usage of the structure every time a read or write is done (LAATOTALCNT is the total count of allocated entries in the list structure, and LAATOTALELECNT is the total count of allocated elements in the list structure). This data should be evaluated periodically to ensure structure resources are being used efficiently. You might be able to delete existing entries to free up space, rebuild the structure to make it larger if rebuild is allowed (IXLCONN macro, ALLOWREBLD parameter), or alter the size of the structure (IXLALTER macro).</p>
C	xxxx0C18	<p><b>Equate Symbol:</b> IXLRNCODELISTFULL</p> <p><b>Meaning:</b> Environmental error. The entry could not be placed on the designated target list because the list is full.</p> <p><b>Action:</b> Either change the maximum number of entries allowed on the list by specifying a new LISTLIMIT on a WRITE_LCONTROLS request. You should be monitoring the usage of the list structure every time a read or write is done. (LAATOTALCNT is the total count of allocated entries in the list structure, and LAATOTALELECNT is the total count of allocated elements in the list structure.) This data should be evaluated periodically to ensure structure resources are being used efficiently.</p>
C	xxxx0C25	<p><b>Equate Symbol:</b> IXLRNCODESTRFAILURE</p> <p><b>Meaning:</b> Environmental error. The list structure failed prior to completion of the request.</p> <p><b>Action:</b> Either rebuild or disconnect from the structure.</p>
C	xxxx0CA0	<p><b>Equate Symbol:</b> IXLRNCODEQUIESCEDSUSPENDFAIL</p> <p><b>Meaning:</b> Environmental error. The request is failed because the structure is quiesced for a system-managed process and SUSPEND=FAIL is specified on the IXLCONN.</p> <p><b>Action:</b> None, if this is expected.</p>
C	xxxxFFFF	<p><b>Equate Symbol:</b> IXLRNCODENOTAVAILABLE</p> <p><b>Meaning:</b> Environmental error. XES functions are not available. This could occur because the coupling facility hardware necessary to provide XES functions is not present.</p> <p><b>Action:</b> Re-IPL the system, or follow your particular failure management protocol.</p>

Table 73. Return and Reason Codes for IXLLIST REQUEST=WRITE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
10	xxxx10xx	<b>Meaning:</b> System error. XES processing failure. The state of the involved structure and the disposition of the request are unpredictable. The answer area (ANSAREA) fields do not contain valid information. <b>Action:</b> Contact the IBM support center.

---

## Chapter 74. IXLLIST REQUEST=WRITE\_LCONTROLS

### Description

---

A WRITE\_LCONTROLS request allows you to modify the list controls for the list specified by LISTNUM. Any one or more of the following list attributes can be changed:

- The list authority (NEWAUTH).
- The list limit (LISTLIMIT) that defines the number of elements or entries allowed on the list.
- The list description (LISTDESC) that you define for the list.

With a coupling facility of CFLEVEL=1 or higher, you can also modify the following additional list controls for the list specified by LISTNUM.

- The list key (LISTKEY) and the maximum value for the list key associated with the list (MAXLISTKEY).
- The list cursor and the direction in which processing with the cursor is to occur (SETCURSOR).  
The direction you specify applies only when a request to update the list cursor specifies CURSORUPDTYPE=NEXTCOND.

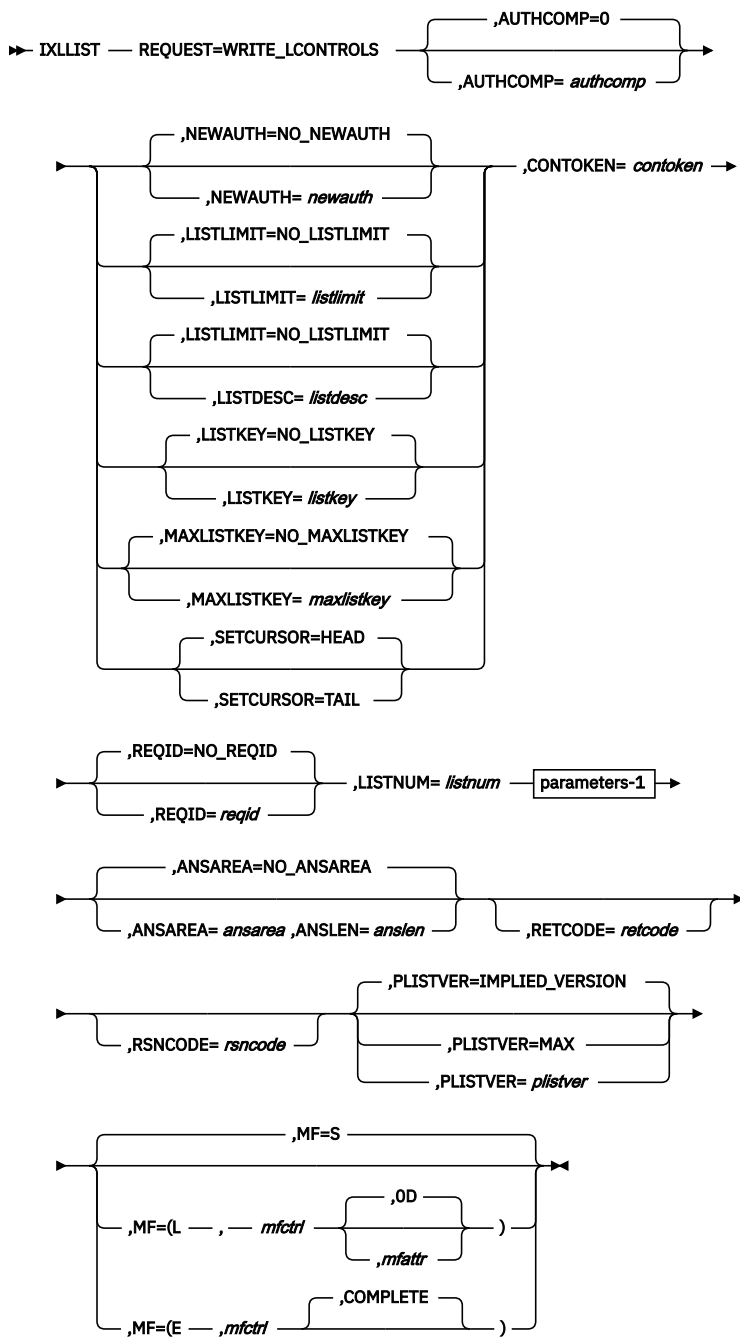
An authority comparison is required for all WRITE\_LCONTROLS requests. If you do not specify AUTHCOMP, the system uses a default value of binary zeros for the authority comparison.

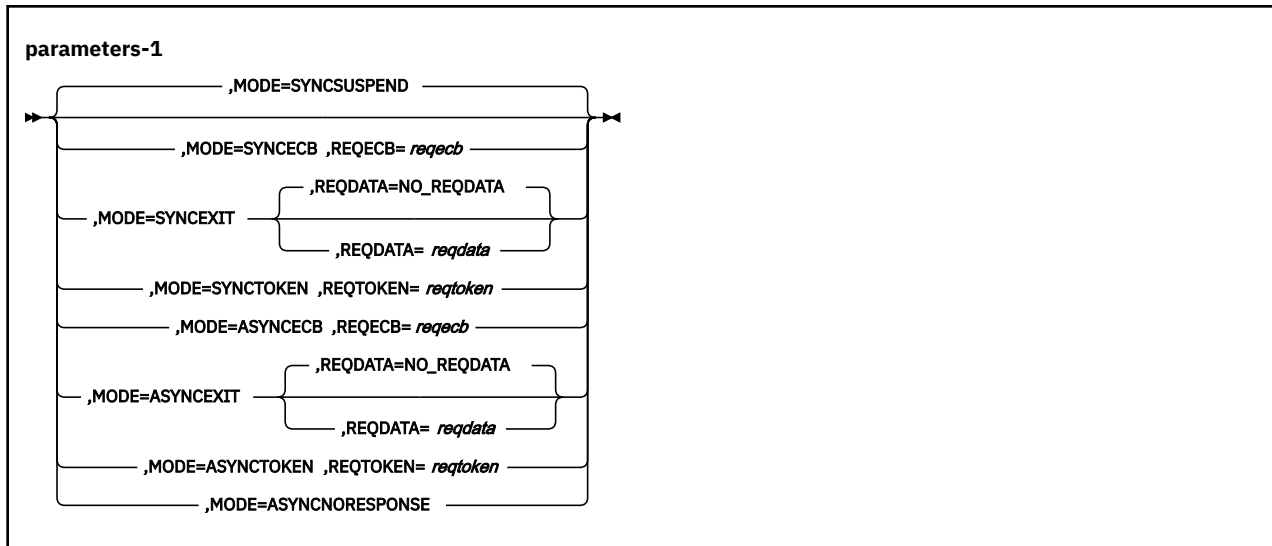
### Syntax Diagram

---

The syntax diagram for IXLLIST REQUEST=WRITE\_LCONTROLS is as follows:

## main diagram



**Note:**

1. If MODE=SYNCTOKEN or MODE=ASYNCTOKEN is specified, ANSAREA=*ansarea* and ANSLEN=*anslen* are required.
2. At least one of NEWAUTH, LISTLIMIT, and LISTDESC must be specified. More than one of these parameters may be specified as well.

## Parameter Descriptions

The parameter descriptions for REQUEST=WRITE\_LCONTROLS are listed in alphabetical order. Default values are underlined:

**REQUEST=WRITE\_LCONTROLS**

Use this input parameter to change the list controls for a specified list.

**,ANSAREA=NO\_ANSAREA****,ANSAREA=*ansarea***

Use this output parameter to specify an answer area to contain information returned from a failed request. The format of the answer area is described by mapping macro IXLYLAA.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an area (with a size of ANSLEN) where the information about the request will be returned.

**,ANSLEN=*anslen***

Use this input parameter to specify the size of the storage area specified by ANSAREA.

Check the prologue of the IXLYLAA mapping macro for the minimum required size of the answer area, or use the length field (LAA\_LEN) in the LAA.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 2-byte field that contains the size, in bytes, of the answer area.

**,AUTHCOMP=0****,AUTHCOMP=*authcomp***

Use this input parameter to specify a value to be compared to the list authority value of the list specified by LISTNUM.

The AUTHCOMP value must equal the list authority of the LISTNUM list, or the request terminates with no change to the structure.

**Note:** At the time the structure was allocated, the list authority value for each list in the structure was initialized to binary zeros.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-byte field that contains the list comparison authority value, or take the default, AUTHCOMP=0.

**,CONTOKEN=contoken**

Use this input parameter to specify the connect token that was returned by the IXLCONN service. The connect token uniquely identifies your connection to the list structure, and must be specified on each IXLLIST invocation.

The connect token is available in the IXLCONN answer area that is mapped by IXLYCONA.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 16-byte field that contains the connect token.

**,LISTDESC=listdesc**

Use this input parameter to specify a description to be associated with the list specified by LISTNUM. The description can be anything you want.

If LISTDESC is not specified, the list description of the LISTNUM list remains unchanged.

**Note:** At the time the structure was allocated, the list description for each list in the structure was initialized to binary zeros.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 32-byte field that contains the list description.

**,LISTKEY=NO LISTKEY****,LISTKEY=listket**

Use this input parameter to specify the list key to be associated with the list. You may assign a list key value to list entries when they are created or moved.

If LISTKEY is not specified, the list key for the designated list remains unchanged.

**Note:** At the time the structure was allocated, the list key for each list in the structure was initialized to binary zeros. Use this parameter only with a coupling facility of CFLEVEL=1 or higher.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the 16-character field that contains the list key to be associated with the list.

**,LISTLIMIT=listlimit**

Use this input parameter to specify the list limit—the maximum allowable number of entries or elements that can reside on the list specified by LISTNUM. Whether the limit is counted in terms of entries or elements depends on what you specified for the LISTCNTLTYPE parameter on the IXLCONN macro.

If LISTLIMIT is not specified, the list limit of the LISTNUM list remains unchanged.

**Note:** At the time the structure was allocated, the list limit for each list in the structure was initialized to the total number of entries or elements in the structure, depending on the value specified for LISTCNTLTYPE on the IXLCONN macro.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 4-byte field that contains the list limit.

**,LISTNUM=listnum**

Use this input parameter to specify the number of the list to which the list controls specified by this request will apply.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 4-byte field that contains the number of the list.

**,MAXLISTKEY=NO MAXLISTKEY****,MAXLISTKEY=maxlistkey**

Use this input parameter to specify the maximum list key value to be associated with the list.

**Note:** At the time the structure was allocated, the maximum list key value for each list in the structure was initialized to binary zeros. Use this parameter only with a coupling facility of CFLEVEL=1 or higher.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the 16-byte field that contains the maximum list key value.

**,MF=S**  
**,MF=(L,mfctrl)**  
**,MF=(L,mfctrl,mfattr)**  
**,MF=(L,mfctrl,0D)**  
**,MF=(M,mfctrl)**  
**,MF=(M,mfctrl,COMPLETE)**  
**,MF=(M,mfctrl,NOCHECK)**  
**,MF=(E,mfctrl)**  
**,MF=(E,mfctrl,COMPLETE)**  
**,MF=(E,mfctrl,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

#### **,mfctrl**

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

#### **,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

#### **,COMPLETE**

#### **,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

#### **COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if SMILE=*var* were an optional parameter and the default is SMILE=NO\_SMILE then it would not be documented. However, if the default was SMILE=:), then it would be documented because a value would be the default.

#### **NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

,**MODE=SYNCSUSPEND**  
 ,**MODE=SYNCECB**  
 ,**MODE=SYNCEXIT**  
 ,**MODE=SYNCTOKEN**  
 ,**MODE=ASYNCECB**  
 ,**MODE=ASYNCEXIT**  
 ,**MODE=ASYNCTOKEN**  
 ,**MODE=ASYNCSNORESPONSE**

Use this input parameter to specify:

- Whether the request is to be performed synchronously or asynchronously
- How you want to be notified of request completion

**MODE=SYNCSUSPEND**

The request is suspended until it can complete synchronously. To use this option, your program must be enabled for I/O and external interrupts.

**MODE=SYNCECB**

The request processes synchronously if possible. If the request processes asynchronously, the ECB specified by REQECB is posted when the request completes.

**MODE=SYNCEXIT**

The request processes synchronously if possible. If the request processes asynchronously, your complete exit is given control when the request completes.

**MODE=SYNCTOKEN**

The request processes synchronously if possible. If the request processes asynchronously, an asynchronous request token is returned in the area specified by REQTOKEN on invocation of the request. Use the returned request token on the IXLFComp macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=SYNCTOKEN is specified.

**MODE=ASYNCECB**

The request processes asynchronously. The ECB specified by REQECB is posted when the request completes.

**MODE=ASYNCEXIT**

The request processes asynchronously. Your complete exit is given control when the request completes.

**MODE=ASYNCTOKEN**

The request processes asynchronously. An asynchronous request token is returned in the area specified by REQTOKEN upon invocation of the request. Use the returned request token on the IXLFComp macro to determine whether your request has completed.

**Note:** ANSAREA is a required parameter when MODE=ASYNCTOKEN is specified.

**MODE=ASYNCSNORESPONSE**

The request processes asynchronously. No notification of request completion is provided. The answer area (ANSAREA) fields will not contain valid information when this mode is specified.

**Note:** You cannot code MODE=ASYNCSNORESPONSE with LOCKINDEX, BUFFER, or BUFLIST.

,**NEWAUTH=newauth**

Use this input parameter to specify a new list authority value to replace the existing list authority value of the list specified by LISTNUM.

If NEWAUTH is not specified, the list authority of the LISTNUM list will remain unchanged.

**Note:** At the time the structure was allocated, the list authority value for each list in the structure was initialized to binary zeros.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-byte field that contains the new list authority value.



**,PLISTVER=IMPLIED\_VERSION****,PLISTVER=MAX****,PLISTVER=*plistver***

Use this input parameter to specify the version of the macro. See [“Understanding IXLLIST Version Support”](#) on page 965 for a descriptions of the options available with PLISTVER.

**,REQDATA=NO\_REQDATA****,REQDATA=*reqdata***

Use this input parameter with MODE=SYNCEXIT or MODE=ASYNCEXIT to pass any data you choose to the complete exit. The exit will get control only if the request is processed asynchronously.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the data to be passed to the complete exit.

**,REQECB=*reqecb***

Use this output parameter with either MODE=SYNCECB or MODE=ASYNCECB to specify the address of an ECB, which is to be posted when the request completes if the request was processed asynchronously.

Before coding REQECB, you must ensure that:

- You initialize the ECB before you issue the request.
- The ECB resides in either common storage or the home address space where IXLCONN was issued.
- Any tasks that wait for the ECB to be posted reside in the home address space where IXLCONN was issued.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that contains the address of the ECB to be posted when the request completes. The ECB must be aligned on a fullword boundary.

**,REQID=NO\_REQID****,REQID=*reqid***

Use this input parameter to specify a user-defined request identifier to be associated with the request. You can specify this request identifier on the IXPURGE macro to cancel a request that has not yet been processed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the user-defined request identifier.

**,REQTOKEN=*reqtoken***

Use this output parameter with either MODE=SYNCTOKEN or MODE=ASYNCTOKEN to specify the address of a storage area to receive the request token that is returned when the request will be processed asynchronously. This token, which uniquely identifies the request, must be used as input to the IXLFCOMP macro, which you use to determine if the request has completed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field where the system will put the request token.

**,RETCODE=*retcode***

Use this output parameter to specify a field to contain the return code. (The return code is also returned in register 15.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the return code when the request has completed.

**,RSNCODE=*rsncode***

Use this output parameter to specify a field to contain the reason code returned, if applicable. (The reason code is also returned in register 0.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the reason code (if any) when the request has completed.

**,SETCURSOR=HEAD****,SETCURSOR=TAIL**

Use this input parameter to indicate that the list cursor and the list cursor direction are to be set.

**Note:** At the time the structure was allocated, the list cursor for each list in the structure was initialized to binary zeros. The default list cursor direction for all lists in the structure was set to a head-to-tail direction. Use this parameter only with a coupling facility of CFLEVEL=1 or higher.

#### **SETCURSOR=HEAD**

The list cursor is to be set to the entry identifier for the first entry on the list and the list cursor direction is to be set in a head-to-tail direction. If the list is empty, the list cursor is reset to binary zeros.

#### **SETCURSOR=TAIL**

The list cursor is to be set to the entry identifier for the last entry on the list and the list cursor direction is to be set in a tail-to-head direction. If the list is empty, the list cursor is reset to binary zeros.

## ABEND Codes

---

Abend X'026' (See *z/OS MVS System Codes* for more information on this abend.)

## Return and Reason Codes

---

When the system returns control to your program:

- GPR 15 (and RETCODE, if specified) contains a return code.
- If applicable, GPR 0 (and RSNCODE, if specified) contains a reason code.

**Note:** The return and reason codes will also be put into the answer area (mapped by IXLYLAA), if there is one, when the request completes.

The IXLYCON macro provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

<b>0</b>	IXLRETCODEOK
<b>4</b>	IXLRETCODEWARNING
<b>8</b>	IXLRETCODEPARMERROR
<b>C</b>	IXLRETCODEENVERROR
<b>10</b>	IXLRETCODECOMPERROR

The following table contains hexadecimal return and reason codes, the equate symbols associated with each reason code, and the meaning and suggested action for each return and reason code.

Table 74. Return and Reason Codes for IXLLIST REQUEST=WRITE\_LCONTROLS Macro

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0	None.	<p><b>Meaning:</b> If you specified a MODE value of ASYNCECB, ASYNCEXIT, ASYNCTOKEN, or ASYNCFNORESPONSE, the request has been successfully initiated. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed. For any other MODE value, the request has successfully completed.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=ASYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=ASYNCEXIT, the connection's complete exit will be given control when the request has completed.</li> <li>• If you specified MODE=ASYNCTOKEN, use REQTOKEN when calling IXLFCOMP to determine when the request has completed.</li> </ul>
4	xxxx0402	<p><b>Equate Symbol:</b> IXLRNCFNOEASYNCH</p> <p><b>Meaning:</b> The request could not be processed synchronously. It will be processed asynchronously. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed.</p> <ul style="list-style-type: none"> <li>• If MODE=SYNCECB was specified, the ECB specified for REQECB will be posted when the request has finished.</li> <li>• If MODE=SYNCEXIT was specified, the connection's complete exit will be given control when the request has finished.</li> <li>• If MODE=SYNCTOKEN was specified, an asynchronous request token was returned in the area specified by REQTOKEN. This request token may be specified on the IXLFCOMP macro to determine when the request has finished.</li> </ul> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=SYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=SYNCEXIT, the connection's complete exit will be given control when the request has completed.</li> <li>• If you specified MODE=SYNCTOKEN, use REQTOKEN when calling IXLFCOMP to determine when the request has completed.</li> </ul>

Table 74. Return and Reason Codes for IXLLIST REQUEST=WRITE\_LCONTROLS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0801	<p><b>Equate Symbol:</b> IXLRSNCODEBADPARMLIST</p> <p><b>Meaning:</b> Program error. The parameter list for this request is not addressable. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Verify that:</p> <ul style="list-style-type: none"> <li>• The parameter list address is uncorrupted.</li> <li>• The parameter list is addressable in the caller's primary address space.</li> <li>• If you are issuing this macro while disabled, the parameter list resides in either page-fixed or DREF storage.</li> <li>• If you are issuing this macro in AR-mode and you specified the parameter list address using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If you are issuing this macro in AR-mode, SYSSTATE ASCENV=AR must be issued before issuing this macro.</li> </ul>
8	xxxx0804	<p><b>Equate Symbol:</b> IXLRSNCODEBADVERSIONNUM (or IXLRSNCODEBADVERSION#)</p> <p><b>Meaning:</b> The version number in the macro parameter list is not compatible with the level of XES currently being used. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that your program did not overlay the parameter list storage.</li> <li>• Verify that your program was assembled with the correct macro library for the release of MVS on which your program is running.</li> </ul>

Table 74. Return and Reason Codes for IXLLIST REQUEST=WRITE\_LCONTROLS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx080A	<p><b>Equate Symbol:</b> IXLSRNCODEBADCONTOKEN</p> <p><b>Meaning:</b> Program error. The specified connect token (CONTOKEN) was not valid for one of the following reasons:</p> <ol style="list-style-type: none"> <li>1. The user with the connection identifier represented by the token has disconnected from the structure.</li> <li>2. The connector's task (the task that issued IXLCONN) ended.</li> <li>3. The specified token is not the token that was returned from IXLCONN.</li> <li>4. The request was issued from an address space other than the address space in which IXLCONN was issued.</li> <li>5. The connect token was invalidated during rebuild.</li> <li>6. The connect token was invalidated by XES.</li> </ol> <p><b>Note:</b> The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Take the action with the corresponding meaning.</p> <ol style="list-style-type: none"> <li>1. Discontinue use of the structure. Perform recovery and cleanup for the structure and check the protocol for the use of the structure to determine why you did not know of the disconnection.</li> <li>2. Discontinue use of the structure. Perform recovery and cleanup for the structure and check the protocol for the use of the structure to determine why you did not know that the task ended.</li> <li>3. The contoken is returned in the CONACONTOKEN field of the answer area specified on the IXLCONN request.</li> <li>4. Issue your request from the same address space the IXLCONN was issued in.</li> <li>5. Wait for the rebuild to complete, and try again.</li> <li>6. Discontinue use of the structure. Perform recovery and cleanup for the structure.</li> </ol>
8	xxxx0824	<p><b>Equate Symbol:</b> IXLSRNCODEWRONGSTRTYPE</p> <p><b>Meaning:</b> Program error. The connection specified by CONTOKEN is not to a list structure.</p> <p><b>Action:</b> Specify the appropriate CONTOKEN value, and rerun your program. The CONTOKEN is returned in the answer area provided on the IXLCONN invocation. The type of structure built is passed on the TYPE parameter of the IXLCONN macro.</p>

Table 74. Return and Reason Codes for IXLLIST REQUEST=WRITE\_LCONTROLS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0838	<p><b>Equate Symbol:</b> IXLRSNCODEBADANSAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by ANSAREA is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The answer area address specified by ANSAREA is valid.</li> <li>• If the caller is running in AR-mode and the ANSAREA parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• The address specified for ANSAREA is addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are calling IXLLIST while disabled, ANSAREA must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLLIST in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLIST macro.</li> </ul>
8	xxxx0839	<p><b>Equate Symbol:</b> IXLRSNCODEBADREQTOKENAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by REQTOKEN is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The request token area specified by REQTOKEN is valid.</li> <li>• If the caller is running in AR-mode and the REQTOKEN parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If you are calling IXLLIST while disabled, REQTOKEN must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLLIST in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLIST macro.</li> </ul>
8	xxxx083D	<p><b>Equate Symbol:</b> IXLRSNCODEBADANSLEN</p> <p><b>Meaning:</b> Program error. The length of the answer area, as specified by ANSLEN, is not sufficient to contain answer area information. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Increase the size of the answer area, and rerun your program. The minimum size of the ANSAREA is in the LAA_LEN field of the IXLYLAA mapping macro.</p>

Table 74. Return and Reason Codes for IXLLIST REQUEST=WRITE\_LCONTROLS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0847	<p><b>Equate Symbol:</b> IXLRSNCODEBADLISTNUMBER</p> <p><b>Meaning:</b> Program error. The specified LISTNUM value exceeds the number of lists for the structure.</p> <p><b>Action:</b> Correct LISTNUM to specify a list number that exists in the structure. The number of lists allocated for a structure is determined on the LISTHEADERS parameter of the IXLCONN macro. The list numbers go from 0 to n-1, where n is the number of lists specified.</p>
8	xxxx0851	<p><b>Equate Symbol:</b> IXLRSNCODENOSUSPENDISABLE</p> <p><b>Meaning:</b> Program error. The request failed because MODE=SYNCSUSPEND was specified, but the caller is disabled and cannot be suspended.</p> <p><b>Action:</b> Either specify another MODE value, or become enabled (release the CPU lock), then reissue the request.</p>
8	xxxx0859	<p><b>Equate Symbol:</b> IXLRSNCODEBADLISTAUTH</p> <p><b>Meaning:</b> The list authority specified for AUTHCOMP failed the comparison specified by AUTHCOMPTYPE for the list authority of the specified list. The current list authority (field LAALISTAUTH) and description (field LAALISTDESC) are returned in the answer area.</p> <p><b>Action:</b> None required; however you might want to take some action depending on your application. The list authority is set to binary zeros when the structure is allocated. When IXLLIST is issued with the NEWAUTH keyword, the list authority is changed if the correct comparison list authority value is specified. Check the answer area to determine the list authority value for the list specified. Verify that the correct list number was specified.</p>
C	xxxx0C06	<p><b>Equate Symbol:</b> IXLRSNCODENOCONN</p> <p><b>Meaning:</b> Environmental error. This system does not have connectivity to the coupling facility that contains the list structure. Possible reasons for this are:</p> <ul style="list-style-type: none"> <li>• The operator issued VARY PATH,OFFLINE.</li> <li>• The operator issued CONFIG CHP,OFFLINE.</li> <li>• Hardware errors to the coupling facility.</li> <li>• Facility or path failure to the coupling facility.</li> </ul> <p><b>Action:</b> Begin rebuilding the structure on a different coupling facility, or disconnect from the structure.</p>

Table 74. Return and Reason Codes for IXLLIST REQUEST=WRITE\_LCONTROLS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C13	<p><b>Equate Symbol:</b> IXLRSNCODEREQPURGED</p> <p><b>Meaning:</b> Environmental error. The request was purged prior to completion of the request. Possible reasons include:</p> <ul style="list-style-type: none"> <li>• The connector failed.</li> <li>• The connector disconnected.</li> <li>• The requestor failed.</li> <li>• The request was purged by IXLPURGE.</li> <li>• Requests were purged when the connector provided an IXLEERSP response for the Rebuild Stop or Rebuild Cleanup event.</li> <li>• The secondary address space was no longer valid.</li> </ul> <p><b>Action:</b> None if this is expected. Otherwise, determine why the connector failed.</p>
C	xxxx0C14	<p><b>Equate Symbol:</b> IXLRSNCODESTATUSUNKNOWN</p> <p><b>Meaning:</b> Environmental error. The request has completed, but the final disposition of the request cannot be determined. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the validity of your data by comparing the expected results with what is in the coupling facility.</li> <li>• Consider running rebuild processing to move the structure to attempt to protect it from further errors of this kind.</li> </ul>
C	xxxx0C25	<p><b>Equate Symbol:</b> IXLRSNCODESTRFAILURE</p> <p><b>Meaning:</b> Environmental error. The list structure failed prior to completion of the request.</p> <p><b>Action:</b> Either rebuild or disconnect from the structure.</p>
C	xxxx0CA0	<p><b>Equate Symbol:</b> IXLRSNCODEQUIESCEDSUSPENDFAIL</p> <p><b>Meaning:</b> Environmental error. The request is failed because the structure is quiesced for a system-managed process and SUSPEND=FAIL is specified on the IXLCONN.</p> <p><b>Action:</b> None, if this is expected.</p>
C	xxxxFFFF	<p><b>Equate Symbol:</b> IXLRSNCODENOTAVAILABLE</p> <p><b>Meaning:</b> Environmental error. XES functions are not available. This could occur because the coupling facility hardware necessary to provide XES functions is not present.</p> <p><b>Action:</b> Re-IPL the system, or follow your particular failure management protocol.</p>



Table 74. Return and Reason Codes for IXLLIST REQUEST=WRITE\_LCONTROLS Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
10	xxxx10xx	<p><b>Meaning:</b> System error. XES processing failure. The state of the involved structure and the disposition of the request are unpredictable. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Contact the IBM support center.</p>



## Chapter 75. IXLLOCK Services

### Description

MVS allows a connected coupling facility application to request the following lock services through the IXLLOCK macro:

- **Obtain** either shared or exclusive ownership of a resource that is not currently owned or is pending ownership by this connected user and optionally assign additional user-defined ownership attributes. (REQUEST=OBTAIN)
- **Alter** the attributes of an owned resource or replace a previous OBTAIN or ALTER request which is pending on the contention exit resource request queue with a more current request. (REQUEST=ALTER)
- **Release** ownership of a resource or, if an outstanding request has been left pending on the contention exit resource request queue, replace the pending request with the more current request to release the resource ownership. As with the ALTER request, you can request that a resource be released before you have received the results of a previous request to OBTAIN or ALTER. (REQUEST=RELEASE)
- **Process multiple** requests for resources with a single IXLLOCK invocation. A lock request block specifies each individual request. You can request that the system service up to 128 requests. The set of request types (OBTAIN, ALTER, or RELEASE) supported by a PROCESSMULT request is a function of the version of the IXLLOCK macro. Version 1 of the IXLLOCK macro supports the PROCESSMULT request type and requires a coupling facility of CFLEVEL=2 or higher. (REQUEST=PROCESSMULT)

The IXLLOCK macro requires that you provide a predefined set of user exit routines that are necessary to accomplish the locking function. You specify the addresses of these routines when you connect to the lock structure with the IXLCONN macro.

- The **contention exit** is executed to resolve contention for a resource. Through the contention exit parameter list (IXLYCEPL), the contention exit can potentially direct XES to grant or deny a request for a resource, modify the ownership characteristics of one or more of the current resource owners, notify a current resource owner that contention exists, or take no action at all. The contention exit that is executed is that of the connected user that XES has assigned contention management responsibilities.
- The **notify exit** can be executed to inform a user that contention exists for a resource that the user owns. Only the contention exit of the connected user chosen to manage the resource contention can request that the notify exit be executed. The notify exit parameter list (IXLYNEPL) provides the user with information about all the current owners and requestors of the resource, in which the user can choose to update its interest in the resource and possibly eliminate the contention. The synchronous update is accomplished with the IXLSYNCH macro.
- The **complete exit** is the means by which XES presents the results of a user's request that could not be processed immediately but was processed asynchronously. The complete exit also is used to notify a user that its ownership state of a resource has been changed (regranted) by the connected user that is managing contention for the resource.
- The **event exit** is the means by which XES reports error and status conditions to connected users. Though not specific to locking functions, use of the event exit must be considered when designing a locking protocol.

### Environment

The following are the environment requirements for the caller.

Environment	Environment requirement
Minimum authorization:	Supervisor state and PSW key 0
Dispatchable unit mode:	Task or SRB

Environment	Environment requirement
Cross memory mode:	Any PASN, any HASN, any SASN. The primary address space must be the same as the primary address space at the time the connection service (IXLCONN) was issued.
AMODE:	31-bit
ASC mode:	Primary or access register (AR)
Interrupt status:	Enabled for I/O and external interrupts
Locks:	No locks held
Control parameters:	Must be in the primary address space See <a href="#">“Restrictions and Limitations”</a> on page 1340.

## Programming Requirements

Before issuing an IXLLOCK service, the connected user must identify itself to the system through the IXLCONN service. On successful completion of the IXLCONN service, a sysplex-wide unique CONNECT token is returned to the user. This token identifies the user's connection to the lock structure. The connect token must be specified on every IXLLOCK request to ensure that the requesting user is allowed access to the designated lock structure.

The IXLYCON macro provides a list of constants for users of IXLLOCK. Include that macro in your program.

## Restrictions and Limitations

The following restrictions apply:

- The caller must provide a 144-byte savearea that starts on a word boundary and is addressable in the caller's primary address space.
- If the caller is running in access register (AR) ASC mode and specifies a macro parameter using explicit register notation, the access register corresponding to the general register must appropriately qualify the general register.
- The virtual storage area specified by the REQBUFFER keyword must reside in fixed or disabled reference storage and must be addressable from the caller's primary address space.
- The PROCESSMULT request type is valid only for structures allocated in a coupling facility of CFLEVEL=2 or higher.

## Input Register Information

Before issuing the IXLLOCK macro, the caller must place in GPR 13 the address of a 144-byte save area to be used by XES. With the exception of GPR 13, the caller does not have to place any information into any register unless using it in register notation for a particular parameter, or using it as a base register.

## Output Register Information

When control returns to the caller of the IXLLOCK macro, the general purpose registers (GPRs) contain:

### Register

#### Contents

**0**

Reason code if GPR15 return code is nonzero

**1**

Used as a work register by the system

**2-13**

Unchanged

**14**

Used as a work register by the system

**15**

Return code

When control returns to the caller of the IXLLOCK macro, the ARs contain:

**Register  
Contents**

**0-1**

Used as a work register by the system

**2-13**

Unchanged

**14-15**

Used as a work register by the system

**For registers that the system changes,** a caller who depends on these registers containing the same value before and after issuing the macro must save these registers before issuing the macro, and restore them after the system returns control.

## Performance Implications

---

See *z/OS MVS Programming: Sysplex Services Guide* for performance information.

## Understanding IXLLOCK version support

---

The IXLLOCK macro supports versions in the range of 0 - 4.

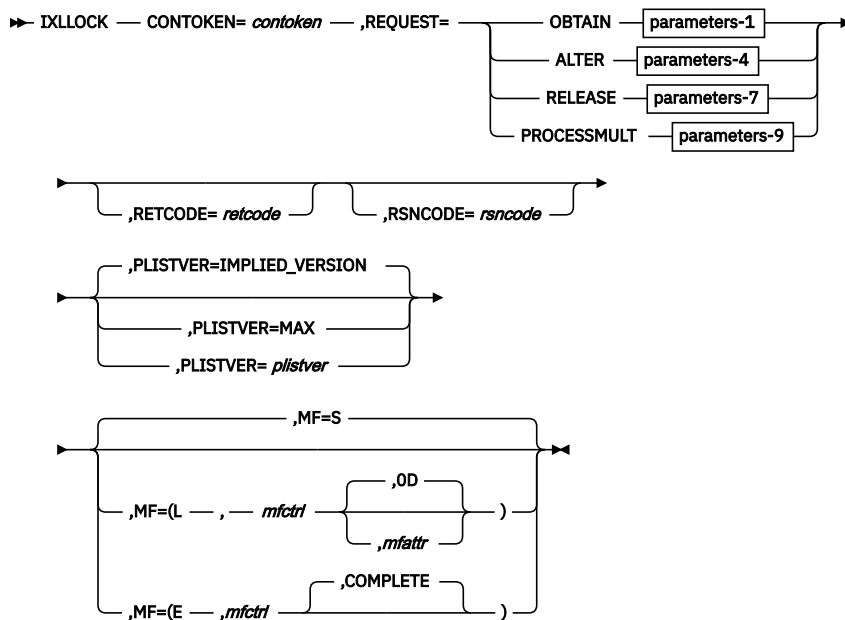
- Keywords not specifically noted here are supported by all versions starting with version 0 of the IXLLOCK macro.
- The following keywords and functions are supported by all versions starting with version 1 of the IXLLOCK macro.
  - MODEVAL
  - MODE=SYNCFAIL
  - MODE=VALUE
  - REQBUFFER
  - REQNUM
  - REQPROC
  - REQUEST=PROCESSMULT
  - SYNCFAILDELAY
- The following keyword is supported by all versions starting with version 2 or higher of the IXLLOCK macro: RNAMELEN
- The following keywords and functions are supported by all versions, starting with version 4 of the IXLLOCK macro.
  - ADUPREQSEQNUM
  - REQVERSION

Specify the version of the parameter list that you want generated with the PLISTVER keyword. See Chapter 2, “Specifying a Macro Version Number,” on page 5 for considerations when specifying the version of the parameter list with PLISTVER.

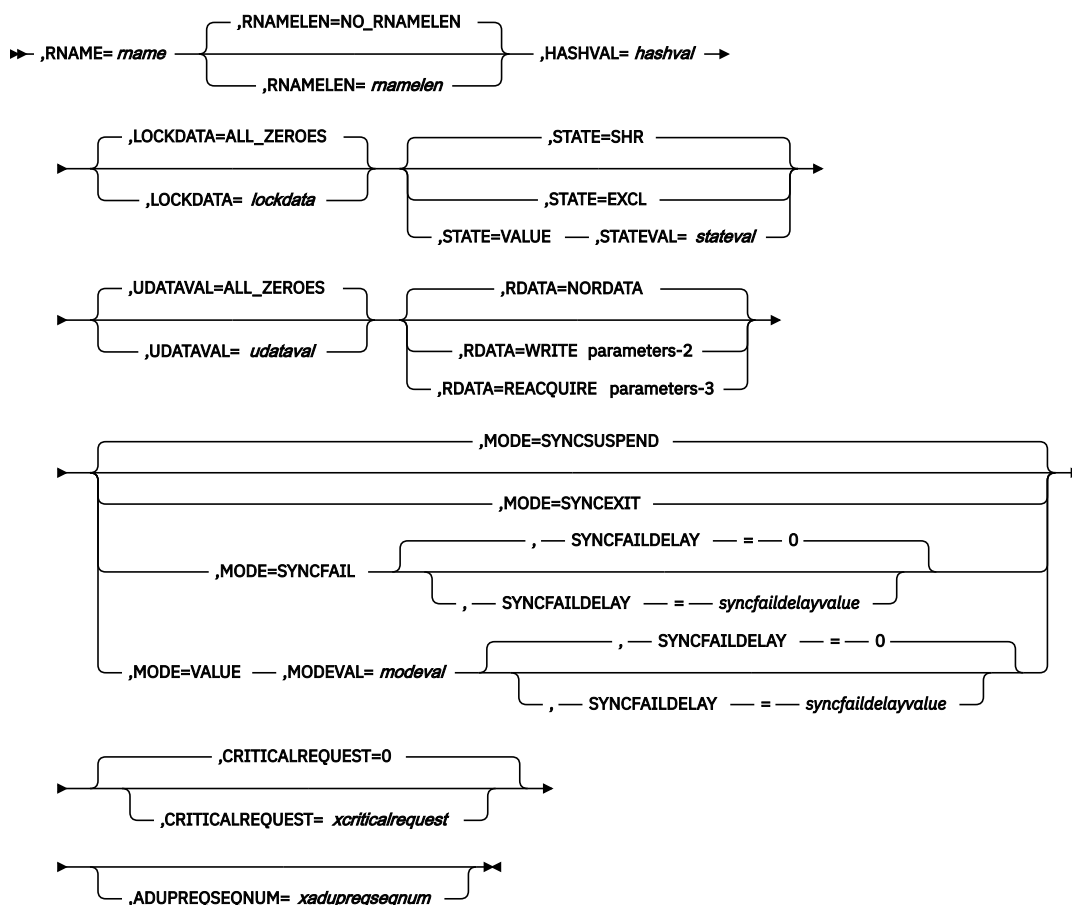
## Syntax diagram

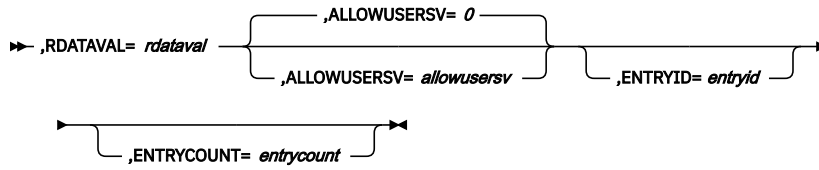
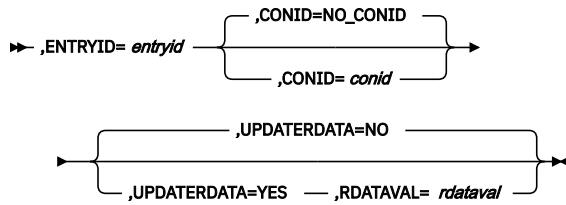
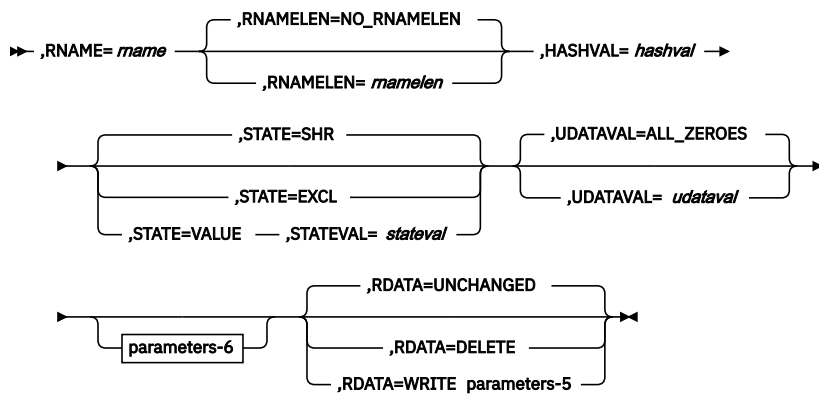
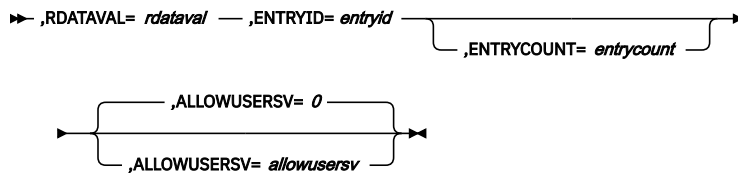
The syntax of the IXLLOCK macro is as follows:

### main diagram

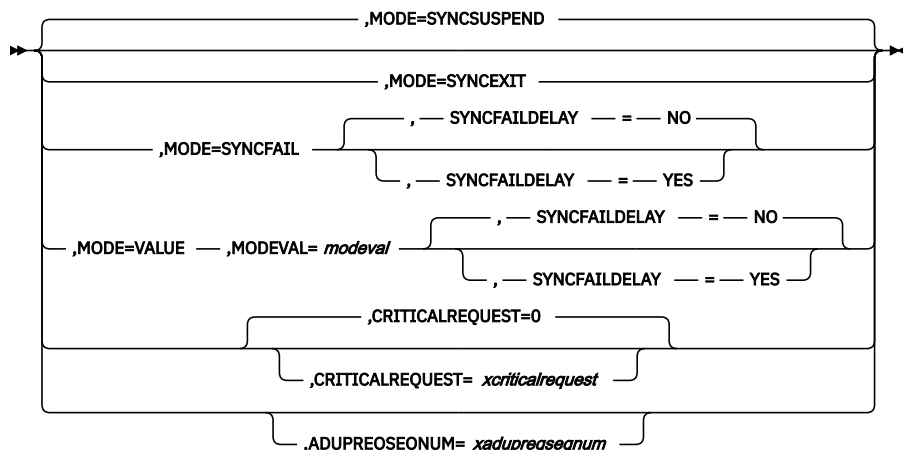


### parameters-1

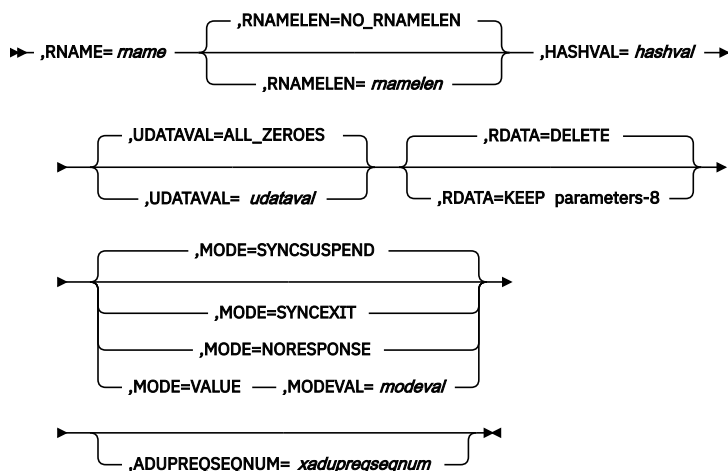


**parameters-2****parameters-3****parameters-4****parameters-5**

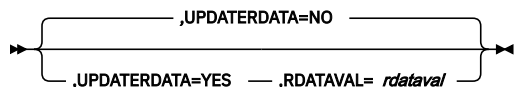
## parameters-6



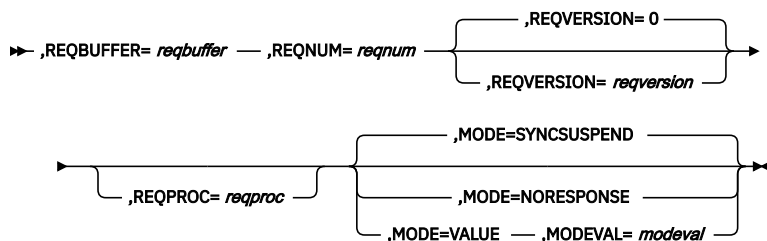
## parameters-7



## parameters-8



## parameters-9



## Parameter Descriptions

The parameter descriptions common to all IXLLOCK request types are listed in alphabetical order. Default values are underlined.



**CONTOKEN=contoken**

Use this input parameter to specify the CONNECT token that was returned in the answer area by the IXLCONN service. CONTOKEN uniquely identifies the user's connection to a lock structure.

**To Code:** Specify the RS-name or address (using a register from 2 to 12) of a 16-character input field that contains the CONNECT token returned in the answer area by the IXLCONN service.

**,MF=S**  
**,MF=(L,mfctrl)**  
**,MF=(L,mfctrl,mfattr)**  
**,MF=(L,mfctrl,0D)**  
**,MF=(M,mfctrl)**  
**,MF=(M,mfctrl,COMPLETE)**  
**,MF=(M,mfctrl,NOCHECK)**  
**,MF=(E,mfctrl)**  
**,MF=(E,mfctrl,COMPLETE)**  
**,MF=(E,mfctrl,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

**,mfctrl**

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

**,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

**,COMPLETE****,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

**COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if SMILE=*var* were an optional parameter and the default is SMILE=NO\_SMILE then it would not be documented. However, if the default was SMILE=:), then it would be documented because a value would be the default.

**NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

**,PLISTVER=IMPLIED\_VERSION**

**,PLISTVER=MAX**

**,PLISTVER=*plistver***

Use this input parameter to specify the version of the macro. See [“Understanding IXLLOCK version support”](#) on page 1341 for a description of the options available with PLISTVER.

**,REQUEST=OBTAIN**

**,REQUEST=ALTER**

**,REQUEST=RELEASE**

**,REQUEST=PROCESSMULT**

Use this input parameter to specify the type of operation requested.

#### **OBTAIN**

Use this input parameter to specify that the connected user is requesting to obtain ownership of the resource identified by the input resource name/hash value pair.

#### **ALTER**

Use this input parameter to request a change to one or more of the attributes of a resource that it currently owns. The ALTER option also may be used to replace a previous OBTAIN or ALTER request that is currently pending on the contention exit resource request queue with a more current request.

#### **RELEASE**

Use this input parameter to specify that the connected user is requesting to release ownership of the resource.

#### **PROCESSMULT**

Use this input parameter to specify that the connected user is requesting that multiple resource requests are to be processed. Each request is specified in a lock request block that the user builds in a storage area.

This request type is valid only for a structure allocated in a coupling facility of CFLEVEL=2 or higher.

**,RETCODE=*retcode***

Use this output parameter to specify the location in which the system is to copy the return code from GPR 15.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword to contain the return code.

**,RSNCODE=*rsncode***

Use this output parameter to specify the location in which the system is to copy the reason code from GPR 0.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword to contain the reason code.

## **REQUEST=OBTAIN**

The parameter descriptions for REQUEST=OBTAIN are listed in alphabetical order. Default values are underlined.

**,ADUPREQSEQNUM=*adupreqseqnum***

Use this output parameter to contain the asynchronous duplexing request sequence number that is assigned to this request. An asynchronous duplexing request sequence number is only generated for the request if:

- The request completes successfully, and
- The structure is duplexed by system-managed asynchronous duplexing when the request completes, and
- The request includes a record data update (write or reacquire), and
- The request might not have been committed in the secondary instance of the structure.

If an asynchronous duplexing request sequence number is generated and the request returns with a return code of `IxlRetCodeOk`, the asynchronous duplexing request sequence number is returned in the `ADUPREQSEQNUM` area. Otherwise the content of this output variable will be zero.

If the request returns with a return code of `IxlRetCodeWarning` and a reason code of `IxlRsnCodeAsynch`, and the request specifies `MODE=SYNCEXIT` or `MODEVAL=IXLMODESYNCEXIT`, any asynchronous duplexing request sequence number that is generated upon successful completion is presented to the user's `COMPLETE` exit.

The asynchronous duplexing request sequence number can be used on a subsequent invocation of `IXLADUPX` to ensure that the request is committed in the secondary instance of the asynchronously duplexed structure.

Specifying `ADUPREQSEQNUM` results in the generation of at least a version 4 parameter list. A version 4 parameter list requires system-managed asynchronous duplexing support. If a version 4 parameter list is used on a system that does not support it, the request is rejected for the unsupported parameter list version (`IxlRsnCodeBadVersion#`). Macro `IXCYQUAA` defines the `QuReqRfAsyncDuplex` bit in the `QuReqFeatures` string that can be used to test for system-managed asynchronous duplexing support. Use `IXCQUERY REQINFO=FEATURES` to get the `QuReqFeatures` string.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-character output field to contain the asynchronous duplexing request sequence number that is assigned to this request.

**,ALLOWUSERSV=0**

**,ALLOWUSERSV=*allowusersv***

Use this input parameter to indicate whether to allow a request that attempts to create a record data entry to proceed if the resulting percentage of free entries at the completion of the request would be less than the established percent entry reserved threshold (if any) for the structure.

The `ALLOWUSERSV` parameter is meaningful only when the `PCTENTRYRSV` parameter is used on an `IXLCONN` service invocation to establish a non-zero percent entry reserved threshold for the lock structure and the lock structure is allocated in a `CFLEVEL=25` or higher coupling facility.

A value of 0 (`IxlLockAllowUseRsvNo`) indicates that if request processing creates an entry and the resulting percentage of free entries at the completion of the request would be less than the established percent entry reserved threshold for the structure, the request is not permitted to create the record data entry and the request will fail with a return code of `IxlRetCodeEnvError`, reason code of `IxlRsnCodeRtFull`.

A value of 1 (`IxlLockAllowUseRsvYes`) indicates that the established percent entry reserved threshold for the structure should be ignored for this request and an entry should be created as long as there is a free record data entry to satisfy the request. Use `IxlLockAllowUseRsvYes` as the value for `AllowUseRsv` during application and connector recovery scenarios when using reserve entries to create a record data entry is deemed necessary.

Any other specified value will have the same behavior as specifying a value of 0 (`IxlLockAllowUseRsvNo`).

DEFAULT: 0

**,CONID=NO\_CONID**

**,CONID=*conid***

Use this input parameter to specify the connection identifier that indicates the connection from which the record data entry is being reacquired. If the record data entry designated by `ENTRYID` is not associated with the connection specified by `CONID`, the `IXLLOCK` request will fail. When a record data entry is successfully reacquired, it becomes associated with the reacquiring connected user.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a one-byte input field that contains the connection identifier associated with the record data entry to be reacquired.

**,CRITICALREQUEST=0**

**,CRITICALREQUEST=*criticalrequest***

Use this input parameter to indicate whether monitoring of storage is to be honored for this request. Valid values are 0 (or `IxllockCriticalRequestNo`) and 1 (or `IxllockCriticalRequestYes`). The

CRITICALREQUEST option only has meaning when MONITORSTORAGE=1 (IxIconnMonitorStorageYes) is also specified on the IXLCONN request.

A value of 0 (IxIlockCriticalRequestNo) specifies to monitor storage usage whenever MONITORSTORAGE=1 (IxIconnMonitorStorageYes) is also specified on the IXLCONN request. When the amount of inuse storage reaches a preestablished threshold, the request is rejected with a return code of IXLRETCODEENVERROR and a reason code of IXLSNCDERESOURCESCONSTRAINED.

A value of 1 (IxIlockCriticalRequestYes) specifies not to monitor storage usage. Preestablished thresholds for the amount of inuse storage is ignored for this request. Only when dataspace storage for this request becomes exhausted will an abend X'026' be issued by the XES storage manager.

Any other specified value will have the same behavior as specifying a value of 0 (IxIlockCriticalRequestNo).

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a one-byte input field that contains the value indicating whether monitoring of storage is to be honored.

#### **,ENTRYCOUNT=entrycount**

Use this output parameter to contain the number of record data entries in the structure that are currently in-use upon successful, synchronous completion of the request. This value is analogous to the IXLYAMDSTRL\_LSEC field that is returned by the XES Accounting and Measurement service, IXLMG. The value returned in the ENTRYCOUNT field can be used in conjunction with the value indicating the maximum number of record data entries supported by the allocated lock structure to monitor structure capacity and anticipate "structure full" conditions. The maximum number of record data entries allowed is returned in the CONALOCKMAXRECORDELEMENTS field of the Connect answer area. You can also use the IXLMG macro to retrieve the value, which will be returned in the IXLYAMDSTRL\_LSEC field of the IXLMG answer area.

Note that if the request is unsuccessful or being processed asynchronously (in which case the results are presented to the user's complete exit), the contents of ENTRYCOUNT are not valid.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword output field to contain the number of record data entries in the structure that are currently in-use.

#### **,ENTRYID=entryid**

Use this input/output parameter to specify the unique identifier assigned to the record data entry.

- When RDATA=WRITE, ENTRYID is an output parameter to contain the unique identifier assigned to the record data entry upon successful, synchronous request completion. If the request is unsuccessful or being processed asynchronously (in which case the results are presented to the user's complete exit), the contents of ENTRYID are not valid.
- When RDATA=REACQUIRE, ENTRYID is an input parameter to contain the unique identifier of the record data entry to be reacquired. The entry with this identifier must already exist.

ENTRYID is a required keyword when RDATA=REACQUIRE or RDATA=WRITE is specified.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 12-character input/output field to contain the unique identifier assigned to the record data entry.

#### **,HASHVAL=hashval**

Use this input parameter to specify a hash value associated with the resource name. The hash value along with the resource name serves to fully qualify an IXLLOCK resource. The method of producing the hash value is completely at the discretion of the connected user. Typically, the value provided for this keyword is the output of a user-defined hashing algorithm that receives a resource name as input.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an input fullword field that contains a hash value associated with the resource name.

#### **,LOCKDATA=ALL\_ZEROES**

#### **,LOCKDATA=lockdata**

Use this input parameter to specify user-defined data to be associated with this resource. The contents of LOCKDATA are at the discretion of the user and have no meaning to the system. The associated LOCKDATA is presented to this connected user's complete exit if the OBTAIN request is

processed asynchronously. The LOCKDATA is also presented to the complete and notify exits to inform the user of subsequent updates (such as the completion of requests to alter this resource) and status regarding the owned resource.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an eight-character input field that contains connected user-defined data to be associated with this resource.

**,MODE=SYNCSUSPEND**

**,MODE=SYNCEXIT**

**,MODE=SYNCFAIL**

**,MODE=VALUE**

Use this input parameter to specify how the request should be processed if it is not able to be serviced immediately. If the request is able to be processed immediately, the MODE keyword is ignored and control is returned to the caller with all information regarding the completed request.

### **SYNCSUSPEND**

Indicates that the OBTAIN request is to be handled synchronously. The caller receives control back only when the request is complete. If necessary, the caller is suspended until the request is complete.

### **SYNCEXIT**

Indicates that the OBTAIN request is to be handled asynchronously. Return and reason codes are returned to the caller indicating that the request will be processed in this manner. Request completion is reported through the connected user's complete exit. The user-specified complete exit might be given control before control returns to the next sequential instruction after the connected user's IXLLOCK request.

### **SYNCFAIL**

Indicates that the OBTAIN request is to be canceled if it cannot be processed without delay. Return and reason codes are returned to the caller indicating that the request has been canceled.

### **SYNCFAILDELAY=0, SYNCFAILDELAY=syncfaildelayvalue**

When you specify MODE=SYNCFAIL, use this optional keyword to handle delays for XES latch serialization when the XES normal latch is being held.

0 indicates that the delay is to be canceled and is the default; the return and reason code provide information.

*syncfaildelayvalue* specifies the value represented in IXLYCON, to control which delays to tolerate by the MODE=SYNCFAIL request. This option is ignored for all other types of requests.

The valid IXLYCON constants for IXLLOCK requests are:

- IXLSYNCFAILDELAYFORLATCHNO, which specifies that if the request encounters delays for internal XES serialization, it is canceled with the IxlRetCodeEnvError return code and IxlRsnCodeNoDelay reason code.
- IXLSYNCFAILDELAYFORLATCHYES, which specifies that if the request encounters delays for normal internal XES serialization, it is NOT canceled with the IxlRetCodeEnvError return code and IxlRsnCodeNoDelay reason code. Note that this does not include serialization on behalf of serialized connection recovery processing.

Note that IXLCONN ALLOWAUTO=YES, SUSPEND=YES behavior is not changed as part of the SYNCFAILDELAY specification. If the system receives an IXLLOCK OBTAIN or ALTER SYNCFAIL request while the target structure is delayed because of system-managed rebuild processing, the request is not deferred regardless of SYNCFAILDELAY. The system will fail the request with the IxlRsnCodeNoDelay reason code.

If you specify a value other than one of the valid IXLYCON constants for an IXLLOCK request, the request fails with reason code IxlRsnCodeBadSyncFailDelay

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a one-byte input field that contains a value that indicates the desired SyncFailDelay behavior for the request to be processed.

**VALUE**

Indicates that the contents of MODEVAL are to be used to specify how the request is to be processed if it cannot be serviced immediately.

**,MODEVAL=modeval**

Use this input parameter to specify the value, as represented in IXLYCON, of the desired mode in which the request is to be processed.

The valid IXLYCON constants for an OBTAIN request are:

- IXLMODESYNCSUSPEND
- IXLMODESYNCEXIT
- IXLMODESYNCFAIL

If you specify a value other than one of the IXLYCON constants that is valid for an OBTAIN request, the IXLLOCK request fails with reason code IXLRNCOEADMODEVAL.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a one-byte input field that contains a value indicating the desired mode in which the request is to be processed.

**SYNCFAILDELAY=0,SYNCFAILDELAY=syncfaildelayvalue**

When you specify MODE=VALUE,MODEVAL=modeval, use this optional keyword to handle delays for XES latch serialization when the XES normal latch is being held.

0 indicates that the delay is to be canceled and is the default; the return and reason code provide information.

*syncfaildelayvalue* specifies the value represented in IXLYCON, to control which delays to tolerate by the MODE=VALUE,MODEVAL=modeval request. This option is ignored for all other types of requests.

The valid IXLYCON constants for IXLLOCK requests are:

- IXLSYNCFAILDELAYFORLATCHNO, which specifies that if the request encounters delays for internal XES serialization, it is canceled with the IxlRetCodeEnvError return code and IxlRsnCodeNoDelay reason code.
- IXLSYNCFAILDELAYFORLATCHYES, which specifies that if the request encounters delays for normal internal XES serialization, it is NOT canceled with the IxlRetCodeEnvError return code and IxlRsnCodeNoDelay reason code. Note that this does not include serialization on behalf of serialized connection recovery processing.

Note that IXLCONN ALLOWAUTO=YES, SUSPEND=YES behavior is not changed as part of the SYNCFAILDELAY specification. If the system receives an IXLLOCK OBTAIN or ALTER SYNCFAIL request while the target structure is delayed because of system-managed rebuild processing, the request is not deferred regardless of SYNCFAILDELAY. The system will fail the request with the IxlRsnCodeNoDelay reason code.

If you specify a value other than one of the valid IXLYCON constants for an IXLLOCK request, the request fails with reason code IxlRsnCodeBadSyncFailDelay

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a one-byte input field that contains a value that indicates the desired SyncFailDelay behavior for the request to be processed.

**,RDATA=NORDATA****,RDATA=WRITE****,RDATA=REACQUIRE**

Use this input parameter to specify the record data operation, if any, that is to be performed as part of obtaining the specified resource.

**NORDATA**

Indicates that no record data entry is to be allocated and associated with the specified resource.

**WRITE**

Indicates that a record data entry is to be allocated and to be associated with the owned resource.

**REACQUIRE**

Indicates that a record data entry identified by ENTRYID is to be reacquired.

The use of the REACQUIRE option is intended to aid in recovery of resources during recovery scenarios. For example,

- Upon reconnecting, a previously failed-persistent connected user of locking services can re-obtain resources that were held by its previous instance and reacquire the existing record data entries to be associated with the new instances of ownership. The user could potentially use the UPDATERDATA suboption to update the contents of the reacquired record data entries to reflect updated state information.
- A connected user of locking services fails such that the related surviving users want to recover the resources held by the failing user. The survivors might want to obtain the specified resources while reacquiring the associated record data entries from the failed connector. The surviving connectors could potentially exploit the CONID suboption to coordinate their processing.

**,RDATAVAL=*rdataval***

Use this input/output parameter to specify the user-defined data to be written to the record data entry.

RDATAVAL is a required parameter when RDATA=WRITE is specified. The entry identifier of the record data entry is returned to the connected user in the ENTRYID field.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 64-character input/output field that contains user-defined data to be written to the record data entry. The contents of RDATAVAL are at the discretion of the connected user and have no meaning to the system. The contention exit may modify the value that is to be written to the record data entry as part of granting the request.

If completion of the OBTAIN request is presented to the user synchronously, the input variable will contain the resultant record data value.

**,RNAME=*rname***

Use this input parameter to specify the resource name for which the request is to be processed. The resource name and length along with the hash value serve to fully qualify an IXLLOCK request.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the character input field that contains the resource name for which the request is being processed.

**,RNAMELEN=NO *RNAMELEN*****,RNAMELEN=*rnamelen***

Use this input parameter to specify the length of the resource name identified by RNAME. The resource name length attribute is established on the IXLCONN invocation. The resource name and length along with the hash value serve to fully qualify an IXLLOCK request.

RNAMELEN is valid only when variable-length resource names are in effect for the lock structure. The value that you specify for RNAMELEN must be between 1 and 300 inclusive.

- If you specify RNAMELEN for a lock structure that does not have the variable-length name attribute, the system rejects the request with reason code IXLRSCODENOVARRNAME.
- If you specify a value for RNAMELEN that is different from the actual length of RNAME, the system uses the RNAMELEN value as the length of RNAME.
- If you do not specify RNAMELEN, the length of the resource name defaults to 64 bytes.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword input field that contains the resource name length of the resource identified by RNAME.

**,STATE=SHR****,STATE=EXCL****,STATE=VALUE**

Use this input parameter to specify the state in which the connected user is requesting to own the resource.

Note that the OBTAIN request may be granted by the contention exit with a STATE different from what was requested. If a user's protocol is exploiting the capability for the contention exit to grant requests with a STATE different than requested, it is recommended that STATE=VALUE be specified for this option to ensure that an output variable will be available to contain the resultant state when request completion is reported synchronously.

**SHR**

Specifies a shared state.

**EXCL**

Specifies an exclusive state.

**VALUE**

Specifies that the contents of STATEVAL are to be used to identify the ownership state.

**,STATEVAL=*stateval***

Use this input/output parameter to specify the value, as represented in IXLYCON, of the desired ownership state.

Note that if a value other than the IXLYCON constants for shared or exclusive is specified, the resource will be requested in the default STATE of share.

Upon successful, synchronous request completion, the input variable will contain the state in which ownership of the resource was granted.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a one-byte input field that contains a value indicating the desired ownership state.

**,UDATAVAL=ALL\_ZEROES****,UDATAVAL=*udataval***

Use this input/output parameter to specify user-defined data to be associated with the resource. The contents of UDATAVAL are at the discretion of the connected user and have no meaning to the system. UDATAVAL is presented to the complete, notify, and contention exits when driven.

Note that the contents of UDATAVAL may be modified by the contention exit as part of granting or denying the request. The contents of UDATAVAL have no meaning to the system.

If request completion is reported to the connected user synchronously, UDATAVAL, if specified, will contain the resultant user data value.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 64-character input/output field that contains user-defined data to be associated with the resource.

**,UPDATERDATA=NO****,UPDATERDATA=YES**

Use this input parameter to specify whether or not to update the contents of the record data entry at the time it is reacquired.

**NO**

Indicates that the data in the record data entry is not to be updated.

**YES**

Indicates that the data in the record data entry is to be updated with the data that is specified by RDATAVAL.

RDATAVAL is a required keyword when UPDATERDATA=YES is specified.

**REQUEST=ALTER**

The parameter descriptions for REQUEST=ALTER are listed in alphabetical order. Default values are underlined.

**,ADUPREQSEQNUM=adupreqseqnum**

Use this output parameter to contain the asynchronous duplexing request sequence number that is assigned to this request. An asynchronous duplexing request sequence number is only generated for the request if:



- The request completes successfully, and
- The structure is duplexed by system-managed asynchronous duplexing when the request completes, and
- The request includes a record data update (write or delete), and
- The request might not have been committed in the secondary instance of the structure.

If an asynchronous duplexing request sequence number is generated and the request returns with a return code of `IxlRetCodeOk`, the asynchronous duplexing request sequence number is returned in the `ADUPREQSEQNUM` area. Otherwise, the content of this output variable is zero.

If the request returns with a return code of `IxlRetCodeWarning` and a reason code of `IxlRsnCodeAsynch`, and the request specifies `MODE=SYNCEXIT` or `MODEVAL=IXLMODESYNCEXIT`, any asynchronous duplexing request sequence number that is generated upon successful completion is presented to the user's `COMPLETE` exit.

The asynchronous duplexing request sequence number can be used on a subsequent invocation of `IXLADUPX` to ensure that the request is committed in the secondary instance of the asynchronously duplexed structure.

Specifying `ADUPREQSEQNUM` results in the generation of at least a version 4 parameter list. A version 4 parameter list requires system-managed asynchronous duplexing support. If a version 4 parameter list is used on a system that does not support it, the request is rejected for the unsupported parameter list version (`IxlRsnCodeBadVersion#`). Macro `IXCYQUAA` defines the `QuReqRfAsynchDuplex` bit in the `QuReqFeatures` string that can be used to test for system-managed asynchronous duplexing support. Use `IXCQUERY REQINFO=FEATURES` to get the `QuReqFeatures` string.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-character output field to contain the asynchronous duplexing request sequence number that is assigned to this request.

**,ALLOWUSERSV=0**

**,ALLOWUSERSV=allowusersv**

Use this input parameter to indicate whether to allow a request that attempts to create a record data entry to proceed if the resulting percentage of free entries at the completion of the request would be less than the established percent entry reserved threshold (if any) for the structure.

The `ALLOWUSERSV` parameter is meaningful only when the `PCTENTRYRSV` parameter is used on an `IXLCONN` service invocation to establish a non-zero percent entry reserved threshold for the lock structure and the lock structure is allocated in a `CFLEVEL=25` or higher coupling facility.

A value of 0 (`IxlLockAllowUserRsvNo`) indicates that if request processing creates an entry and the resulting percentage of free entries at the completion of the request would be less than the established percent entry reserved threshold for the structure, the request is not permitted to create the record data entry and the request will fail with a return code of `IxlRetCodeEnvError`, reason code of `IxlRsnCodeRtFull`.

A value of 1 (`IxlLockAllowUserRsvYes`) indicates that the established percent entry reserved threshold for the structure should be ignored for this request and an entry should be created as long as there is a free record data entry to satisfy the request. Use `IxlLockAllowUserRsvYes` as the value for `AllowUserRsv` during application and connector recovery scenarios when using reserve entries to create a record data entry is deemed necessary.

Any other specified value will have the same behavior as specifying a value of 0 (`IxlLockAllowUserRsvNo`).

DEFAULT: 0

**,CRITICALREQUEST=0**

**,CRITICALREQUEST=criticalrequest**

Use this input parameter to indicate whether monitoring of storage is to be honored for this request. Valid values are 0 (or `IxllockCriticalRequestNo`) and 1 (or `IxllockCriticalRequestYes`). The `CRITICALREQUEST` option only has meaning when `MONITORSTORAGE=1` (`IxlconnMonitorStorageYes`) is also specified on the `IXLCONN` request.

A value of 0 (IxlockCriticalRequestNo) specifies to monitor storage usage whenever MONITORSTORAGE=1 (IxconnMonitorStorageYes) is also specified on the IXLCONN request. When the amount of inuse storage reaches a preestablished threshold, the request will be rejected with a return code of IXLRETCODEENVEERROR and a reason code of IXLRSNCODERESOURCESCONSTRAINED.

A value of 1 (IxlockCriticalRequestYes) specifies not to monitor storage usage. Preestablished thresholds for the amount of inuse storage is ignored for this request. Only when dataspace storage for this request becomes exhausted will an abend X'026' be issued by the XES storage manager.

Any other specified value will have the same behavior as specifying a value of 0 (IxlockCriticalRequestNo).

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a one-byte input field that contains the value indicating whether monitoring of storage is to be honored.

**,ENTRYCOUNT=entrycount**

Use this output parameter to contain the number of record data entries in the structure that are currently in-use upon successful, synchronous completion of the request. This value is analogous to the IXLYAMDSTRL\_LSEC field that is returned by the XES Accounting and Measurement service, IXLMG. The value returned in the ENTRYCOUNT field can be used in conjunction with the value indicating the maximum number of record data entries supported by the allocated lock structure to monitor structure capacity and anticipate "structure full" conditions. The maximum number of record data entries allowed is returned in the CONALOCKMAXRECORDELEMENTS field of the Connect answer area. You can also use the IXLMG macro to retrieve the value, which will be returned in the IXLYAMDSTRL\_LSEC field of the IXLMG answer area.

Note that if the request is unsuccessful or being processed asynchronously (in which case the results are presented to the user's complete exit), the contents of ENTRYCOUNT are not valid.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword output field to contain the number of record data entries in the structure that are currently in-use.

**,ENTRYID=entryid**

Use this output parameter to specify the unique identifier assigned to the record data entry upon successful, synchronous request completion. If a new record data entry was allocated by this request, the identifier indicates the newly allocated entry. If the request resulted in the update of an existing record data entry that had been associated with this resource through a previous OBTAIN or ALTER request, the identifier indicates the updated entry. If the request is unsuccessful or being processed asynchronously (in which case the results are presented to the user's complete exit), the contents of ENTRYID are not valid.

ENTRYID is a required keyword when RDATA=WRITE is specified.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 12-character output field to contain the unique identifier that is assigned to the record data entry.

**,HASHVAL=hashval**

Use this input parameter to specify a hash value that is associated with the resource name. The hash value along with the resource name serves to fully qualify an IXLLOCK resource. The method of producing the hash value is completely at the discretion of the connected user. Typically, the value provided for this keyword is the output of a user-defined hashing algorithm that receives a resource name as input.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an input fullword field that contains a hash value that is associated with the resource name.

**,MODE=SYNCSUSPEND**

**,MODE=SYNCEXIT**

**,MODE=SYNCFAIL**

**,MODE=VALUE**

Use this input parameter to specify how the request should be processed if it is not able to be serviced immediately. If the request is able to be processed immediately, the MODE keyword is ignored and control is returned to the caller with all information regarding the completed request.

**SYNCSUSPEND**

Indicates that the ALTER request is to be handled synchronously. The caller receives control back only when the request is complete. If necessary, the caller is suspended until the request completes.

**SYNCEXIT**

Indicates that the ALTER request is to be handled asynchronously. Return and reason codes are returned to the caller indicating that the request will be processed in this manner. Request completion is reported through the connected user's complete exit. The user-specified complete exit may be given control before control returns to the next sequential instruction after the connected user's IXLLOCK request.

**SYNCFAIL**

Indicates that the ALTER request is to be canceled if it cannot be processed without delay. Return and reason codes are returned to the caller indicating that the request has been canceled.

**SYNCFAILDELAY=0, SYNCFAILDELAY=syncfaildelayvalue**

When you specify MODE=SYNCFAIL, use this optional keyword to handle delays for XES latch serialization when the XES normal latch is being held.

0 indicates that the delay is to be canceled and is the default; the return and reason code provide information.

*syncfaildelayvalue* specifies the value represented in IXLYCON, to control which delays to tolerate by the MODE=SYNCFAIL request. This option is ignored for all other types of requests.

The valid IXLYCON constants for IXLLOCK requests are:

- IXLSYNCFAILDELAYFORLATCHNO, which specifies that if the request encounters delays for internal XES serialization, it is canceled with the IxlRetCodeEnvError return code and IxlRsnCodeNoDelay reason code.
- IXLSYNCFAILDELAYFORLATCHYES, which specifies that if the request encounters delays for normal internal XES serialization, it is NOT canceled with the IxlRetCodeEnvError return code and IxlRsnCodeNoDelay reason code. Note that this does not include serialization on behalf of serialized connection recovery processing.

Note that IXLCONN ALLOWAUTO=YES, SUSPEND=YES behavior is not changed as part of the SYNCFAILDELAY specification. If the system receives an IXLLOCK OBTAIN or ALTER SYNCFAIL request while the target structure is delayed because of system-managed rebuild processing, the request is not deferred regardless of SYNCFAILDELAY. The system will fail the request with the IxlRsnCodeNoDelay reason code.

If you specify a value other than one of the valid IXLYCON constants for an IXLLOCK request, the request fails with reason code IxlRsnCodeBadSyncFailDelay

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a one-byte input field that contains a value that indicates the desired SyncFailDelay behavior for the request to be processed.

**VALUE**

Indicates that the contents of MODEVAL are to be used to specify how the request is to be processed if it cannot be serviced immediately.

**,MODEVAL=modeval**

Use this input parameter to specify the value, as represented in IXLYCON, of the desired mode in which the request is to be processed.

The valid IXLYCON constants for an ALTER request are:

- IXLMODESYNCSUSPEND
- IXLMODESYNCEXIT
- IXLMODESYNCFAIL

If you specify a value other than one of the IXLYCON constants that is valid for an ALTER request, the IXLLOCK request fails with reason code IXLRSCODEBADMODEVAL.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a one-byte input field that contains a value indicating the desired mode in which the request is to be processed.

**SYNCFAILDELAY=0,SYNCFAILDELAY=*syncfaildelayvalue***

When you specify MODE=VALUE,MODEVAL=*modeval*, use this optional keyword to handle delays for XES latch serialization when the XES normal latch is being held.

0 indicates that the delay is to be canceled and is the default; the return and reason code provide information.

*syncfaildelayvalue* specifies the value represented in IXLYCON, to control which delays to tolerate by the MODE=VALUE,MODEVAL=*modeval* request. This option is ignored for all other types of requests.

The valid IXLYCON constants for IXLLOCK requests are:

- IXLSYNCFAILDELAYFORLATCHNO, which specifies that if the request encounters delays for internal XES serialization, it is canceled with the IxlRetCodeEnvError return code and IxlRsnCodeNoDelay reason code.
- IXLSYNCFAILDELAYFORLATCHYES, which specifies that if the request encounters delays for normal internal XES serialization, it is NOT canceled with the IxlRetCodeEnvError return code and IxlRsnCodeNoDelay reason code. Note that this does not include serialization on behalf of serialized connection recovery processing.

Note that IXLCONN ALLOWAUTO=YES, SUSPEND=YES behavior is not changed as part of the SYNCFAILDELAY specification. If the system receives an IXLLOCK OBTAIN or ALTER SYNCFAIL request while the target structure is delayed because of system-managed rebuild processing, the request is not deferred regardless of SYNCFAILDELAY. The system will fail the request with the IxlRsnCodeNoDelay reason code.

If you specify a value other than one of the valid IXLYCON constants for an IXLLOCK request, the request fails with reason code IxlRsnCodeBadSyncFailDelay

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a one-byte input field that contains a value that indicates the desired SyncFailDelay behavior for the request to be processed.

**,RDATA=UNCHANGED**

**,RDATA=DELETE**

**,RDATA=WRITE**

Use this input parameter to specify the record data operation, if any, that is to be performed as part of obtaining the specified resource.

Note that specification of the WRITE or DELETE options will result in a parameter error if the lock structure indicated by the input CONTOKEN does not provide recording capabilities.

**UNCHANGED**

Indicates that the record data entry associated with this resource, if any, is not to be changed.

**DELETE**

Indicates that the record data entry associated with this resource is to be deleted. If no record data entry is currently allocated and associated with this resource, then this keyword is ignored.

**WRITE**

Indicates that the record data entry identified by ENTRYID is to be updated with the data specified by the RDATAVAL keyword. If a record data entry is currently associated with the resource, its contents will be updated. Otherwise, a new entry will be allocated.

**,RDATAVAL=*rdataval***

Use this input/output parameter to specify the user-defined data to be written to the record data entry.

RDATAVAL is a required parameter when RDATA=WRITE is specified. The entry identifier of the record data entry is returned to the connected user in the ENTRYID field.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 64-character input/output field that contains user-defined data to be written to the record data entry. The contents of RDATAVAL are at the discretion of the connected user and have no meaning to the system. The contention exit may modify the value that is to be written to the record data entry as part of granting the request.

If completion of the OBTAIN request is presented to the user synchronously, the input variable will contain the resultant record data value.

**,RNAME=rname**

Use this input parameter to specify the resource name for which the request is to be processed. The resource name and length along with the hash value serve to fully qualify an IXLLOCK request.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the character input field that contains the resource name for which the request is being processed.

**,RNAMELEN=NO\_RNAMELEN**

**,RNAMELEN=rnamelen**

Use this input parameter to specify the length of the resource name identified by RNAME. The resource name length attribute is established on the IXLCONN invocation. The resource name and length along with the hash value serve to fully qualify an IXLLOCK request.

RNAMELEN is valid only when variable-length resource names are in effect for the lock structure. The value you specify for RNAMELEN must be between 1 and 300 inclusive.

- If you specify RNAMELEN for a lock structure that does not have the variable-length name attribute, the system rejects the request with reason code IXLSNCOENOVARRNAME.
- If you specify a value for RNAMELEN that is different from the actual length of RNAME, the system uses the RNAMELEN value as the length of RNAME.
- If you do not specify RNAMELEN, the length of the resource name defaults to 64 bytes.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword input field that contains the resource name length of the resource identified by RNAME.

**,STATE=SHR**

**,STATE=EXCL**

**,STATE=VALUE**

Use this input parameter to specify the state in which the connected user is requesting to own the resource.

Note that the ALTER request may be granted by the contention exit with a STATE different from what was requested. If a user's protocol is exploiting the capability for the contention exit to grant requests with a STATE different than requested, it is recommended that STATE=VALUE be specified for this option to ensure that an output variable will be available to contain the resultant state when request completion is reported synchronously.

**SHR**

Specifies a shared state.

**EXCL**

Specifies an exclusive state.

**VALUE**

Specifies that the contents of STATEVAL are to be used to identify the ownership state.

**,STATEVAL=stateval**

Use this input parameter to specify the value, as represented in IXLYCON, of the desired ownership state.

Note that if a value other than the IXLYCON constants for shared or exclusive is specified, the resource will be requested in the default STATE of share.

Upon successful, synchronous request completion, the input variable will contain the state in which ownership of the resource was granted.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a one-byte input field that contains a value indicating the desired ownership state.

**,UDATAVAL=ALL\_ZEROES**

**,UDATAVAL=udataval**

Use this input/output parameter to specify user-defined data to be associated with the resource. The contents of UDATAVAL are at the discretion of the connected user and have no meaning to the system. UDATAVAL is presented to the complete, notify, and contention exits when driven.

Note that the contents of UDATAVAL may be modified by the contention exit as part of granting or denying the request. The contents of UDATAVAL have no meaning to the system.

If request completion is reported to the connected user synchronously, UDATAVAL, if specified, will contain the resultant user data value.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 64-character input/output field that contains user-defined data to be associated with the resource.

## REQUEST=RELEASE

The parameter descriptions for REQUEST=RELEASE are listed in alphabetical order. Default values are underlined.

**,ADUPREQSEQNUM=adupreqseqnum**

Use this output parameter to contain the asynchronous duplexing request sequence number that is assigned to this request. An asynchronous duplexing request sequence number is only generated for the request if:

- The request completes successfully, and
- The structure is duplexed by system-managed asynchronous duplexing when the request completes, and
- The request includes a record data update (Delete, or Keep with UpdateRdata=Yes), and
- The request might not have been committed in the secondary instance of the structure.

If an asynchronous duplexing request sequence number is generated and the request returns with a return code of IxlRetCodeOk, the asynchronous duplexing request sequence number is returned in the ADUPREQSEQNUM area. Otherwise, the content of this output variable is zero.

If the request returns with a return code of IxlRetCodeWarning and a reason code of IxlRsnCodeAsynch, and the request specifies MODE=SYNCEXIT or MODEVAL=IXLMODESYNCEXIT, any asynchronous duplexing request sequence number that is generated upon successful completion is presented to the user's COMPLETE exit.

The asynchronous duplexing request sequence number can be used on a subsequent invocation of IXLADUPX to ensure that the request is committed in the secondary instance of the asynchronously duplexed structure.

Specifying ADUPREQSEQNUM results in the generation of at least a version 4 parameter list. A version 4 parameter list requires system-managed asynchronous duplexing support. If a version 4 parameter list is used on a system that does not support it, the request is rejected for the unsupported parameter list version (IxlRsnCodeBadVersion#). Macro IXCYQUAA defines the QuReqRfAsyncDuplex bit in the QuReqFeatures string that can be used to test for system-managed asynchronous duplexing support. Use IXCQUERY REQINFO=FEATURES to get the QuReqFeatures string.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-character output field to contain the asynchronous duplexing request sequence number that is assigned to this request.

**,HASHVAL=hashval**

Use this input parameter to specify a hash value that is associated with the resource name. The hash value along with the resource name serves to fully qualify an IXLLOCK resource. The method of producing the hash value is completely at the discretion of the connected user. Typically, the value provided for this keyword is the output of a user-defined hashing algorithm that receives a resource name as input.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an input fullword field that contains a hash value that is associated with the resource name.

**,MODE=SYNCSUSPEND**

**,MODE=SYNCEXIT**

**,MODE=NORESPONSE**

**,MODE=VALUE**

Use this input parameter to specify how the request should be processed if it is not able to be serviced immediately. If the request is able to be processed immediately, the MODE keyword is ignored and control is returned to the caller with all information regarding the completed request.

#### **SYNCSUSPEND**

Indicates that the RELEASE request be handled synchronously. The caller receives control back only when the request is complete. If necessary, the caller is suspended until the request completes.

#### **SYNCEXIT**

Indicates that the RELEASE request be handled asynchronously. Return and reason codes are returned to the caller indicating that the request will be processed in this manner. Request completion is reported through the connected user's complete exit. The user-specified complete exit may be given control before control returns to the next sequential instruction after the connected user's IXLLOCK request.

#### **NORESPONSE**

Indicates that the requester does not want to be informed when the RELEASE request is complete. Return and reason codes are returned to the caller indicating that the request will be processed asynchronously. However, the connected user's complete exit will not be invoked to report request completion.

#### **VALUE**

Indicates that the contents of MODEVAL are to be used to specify how the request is to be processed if it cannot be serviced immediately.

**,MODEVAL=modeval**

Use this input parameter to specify the value, as represented in IXLYCON, of the desired mode in which the request is to be processed.

The valid IXLYCON constants for a RELEASE request are:

- IXLMODESYNCSUSPEND
- IXLMODESYNCEXIT
- IXLMODENORESPONSE

If you specify a value other than one of the IXLYCON constants that is valid for a RELEASE request, the IXLLOCK request fails with reason code IXLRSNCODEBADMODEVAL.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a one-byte input field that contains a value indicating the desired mode in which the request is to be processed.

**,RDATA=DELETE**

**,RDATA=KEEP**

Use this input parameter to specify the record data operation, if any, that is to be performed as part of obtaining the specified resource.

#### **DELETE**

Indicates that the record data entry that is associated with this resource is to be deleted. If no record data entry is currently allocated and associated with this resource, then this keyword is ignored.

#### **KEEP**

Indicates that the record data entry that is associated with this resource is to be kept even though ownership of the resource is being relinquished.

Note that the record data entry that was kept might eventually be reassociated with a new resource through the REACQUIRE option of IXLLOCK REQUEST=OBTAIN or manipulated through

the XES record data service, IXLRT. If the KEEP option is specified and no associated record data entry exists, informational return and reason codes will be returned to the invoker.

**,RDATAVAL=*rdataval***

Use this input/output parameter to specify the user-defined data to be written to the record data entry. The contents of RDATAVAL are at the discretion of the connected user and have no meaning to the system. The contention exit may modify the value that is to be written to the record data entry as part of granting the request.

If completion of the OBTAIN request is presented to the user synchronously, the input variable will contain the resultant record data value.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 64-character input/output field that contains user-defined data to be written to the record data entry. The contents of RDATAVAL are at the discretion of the connected user and have no meaning to the system. The contention exit might modify the value that is to be written to the record data entry as part of granting the request.

**,RNAME=*rname***

Use this input parameter to specify the resource name for which the request is to be processed. The resource name and length along with the hash value serve to fully qualify an IXLLOCK request.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the character input field that contains the resource name for which the request is being processed.

**,RNAMELEN=NO *RNAMELEN***

**,RNAMELEN=*rnamelen***

Use this input parameter to specify the length of the resource name identified by RNAME. The resource name length attribute is established on the IXLCONN invocation. The resource name and length along with the hash value serve to fully qualify an IXLLOCK request.

RNAMELEN is valid only when variable-length resource names are in effect for the lock structure. The value you specify for RNAMELEN must be between 1 and 300 inclusive.

- If you specify RNAMELEN for a lock structure that does not have the variable-length name attribute, the system rejects the request with reason code IXLRSCODENOVARRNAME.
- If you specify a value for RNAMELEN that is different from the actual length of RNAME, the system uses the RNAMELEN value as the length of RNAME.
- If you do not specify RNAMELEN, the length of the resource name defaults to 64 bytes.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword input field that contains the resource name length of the resource identified by RNAME.

**,UDATAVAL=ALL *ZEROES***

**,UDATAVAL=*udataval***

Use this input/output parameter to specify user-defined data to be associated with the resource. The contents of UDATAVAL are at the discretion of the connected user and have no meaning to the system. UDATAVAL is presented to the complete, notify, and contention exits when driven.

Note that while a RELEASE request cannot be denied by the contention exit, the contents of the user data associated with the request might be modified.

If request completion is reported to the connected user synchronously, UDATAVAL, if specified, will contain the resultant user data value.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 64-character input/output field that contains user-defined data to be associated with the resource.

**,UPDATERDATA=NO**

**,UPDATERDATA=YES**

Use this input parameter to specify whether or not to update the data in the record data entry that has been kept.



**NO**

Indicates that the data in the record data entry is not to be updated.

**YES**

Indicates that the data in the record data entry is to be updated with the data that is specified by RDATAVAL. RDATAVAL is a required keyword when UPDATERDATA=YES is specified.

**REQUEST=PROCESSMULT**

This request type is valid only for a structure that is allocated in a coupling facility of CFLEVEL=2 or higher. The parameter descriptions for REQUEST=PROCESSMULT are listed in alphabetical order.

**,MODE=SYNCSUSPEND****,MODE=NORESPONSE****,MODE=VALUE**

Use this input parameter to specify how the request should be processed if it is not able to be serviced immediately. If the request is able to be processed immediately, the MODE keyword is ignored and control is returned to the invoker with all information regarding the completed request.

**SYNCSUSPEND**

Indicates that the PROCESSMULT request is to be handled synchronously. The caller receives control back only when the request is complete. If necessary, the caller is suspended until the request is complete.

**NORESPONSE**

Indicates that the requester does not want to be informed when the PROCESSMULT request is complete. Return and reason codes are returned to the caller indicating that the request will be processed asynchronously. However, the connected user's COMPLETE exit will not be invoked to report request completion.

**VALUE**

Indicates that the contents of MODEVAL are to be used to specify how the request is to be processed if it cannot be serviced immediately.

**,MODEVAL=modeval**

Use this input parameter to specify the value as represented in IXLYCON, of the desired mode in which the request is to be processed.

The valid IXLYCON constants for a PROCESSMULT request are as follows:

- IXLMODESYNCSUSPEND
- IXLMODENORESPONSE

If you specify a value other than one of the IXLYCON constants that is valid for a PROCESSMULT request, the IXLLOCK request fails with reason code IXLRSNCODEBADMODEVAL.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a one-byte input field that contains a value indicating the desired mode in which the request is to be processed.

**,REQBUFFER=reqbuffer**

Use this input parameter to specify the virtual storage area that contains from 1 to 128 lock request blocks that correspond to resource requests. The macro IXLYLRB maps each lock request block. The virtual storage area must reside in fixed or disabled reference storage and be addressable from the caller's primary address space.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an input buffer that contains the lock request blocks to be processed. The length of the buffer is determined by the number of lock request blocks that it contains.

**,REQNUM=reqnum**

Use this input parameter to specify the number of lock request blocks that are in the area that is identified by REQBUFFER. The value of REQNUM must be in the range of 1 to 128.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a halfword input field that contains the number of lock request blocks in the REQBUFFER area.

**,REQPROC=reqproc**

Use this output parameter to contain the number of lock request blocks that the system processed.

When the PROCESSMULT request is completed synchronously, there are two situations:

- If all lock request blocks were processed, this value is equal to the value specified for the REQNUM keyword.
- If an error occurred so that only a partial set of lock request blocks were processed, this value indicates the number of lock request blocks that were processed before the error occurred. The LRBs that follow the LRB designated by REQPROC are in an indeterminate state.

When the PROCESSMULT request is completed asynchronously, this value is zero.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a halfword output field to contain the number of lock request blocks that the system processed.

**,REQVERSION=0****,REQVERSION=reqversion**

Use this input parameter to specify the version number of the Lock Request Blocks (LRB) in the REQBUFFER list.

REQVERSION values must be in the range of 0 to 1. A value of 0 indicates that the LRBs are mapped by LRB\_Release\_Ver0. A value of 1 indicates that the LRBs are mapped by LRB\_Release\_Ver1. Consult the IXLYLRB mapping macro for information on the data to be returned for each version.

Specifying REQVERSION will result in the generation of at least a version 4 parameter list. A version 4 parameter list requires system-managed asynchronous duplexing support. If a version 4 parameter list is used on a system that does not support it, the request is rejected for the unsupported parameter list version (IxIRsnCodeBadVersion#). Macro IXCYQUAA defines the QuReqRfAsyncDuplex bit in the QuReqFeatures string that can be used to test for system-managed asynchronous duplexing support. Use IXCQUERY REQINFO=FEATURES to get the QuReqFeatures string.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a one-byte input field that contains a value indicating the version number of the Lock Request Blocks in the REQBUFFER list.

## ABEND Codes

---

None.

## Return and reason codes

---

When the IXLLOCK macro returns control to your program:

- GPR 15 (and *retcode*, if you coded RETCODE) contains a return code.
- GPR 0 (and *rsncode*, if you coded RSNCODE) contains a reason code if applicable.

Macro IXLYCON provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

**0**

IXLRETCODEOK

**4**

IXLRETCODEWARNING

**8**

IXLRETCODEPARMERROR

**C**

IXLRETCODEENVERROR

**10**

IXLRETCODECOMPERROR

The following table identifies the hexadecimal return and reason codes and the equate symbol associated with each reason code.

Table 75. Return and reason codes for the IXLLOCK macro		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0	None.	<p><b>Meaning:</b> IXLLOCK request successful.</p> <ul style="list-style-type: none"> <li>For an OBTAIN or ALTER request, ownership has been granted in the requested state. User data is the same as requested.</li> <li>For a RELEASE request specifying MODE=SYNCSUSPEND or MODE=SYNCEXIT, processing is complete.</li> <li>For a RELEASE request specifying MODE=NORESPONSE, the request has been accepted.</li> <li>For a PROCESSMULT request, all LRB processing is complete. Individual return and reason codes are returned for each LRB.</li> </ul> <p><b>Action:</b> None.</p>
4	xxxx0402	<p><b>Equate Symbol:</b> IXLRSNCODEASYNCH</p> <p><b>Meaning:</b> Request will be completed asynchronously. If MODE=SYNCEXIT or MODEVAL=IXLMODESYNCEXIT was specified, the system reports request completion through the connected user's COMPLETE exit. For IXLLOCK RELEASE and IXLLOCK PROCESSMULT requests that specified MODE=NORESPONSE or MODEVAL=IXLMODENORESPONSE, request completion is not reported to the connected user.</p> <p><b>Action:</b> If you specified MODE=SYNCEXIT, the user's complete exit will be given control when the request is complete.</p>
4	xxxx0418	<p><b>Equate Symbol:</b> IXLRSNCODENOELEMENTTOKEEP</p> <p><b>Meaning:</b> The IXLLOCK RELEASE request is successful. The user specified to keep the record data entry associated with the resource, but there was no data entry to keep.</p> <p><b>Action:</b> None expected. However, if you were expecting a record data entry to be present, determine why it was not.</p>
4	xxxx0419	<p><b>Equate Symbol:</b> IXLRSNCODENOUUPDATEONKEEP</p> <p><b>Meaning:</b> The IXLLOCK RELEASE request is successful. The user specified to keep the associated record data entry and update its contents. The element was kept, but its contents were unable to be updated.</p> <p><b>Action:</b> Consider using IXLRT to update the contents of the record data entry.</p>
4	xxxx0424	<p><b>Equate Symbol:</b> IXLRSNCODENODELETEONRELEASE</p> <p><b>Meaning:</b> The IXLLOCK RELEASE request is successful. The user specified to delete the associated record data entry, but the entry was unable to be deleted.</p> <p><b>Action:</b> None expected.</p>

Table 75. Return and reason codes for the IXLLOCK macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0801	<p><b>Equate Symbol:</b> IXLRNCODEBADPARMLIST</p> <p><b>Meaning:</b> Program error. The parameter list for this request is not addressable. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Verify that:</p> <ul style="list-style-type: none"> <li>• The parameter list address is uncorrupted.</li> <li>• The parameter list is addressable in the caller's primary address space.</li> <li>• If you are issuing this macro while disabled, the parameter list resides in either page-fixed or DREF storage.</li> <li>• If you are issuing this macro in AR-mode and you specified the parameter list address using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If you are issuing this macro in AR-mode, SYSSTATE ASCENV=AR must be issued before issuing this macro.</li> </ul>
8	xxxx0804	<p><b>Equate Symbol:</b> IXLRNCODEBADVERSION#</p> <p><b>Meaning:</b> Program error. The version number in the parameter is not valid. This usually indicates that the level of the macro is incompatible with the level of the XES service code.</p> <p><b>Action:</b> Verify that your program was assembled with the correct macro library for the release of MVS that your program is running on.</p>
8	xxxx0807	<p><b>Equate Symbol:</b> IXLRNCODENOTENABLED</p> <p><b>Meaning:</b> Caller was not enabled.</p> <p><b>Action:</b> Verify that the program is enabled for I/O and external interrupts.</p>
8	xxxx080A	<p><b>Equate Symbol:</b> IXLRNCODEBADCONTOKEN</p> <p><b>Meaning:</b> Program error. The input contoken is not valid. The contoken may no longer be valid for one of the following reasons: disconnect has occurred, EOT of the connector's task, input contoken is not the contoken returned from IXLCONN, the request was issued outside the connector's address space, or the contoken has been invalidated for rebuild.</p> <p><b>Action:</b> Verify that the CONTOKEN value specified is valid and for the correct structure.</p>
8	xxxx0810	<p><b>Equate Symbol:</b> IXLRNCODERESOURCENOTFOUND</p> <p><b>Meaning:</b> Program error. Resource specified on ALTER or RELEASE request is not owned or pending ownership. Note that this condition can occur when the request to obtain ownership failed or was denied by the contention exit of the connected user who has been selected to manage the resource contention.</p> <p><b>Action:</b> Retry request to gain (OBTAIN) ownership of the resource.</p>

Table 75. Return and reason codes for the IXLLOCK macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0812	<b>Equate Symbol:</b> IXLRSNCODEALREADYOWNED <b>Meaning:</b> Program error. Resource specified on OBTAIN request is already owned. <b>Action:</b> Check protocol to determine why duplicate request for resource has been issued.
8	xxxx0813	<b>Equate Symbol:</b> IXLRSNCODEALREADYPENDING <b>Meaning:</b> Program error. Resource specified on OBTAIN request is already pending ownership. <b>Action:</b> Check protocol to determine why duplicate request for resource has been issued.
8	xxxx0816	<b>Equate Symbol:</b> IXLRSNCODENORTEXISTS <b>Meaning:</b> Program error. A request to WRITE or REACQUIRE a record data entry was unable to be processed due to recording not being active in the lock structure indicated by the input contoken. For recording to be active, the first user to connect to the lock structure must have specified RECORD=YES on its IXLCONN invocation. <b>Action:</b> In order for recording to become active, that structure would have to be deleted and reallocated with RECORD=YES.
8	xxxx0817	<b>Equate Symbol:</b> IXLRSNCODEBADCONID <b>Meaning:</b> Program error. The request to conditionally reacquire an existing record data entry based on the connection with which it is associated has failed. The record data entry to be reacquired was not associated with the connection indicated by the input CONID keyword. <b>Action:</b> Verify that the CONID value was specified correctly.
8	xxxx0818	<b>Equate Symbol:</b> IXLRSNCODENOTLOCKSTR <b>Meaning:</b> Program error. The contoken specified does not represent a lock structure. <b>Action:</b> Verify that the CONTOKEN is specified correctly and is for the intended structure.
8	xxxx0844	<b>Equate Symbol:</b> IXLRSNCODEBADID <b>Meaning:</b> Program error. The attempt to REACQUIRE an existing record data entry has failed because the input ENTRYID does not designate an existing record data entry. <b>Action:</b> Verify that the ENTRYID is specified correctly.

Table 75. Return and reason codes for the IXLLOCK macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0876	<p><b>Equate Symbol:</b> IXLRNCODEBADREQNUM</p> <p><b>Meaning:</b> Program error. The REQNUM value specified on IXLLOCK REQUEST=PROCESSMULT is not valid. The value must be between 1 and 128 inclusive. Processing is halted with no entries in the REQBUFFER having been processed.</p> <p><b>Action:</b> Verify that the REQNUM value is specified correctly.</p>
8	xxxx0877	<p><b>Equate Symbol:</b> IXLRNCODEBADLRBTYPE</p> <p><b>Meaning:</b> Program error. A lock request block (LRB) that is input on an IXLLOCK REQUEST=PROCESSMULT request contains a value in the LRB_XTYPE field that is not valid.</p> <p><b>Action:</b> Verify that the LRB_XTYPE field contains the value of LRB_XTYPE_RELEASEVERSO. If specified, the REQPROC field contains the number of LRBs processed before this error was encountered when the PROCESSMULT request is completed synchronously.</p>
8	xxxx0878	<p><b>Equate Symbol:</b> IXLRNCODEBADREQBUFFER</p> <p><b>Meaning:</b> Program error. An error occurred while XES was attempting to access the storage area defined by REQBUFFER. The number of LRBs processed is returned in the REQPROC field if specified.</p> <p><b>Action:</b> An error occurred while XES was attempting to access the storage area defined by REQBUFFER. The number of LRBs processed is returned in the REQPROC field if specified and the PROCESSMULT request completed synchronously.</p>
8	xxxx0879	<p><b>Equate Symbol:</b> IXLRNCODEBADMODEVAL</p> <p><b>Meaning:</b> Program error. The value specified for MODEVAL is not valid.</p> <p><b>Action:</b> Verify that the value specified for MODEVAL is one of the possible mode value constants provided in the IXLYCON macro and that it is a valid value for the type of request being processed. If an LRB contains a mode value that is not supported, the system halts the request. The number of LRBs processed prior to the error is returned in the REQPROC field when the PROCESSMULT request completes synchronously.</p>
8	xxxx087A	<p><b>Equate Symbol:</b> IXLRNCODEBADRNAMELEN</p> <p><b>Meaning:</b> Program error. The value specified for RNAMELEN is not valid.</p> <p><b>Action:</b> Verify that the length specified for RNAMELEN is between 1 and 300.</p>

Table 75. Return and reason codes for the IXLLOCK macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx087B	<p><b>Equate Symbol:</b> IXLRSCODENOVARRNAME</p> <p><b>Meaning:</b> Program error. An IXLLOCK request that specified a variable length resource name is not valid because the variable length name attribute is not in effect for the lock structure represented by the input contoken. Specify the variable length name attribute at structure allocation time with the RNAMELEN keyword of the IXLCONN macro.</p> <p><b>Action:</b> If you want to use variable-length resource names, ensure that the initial IXLCONN invocation to connect to the lock structure specifies RNAMELEN=VAR300.</p>
8	xxxx087C	<p><b>Equate Symbol:</b> IXLRSCODEBADSYNCFAILDELAY</p> <p><b>Meaning:</b> Program error. The value specified for SYNCFAILDELAY is not valid.</p> <p><b>Action:</b> Verify that the value specified for SYNCFAILDELAY is one of the possible SYNCFAILDELAY value constants provided in the IXLCON macro.</p>
8	xxxx088F	<p><b>Equate Symbol:</b> IXLRSCODEBADREQVERSION</p> <p><b>Meaning:</b> Program error. The value specified for REQVERSION is not valid.</p> <p><b>Action:</b> Verify that the value specified for REQVERSION is in the range of 0 to 1.</p>
C	xxxx0C06	<p><b>Equate Symbol:</b> IXLRSCODENOCOONN</p> <p><b>Meaning:</b> Environmental error. There is no connectivity to the lock structure. This may occur due to operator commands such as VARY PATH OFFLINE or CONFIG CHP OFFLINE or hardware errors such as coupling facility or path failures. The contoken will be invalidated.</p> <p><b>Action:</b> Either disconnect from the structure using IXLDISC or rebuild the structure using IXLREBLD.</p>
C	xxxx0C0B	<p><b>Equate Symbol:</b> IXLRSCODERTFULL</p> <p><b>Meaning:</b> Environmental error. Record portion of the lock structure is full. This reason code may also be issued when the creation of a record data entry would result in the percentage of free entries being less than the percent entry reserved value in effect for the structure.</p> <p><b>Action:</b> If your protocol allows, attempt to rebuild the lock structure or alter its size so that additional record data might be available, or if the structure has a percent entry reserved threshold established, specify the ALLOWUSERSV keyword on the IXLLOCK request</p>

Table 75. Return and reason codes for the IXLLOCK macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C0D	<p><b>Equate Symbol:</b> IXLRNCSUPERSEDED</p> <p><b>Meaning:</b> Environmental error. OBTAIN or ALTER request has been superseded by a more current request for the resource.</p> <p><b>Action:</b> None expected.</p>
C	xxxx0C0F	<p><b>Equate Symbol:</b> IXLRNCODEDENIED</p> <p><b>Meaning:</b> Environmental error. OBTAIN or ALTER request is not granted. Resource request is denied (canceled) by a related connected user that is managing the contention environment for the resource.</p> <p><b>Action:</b> Retry request at a later time.</p>
C	xxxx0C13	<p><b>Equate Symbol:</b> IXLRNCODEREQPURGED</p> <p><b>Meaning:</b> Environmental error. The request was purged prior to completion of the request. Possible reasons include:</p> <ul style="list-style-type: none"> <li>• The connector failed.</li> <li>• The connector disconnected.</li> <li>• The requestor failed.</li> <li>• The request was purged by IXLPURGE.</li> <li>• Requests were purged when the connector provided an IXLEERSP response for the Rebuild Stop or Rebuild Cleanup event.</li> <li>• The secondary address space was no longer valid.</li> </ul> <p><b>Action:</b> None if this is expected. Otherwise, determine why the connector failed.</p>
C	xxxx0C20	<p><b>Equate Symbol:</b> IXLRNCODERESOURCESCONSTRAINED</p> <p><b>Meaning:</b> Environmental error. The amount of inuse storage is above a preestablished threshold. Incoming OBTAIN and ALTER requests will be delayed until sufficient storage is reclaimed to fall below the threshold.</p> <p><b>Action:</b> Wait until the amount of in use storage falls below the preestablished threshold. This occurs when sufficient resources have been released with subsequent UNLOCK or PROCESSMULT requests.</p> <p>OBTAIN or ALTER requests that cannot be delayed until sufficient storage is reclaimed should be specified with the CRITICALREQUEST option specified (IxlockCriticalRequestYes). Note that specifying this option for a request may cause an abend X'026' to be issued by the XES storage manager.</p>



Table 75. Return and reason codes for the IXLLOCK macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C25	<b>Equate Symbol:</b> IXLRSNCODESTRFAILURE <b>Meaning:</b> Environmental error. Prior to completion of the request, the lock structure failed. <b>Action:</b> Attempt to rebuild the structure using IXLREBLD or disconnect from the structure using IXLDISC.
C	xxxx0C4C	<b>Equate Symbol:</b> IXLRSNCODERESOURCENOLONGEROWNED <b>Meaning:</b> Environmental error. An IXLLOCK ALTER or IXLLOCK RELEASE request for a resource failed because the resource is no longer owned. <b>Action:</b> Check your protocol to determine why the resource is no longer owned.
C	xxxx0C68	<b>Equate Symbol:</b> IXLRSNCODEBADREQCFLEVEL <b>Meaning:</b> Environmental error. The request type is not permitted for the level of coupling facility in which the target structure is allocated. <b>Action:</b> Either disconnect from the structure (using IXLDISC) or initiate a rebuild of the structure, if allowed (using IXLREBLD) in a coupling facility of the correct CFLEVEL.
C	xxxx0C69	<b>Equate Symbol:</b> IXLRSNCODENODELAY <b>Meaning:</b> Environmental error. An IXLLOCK request in which the user specified MODE=SYNCFAIL encountered a delay. The request is canceled. <b>Action:</b> Retry request at a later time.
C	FFFFFFFF	<b>Equate Symbol:</b> IXLRSNCODENOTAVAILABLE <b>Meaning:</b> XES functions are not available. This can occur because the coupling facility hardware necessary to provide XES functions is not present. <b>Action:</b> Re-IPL the system, or follow your particular failure management protocol.
10	xxxx10xx	<b>Meaning:</b> Failure in XES processing. The state of the involved structure and the disposition of the request are unpredictable. <b>Action:</b> Save the reason code information and contact the IBM support center.



## Chapter 76. IXLLSTC – XES List Structure Control Services

### Description

The IXLLSTC macro provides operations to atomically perform locking functions, monitor list and sublist state changes, read and write list controls, read event queue controls and event monitor controls, and dequeue event monitor controls.

The IXLLSTC macro provides list services equivalent to the following IXLLIST request types:

- IXLLIST REQUEST=DEQ\_EVENTQ
- IXLLIST REQUEST=LOCK
- IXLLIST REQUEST=MONITOR\_EVENTQ
- IXLLIST REQUEST=MONITOR\_LIST
- IXLLIST REQUEST=MONITOR\_SUBLIST
- IXLLIST REQUEST=MONITOR\_SUBLISTS
- IXLLIST REQUEST=READ\_EMCONTROLS
- IXLLIST REQUEST=READ\_EQCONTROLS
- IXLLIST REQUEST=READ\_LCONTROLS
- IXLLIST REQUEST=WRITE\_LCONTROLS

The functions available with the IXLLSTC macro that do not have an IXLLIST equivalent are the following:

- Keyrange monitoring (with REQUEST=MONITOR\_KEYRANGE) allows you to monitor the empty or not-empty state of a set of keyrange values, based on user-defined threshold counts. Both the threshold counts and the starting and ending keyrange values are defined with IXLLSTC REQUEST=WRITE\_LCONTROLS.
- Support for secondary keys is provided on the REQUEST=READ\_EQCONTROLS, REQUEST=READ\_EMCONTROLS, REQUEST=DEQ\_EVENTQ, REQUEST=MONITOR\_EVENTQ, MONITOR\_SUBLIST, MONITOR\_SUBLISTS request types.
- Support for a request that whenever a list entry is queued to a monitored sublist, an EMC is queued to the registered user's event queue is provided on the REQUEST=MONITOR\_SUBLIST and REQUEST=MONITOR\_SUBLISTS request types.

The functions that are available with the IXLLSTC macro (on a z/OS system that supports coupling facility CFLEVEL=22 or higher function) for list structures that are allocated in a coupling facility of CFLEVEL=22 or higher are:

- List monitoring for list full to not-full state transitions and from not-full to full, can be established by using REQUEST=MONITOR\_LIST, MONITORTYPE=NOTFULL. The IXLVECTR service can be used to determine whether the list is considered full or not-full based on the list limits established with an IXLLSTC WRITE\_LCONTROLS request.

The macro IXCYQUAA defines the QuReqRfNotFullMonitoring bit in the QuReqFeatures string, which can be used to determine if list not-full monitoring support and IXLLSTC MONITORTYPE keyword is supported on the system. Use IXCQUERY REQINFO=FEATURES to retrieve the QuReqFeatures string.

- More frequent updates to the list notification vector can be requested by specifying NOTIFICATION=EVERY on the IXLLSTC REQUEST=MONITOR\_LIST or REQUEST=MONITOR\_KEYRANGE request. NOTIFICATION=EVERY causes the list notification vector to be updated when the list or

keyrange transitions from the not-empty state to the empty state and every time a list entry is added to the list or keyrange and the established not-empty threshold count is exceeded.

The combination of NOTIFICATION=EVERY and DRIVEEXIT=YES (specified on the REQUEST=MONITOR\_LIST or REQUEST=MONITOR\_KEYRANGE request) gives the connection's list transition exit the opportunity to receive control more frequently.

The macro IXCYQUAA defines the QuReqRfAggressiveNotify bit in the QuReqFeatures string, which can be used to determine if aggressive notifications for list and keyranges are supported on the system. Use IXCQUERY REQINFO=FEATURES to retrieve the QuReqFeatures string.

- List and keyrange notification delays that are defined for a structure in the CFRM active policy can be turned on or off on a list and keyrange basis. By default, when specified for a structure, notification delays are applied when notifying monitoring instances for list and keyrange transitions from an empty to not-empty state. IXLLSTC REQUEST=WRITE\_LCONTROLS can be used to deactivate and later reactivate notification delays for a specific list or keyrange by using the LISTNOTIFYDELAY and KEYRNOTIFYDELAY keywords.

See *z/OS MVS Programming: Sysplex Services Guide* for information about using the IXLLSTC macro.

## Environment

The following are the environment requirements for the caller.

Environment	Environment requirement
Minimum authorization:	Supervisor state and PSW key 0-7
Dispatchable unit mode:	Task or SRB
Cross memory mode:	Any PASN, any HASN, any SASN. The current primary address space must be the same as the primary address space at the time the connection service (IXLCONN) was issued for the structure.
AMODE:	31-bit
ASC mode:	Primary or access register (AR)
Interrupt status:	Enabled or disabled for I/O and external interrupts
Locks:	Disabled callers must be legally disabled by holding the CPU lock and cannot hold other disabled locks. Enabled callers must not hold any locks. When MODE=SYNCSUSPEND is specified, the caller must be enabled.
Control parameters:	See “Restrictions” on page 1373

## Programming Requirements

- If your program is in AR mode, issue the SYSSTATE ASCENV=AR macro before IXLLSTC. SYSSTATE ASCENV=AR tells the system to generate code appropriate for AR mode.
- Include the IXLYCON mapping macro in your program. This macro provides a list of equate symbols for users of XES services and exits.
- Include mapping macros IXLYEMC, IXLYLAA, IXLYLCTL, IXLYLMI, and IXLYMSRI in your program as necessary. Table 76 on page 1372 lists these macros, the information and areas they map, and the particular IXLLSTC requests and parameters they apply to.

Table 76. Mapping Macros for IXLLSTC

Mapping Macro	Information	Area	Request/Parameter
IXLYEMC	Event monitor controls	BUFLIST buffers, BUFFER area	DEQ_EVENTQ

Table 76. Mapping Macros for IXLLSTC (continued)

Mapping Macro	Information	Area	Request/Parameter
IXLYLAA	Answer area output	ANSAREA area	All requests
IXLYLMI	List monitoring information	BUFLIST buffers, BUFFER area	READ_LCONTROLS
IXLYMSRI	Monitor sublists registration information	BUFLIST buffers, BUFFER area	MONITOR_SUBLISTS

## Restrictions

- If you specify BUFLIST and PAGEABLE=YES, all of the buffers in the list must be in the same area of storage; you cannot mix common and private storage addresses for the buffers in the list.
- This service cannot be invoked by callers running as a disabled interrupt exit (DIE).
- The caller's parameter list must be addressable in the caller's primary address space.
- If the caller is running in AR ASC mode and specifies a parameter using explicit register notation, the access register corresponding to the general register must appropriately qualify the general register.
- The virtual storage areas specified by the ADJAREA and ANSAREA parameters must be addressable in the caller's primary address space or from the caller's PASN access list.
- The virtual storage areas specified by parameters other than ADJAREA and ANSAREA can be addressable in the caller's primary, secondary, or home address spaces, from the PASN access list, or from the dispatchable unit access list (DU-AL).
- If the caller is disabled then the parameter list and all storage areas addressed by the macro parameters must reside in either nonpageable or disabled reference storage.

## Input Register Information

Before issuing the IXLLSTC macro, the caller does not have to place any information into any register unless using it in register notation for a particular parameter, or using it as a base register.

## Output Register Information

When control returns to the caller, the GPRs contain:

### Register

#### Contents

**0**

Reason code

**1**

Used as work register by the system

**2-13**

Unchanged

**14**

Used as work register by the system

**15**

Return code

When control returns to the caller, the ARs contain:

### Register

#### Contents

**0-1**

Used as work registers by the system

**2-13**

Unchanged

**14-15**

Used as work registers by the system

## Performance Implications

---

See *z/OS MVS Programming: Sysplex Services Guide* for performance information.

## Understanding IXLLSTC Version Support

---

The IXLLSTC macro supports versions 0, 1, 2, 4 and 5 — the version number corresponds to the level of the IXLLIST or IXLLSTC macro in which a keyword or function was introduced. The higher the version number, the more recent the version of the macro. Some IXLLSTC keywords are supported for multiple versions, but have different functions for each version.

- Keywords not specifically noted here are supported by all versions starting with version 0 and higher of the IXLLSTC macro.
- The following keywords are supported by all versions starting with version 1 and higher of the IXLLSTC macro.
  - LISTKEY
  - MAXLISTKEY
  - SETCURSOR
- The following keywords and functions are supported by all versions starting with version 2 and higher of the IXLLSTC macro.
  - ENDINDEX
  - ENTRYKEY
  - KEYTYPE
  - MOSVECTOR
  - NOTIFICATION
  - STARTINDEX
  - UNC
- The following keywords and functions are supported by all versions starting with version 4 and higher of the IXLLSTC macro.
  - KEYRANGEEND
  - KEYRANGESTART
  - KREMPY
  - KRNOTEMPTY
  - LISTEMPTY
  - LISTNOTEMPTY
  - SECONDARYKEY
- The following keywords and functions are supported by all versions starting with version 5 and higher of the IXLLSTC macro.
  - MONITORTYPE
  - KEYRNOTIFYDELAY
  - LISTNOTIFYDELAY

Specify the version of the parameter list that you want generated with the PLISTVER keyword. See Chapter 2, “Specifying a Macro Version Number,” on page 5 for considerations when specifying the version of the parameter list with PLISTVER.

## Summary of Version-Dependent Parameter Functions

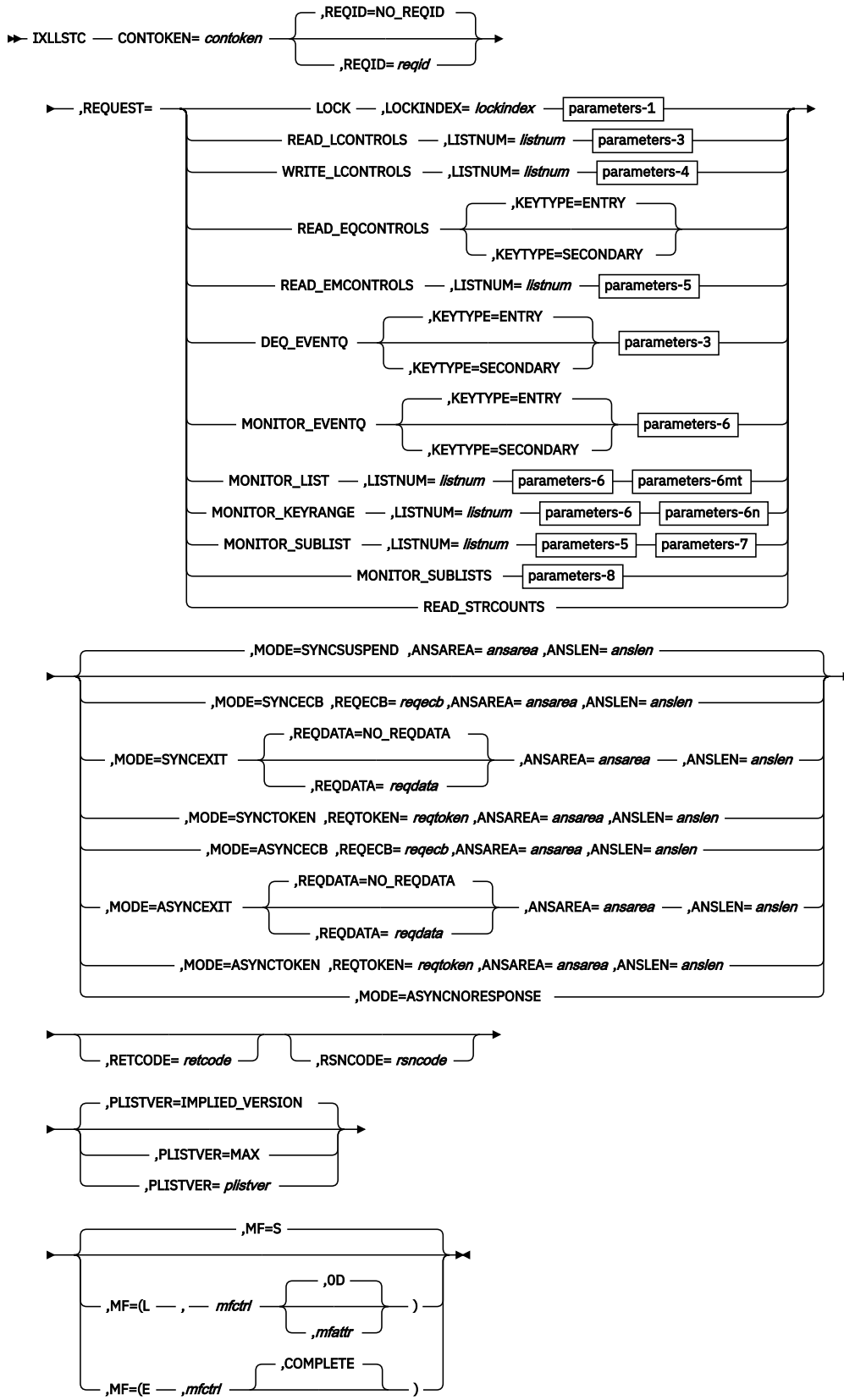
Different parameter list versions may be generated depending on the value specified for certain keywords. The following table summarizes the minimum version required to support such specifications. When specifying PLISTVER, be sure that it is at least as high as the highest version number of all the key values being used.

<i>Table 77. IXLLSTC Version Support</i>		
Keyword	Version	Key Value
REQUEST	0	LOCK, READ_LCONTROLS,
		WRITE_LCONTROLS, MONITOR_LIST
	2	READ_EQCONTROLS, READ_EMCONTROLS,
		DEQ_EVENTQ, MONITOR_EVENTQ,
		MONITOR_SUBLIST, MONITOR_SUBLISTS
	4	MONITOR_KEYRANGE
KEYTYPE	2	ENTRY
	4	SECONDARY
NOTIFICATION	2	FIRST
	4	EVERY for MONITOR_SUBLIST
	5	EVERY for MONITOR_LIST and MONITOR_KEYRANGE

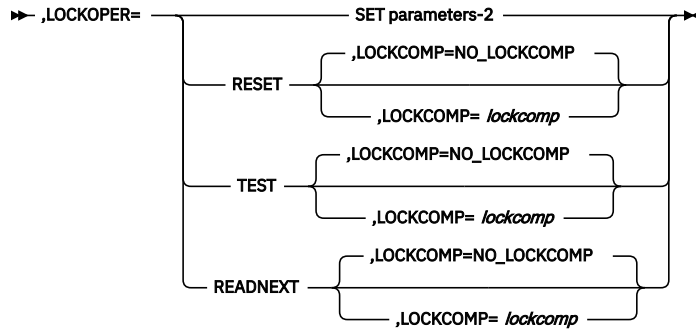
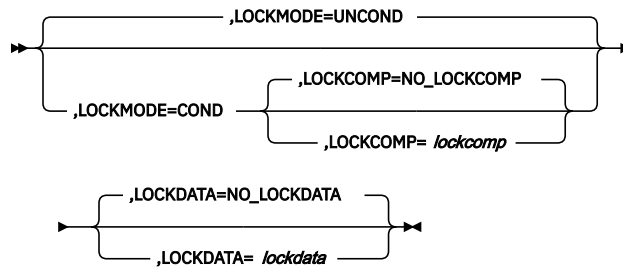
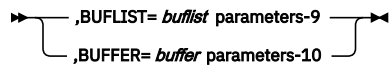
## Syntax Diagram

The syntax diagram for IXLLSTC is as follows:

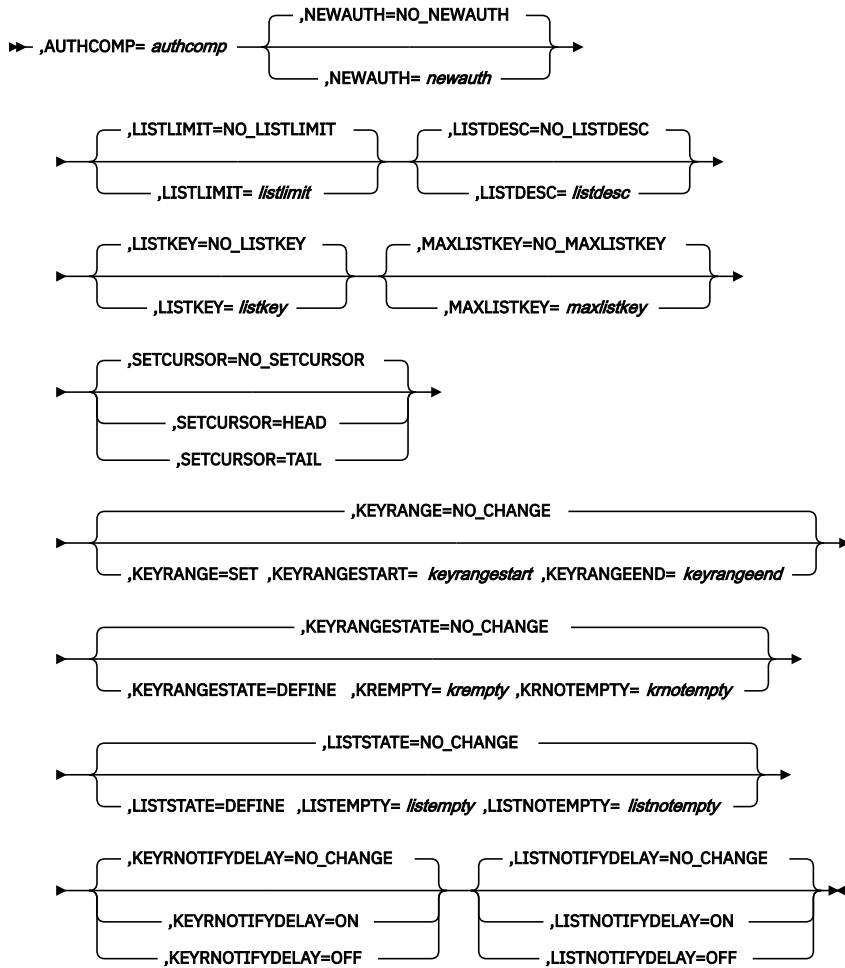
## main diagram



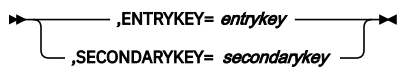


**parameters-1****parameters-2****parameters-3**

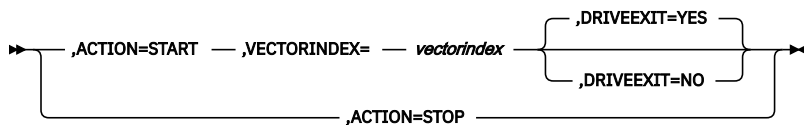
## parameters-4



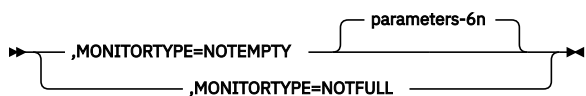
## parameters-5



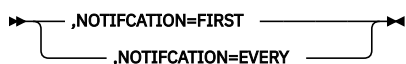
## parameters-6

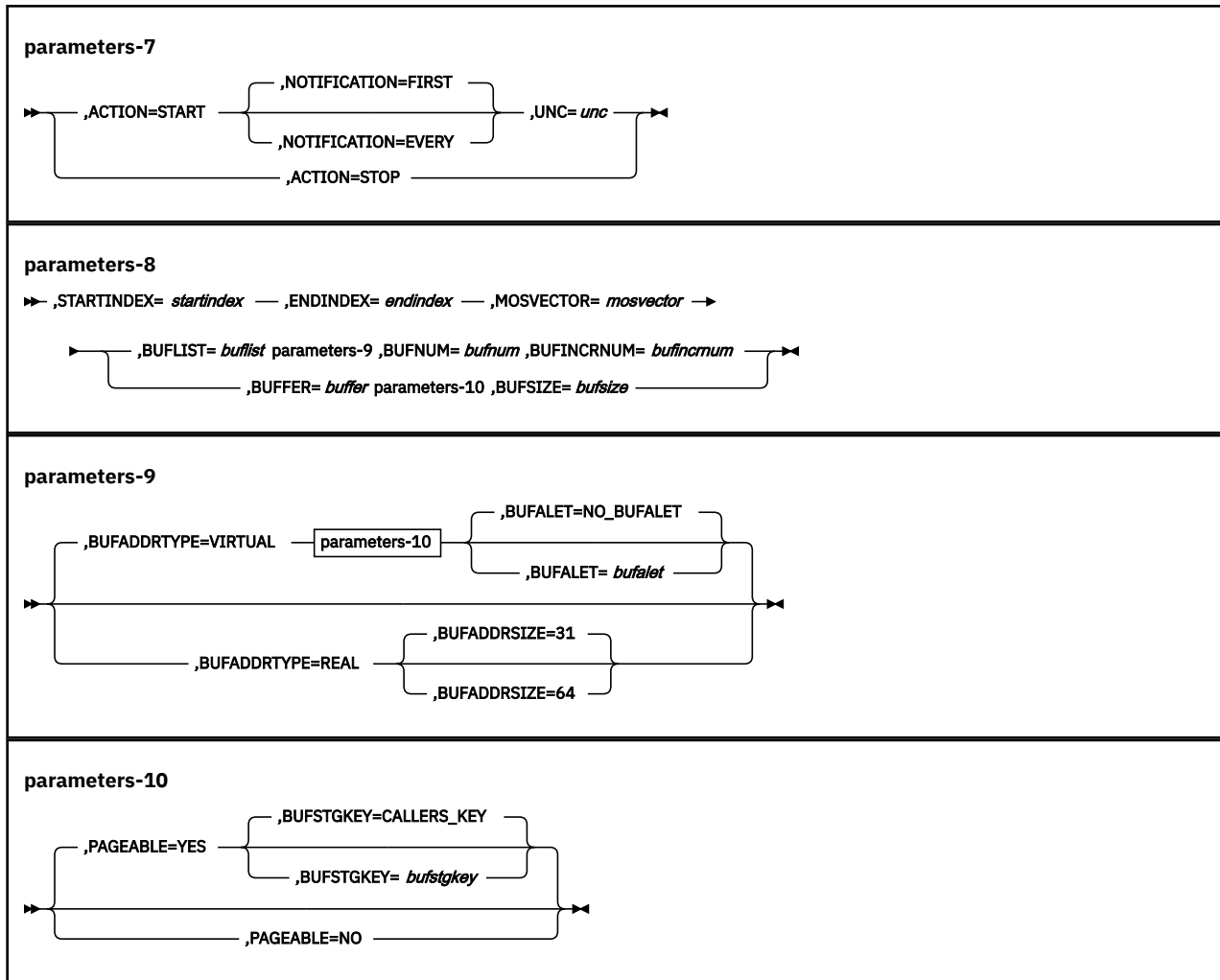


## parameters-6mt



## parameters-6n





## Parameter Descriptions

The parameter descriptions for IXLLSTC are listed in alphabetical order. Default values are underlined:

### **,ACTION=START**

### **,ACTION=STOP**

Depending on the IXLLSTC request type, use this input parameter to specify:

- For REQUEST=MONITOR\_EVENTQ, whether event queue monitoring is to be started or stopped. Requires that the list structure be allocated in a coupling facility of CFLEVEL=3 or higher.
- For REQUEST=MONITOR\_LIST, whether list monitoring is to be started or stopped.
- For REQUEST=MONITOR\_KEYRANGE, whether key-range monitoring is to be started or stopped. Requires that the list structure be allocated in a coupling facility of CFLEVEL=9 or higher.
- For REQUEST=MONITOR\_SUBLIST, whether sublist monitoring is to be started or stopped. Requires that the list structure be allocated in a coupling facility of CFLEVEL=3 or higher.

### **START**

- For REQUEST=MONITOR\_EVENTQ, start event queue monitoring for the connection specified by CONTOKEN for the specified event queue.
- For REQUEST=MONITOR\_LIST, start monitoring for the connection specified by CONTOKEN for the list specified by LISTNUM.
- For REQUEST=MONITOR\_KEYRANGE, start key-range monitoring for the connection specified by CONTOKEN for the list specified by LISTNUM.

- For REQUEST=MONITOR\_SUBLIST, start sublist monitoring for the connection specified by CONTOKEN for the designated sublist.

**STOP**

- For REQUEST=MONITOR\_EVENTQ, stop monitoring the event queue for the connection specified by CONTOKEN.
- For REQUEST=MONITOR\_LIST, stop list monitoring for the connection specified by CONTOKEN for the list specified by LISTNUM.
- For REQUEST=MONITOR\_KEYRANGE, stop key-range monitoring for the connection specified by CONTOKEN for the list specified by LISTNUM.
- For REQUEST=MONITOR\_SUBLIST, stop monitoring of the designated sublist for the connection specified by CONTOKEN.

**,ANSAREA=ansarea**

Use this input parameter to specify an answer area to contain information returned from the request. The format of the answer area is described by mapping macro IXLYLAA.

Not all fields in the answer area are applicable to all request types. Request type descriptions indicate which answer area fields are applicable for successful request completion cases. Return and reason code descriptions indicate which answer area fields are applicable for non-successful completing requests.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an area (with a length of ANSLEN) that will contain the information returned by the request.

**,ANSLEN=anslen**

Use this input parameter to specify the size of the storage area specified by ANSAREA.

Check the prologue of the IXLYLAA mapping macro for the minimum required size of the answer area, or use the length field (LAA\_LEN) in the LAA.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 2-byte field that contains the size, in bytes, of the answer area.

**,AUTHCOMP=authcomp**

Use this input parameter to specify a value to be compared to the list authority value of the list designated by LISTNUM. If the specified list authority fails to equal the list authority for the designated list, the IXLLSTC operation is terminated with no resultant change to the structure. Indicative return and reason codes are provided to the invoker.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16 byte field that contains the list authority value.

**,BUFADDRSIZE=31****,BUFADDRSIZE=64**

Use this input parameter to specify whether a 31-bit or a 64-bit address is specified by a BUFLIST entry.

**31**

The entry in BUFLIST is 31 bits in size.

**64**

The entry in BUFLIST is 64 bits in size.

**,BUFADDRTYPE=VIRTUAL****,BUFADDRTYPE=REAL**

Use this input parameter to specify whether the buffer addresses specified in the BUFLIST list are virtual storage or real storage addresses.

**VIRTUAL**

The buffer addresses are virtual storage addresses. The virtual storage can be pageable or nonpageable. See the PAGEABLE parameter for information about managing storage binds when specifying virtual storage addresses.

**REAL**

The buffer addresses are real storage addresses.

It is the caller's responsibility to manage the binds between the data buffer virtual storage and the real storage addresses provided. The caller must ensure that the data buffer virtual storage remains bound to the real storage addresses provided until the request completes.

**,BUFALET=NO\_BUFALET****,BUFALET=*bufalet***

Use this input parameter to specify an access list entry token (ALET) to be used in referencing all of the buffers specified by BUFLIST.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 4-byte field that contains the ALET.

**,BUFFER=*buffer***

Use this input or output parameter to hold either input data for the request or output data from the request. Only 31-bit addressable virtual storage areas (below 2GB) are supported by the BUFFER specification. High virtual storage areas (above 2GB) can only be specified via the BUFLIST specification.

- For READ\_LCONTROLS requests, the buffer must be 4096 bytes on a 4096-byte boundary.

Upon successful completion of a READ\_LCONTROLS request, the BUFFER area contains, starting at offset zero, an array of list monitoring information for the specified list and an array of key-range monitoring information for the specified list. The relative position of an array element associates it with a connection identifier. The first array element is associated with a connection identifier of zero, and is reserved. The length and contents of each array element is defined by mapping macro IXLYLMI.

- For DEQ\_EVENTQ requests, the buffer must be 4096 bytes on a 4096-byte boundary.

Upon successful completion of a DEQ\_EVENTQ request, the BUFFER area contains, starting at offset zero, an array of event monitor controls that were dequeued from the event queue. The length and contents of each array element is defined by mapping macro IXLYEMC.

- For MONITOR\_SUBLISTS requests, the BUFSIZE keyword specifies the size of the buffer. You can define the buffer size to be a total size of up to 65536 bytes. Depending on the size you select, the following restrictions apply:
  - If you specify a buffer size of less than or equal to 4096 bytes, you must ensure that the buffer:
    - Is 256, 512, 1024, 2048, or 4096 bytes.
    - Starts on a 256-byte boundary.
    - Does not cross a 4096-byte boundary.
  - If you specify a buffer size of greater than 4096 bytes, you must ensure that the buffer:
    - Is a multiple of 4096 bytes.
    - Is less than or equal to 65536 bytes.
    - Starts on a 4096-byte boundary.

For a MONITOR\_SUBLISTS request, the BUFFER is used to input an array of entries, each mapped by IXLYMSRI, and each of which contains the information necessary to register as a sublist monitor for one sublist.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an area (with a length of BUFSIZE) that contains the entry data.

**,BUFINCRNUM=*bufincrnum***

Use this input parameter to specify the number of 256-byte segments comprising each buffer in the BUFLIST list.

Valid BUFINCRNUM values are 1, 2, 4, 8, or 16, which correspond to BUFLIST buffer sizes of 256, 512, 1024, 2048, and 4096 bytes respectively.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 1-byte field that contains 1, 2, 4, 8, or 16.

**,BUFLIST=buflist**

Use this input or output parameter to specify a list of buffers to contain input data for the request or to receive output data from the request.

Either 31-bit addressable (below 2GB) or 64-bit addressable (above 2GB) real or virtual storage areas are supported for the BUFLIST specification, depending on the specifications for the BUFADDRTYPE and BUFADDRSIZE keywords. However, pageable high shared virtual storage areas (above 2GB) may not be used.

The buffer address description is an 8-byte element. The first four bytes of the element is reserved space. The second four bytes of the element contains the address of the buffer.

- For READ\_LCONTROLS requests, only one buffer may be passed. The buffer must be 4096 bytes in length and must start on a 4096-byte boundary.

Upon successful completion of a READ\_LCONTROLS request, the BUFLIST buffer contains, starting at offset zero, an array of list monitoring information for the specified list and an array of key-range monitoring information for the specified list. The relative position of an array element associates it with a connection identifier. The first array element is associated with a connection identifier of zero, and is reserved. The length and contents of each array element is defined by mapping macro IXLYLMI.

- For DEQ\_EVENTQ requests, only one buffer may be passed. The buffer must be 4096 bytes in length on a 4096-byte boundary.

Upon successful completion of a DEQ\_EVENTQ request, the BUFLIST buffer contains, starting at offset zero, an array of event monitor controls that were dequeued from the event queue. The length and contents of each array element is defined by mapping macro IXLYEMC.

- For MONITOR\_SUBLISTS requests, there may be 1 to 16 buffers in the list. Each buffer in the list must be the same size and must reside in the same address space or data space. Data is fetched from the buffers in the order specified.

For MONITOR\_SUBLISTS requests, the length of a buffer must be a multiple of 256 bytes between 256 and 4096. Each buffer must start on a 256-byte boundary and must not cross a 4096-byte boundary.

For a MONITOR\_SUBLISTS request, the BUFLIST is used to input an array of entries, each mapped by IXLYMSRI, and each of which contains the information necessary to register as a sublist monitor for one sublist.

**Note:** The buffers do not have to be contiguous in storage. (List services treats BUFLIST buffers as a single, contiguous buffer, even if the buffers are not contiguous.)

See the BUFNUM and BUFINCRNUM parameter descriptions for defining the number and size of buffers.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 128-byte area that contains the list of buffer addresses.

**,BUFNUM=bufnum**

Use this input parameter to specify the number of buffers in the BUFLIST list. Valid BUFNUM values are from 1 to 16.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 1-byte field that contains the number of buffers in the buffer list.

**,BUFSIZE=bufsize**

Use this input parameter to specify the size of the BUFFER area. See the BUFFER parameter description for valid buffer sizes.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a fullword field that contains the size of the buffer (BUFFER) in bytes.

**,BUFSTGKEY=CALLERS\_KEY****,BUFSTGKEY=bufstgkey**

Use this input parameter to specify a storage key that you define and use when referencing the buffers specified by BUFLIST or the buffer that is specified by BUFFER.

If you do not specify BUFSTGKEY, or if you specify BUFSTGKEY=CALLERS\_KEY, all references to one or more buffers are performed by using the caller's PSW key.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of an 8-bit field that contains the storage key in the format B'kkkkxxxx', where kkkk is the key and xxxx is ignored.

**CONTOKEN=contoken**

Use this input parameter to specify the connect token that was returned by the IXLCONN service. The connect token uniquely identifies your connection to the list structure.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-byte field that contains the connect token.

**,DRIVEEXIT=YES****,DRIVEEXIT=NO**

Depending on the IXLLSTC request type, use this input parameter to specify:

- For REQUEST=MONITOR\_EVENTQ, whether XES should drive the connection's list transition exit when the state of the specified user's event queue changes from empty to not-empty.
- For REQUEST=MONITOR\_LIST, whether XES should drive the connection's list transition exit when the state of a list changes from empty to not-empty or from full to not-full.

MONITORTYPE determines the type of list transition that causes XES to drive the connection's list transition exit.

- For REQUEST=MONITOR\_KEYRANGE, whether XES should drive the connection's list transition exit when the state of the key-range changes from empty to not-empty.

**YES**

- For REQUEST=MONITOR\_EVENTQ, when the state of the user's event queue changes from empty to not-empty, XES drives the connection's list transition exit.
- For REQUEST=MONITOR\_LIST, when the state of a list changes from empty to not-empty or from full to not-full, XES drives the connection's list transition exit. When NOTIFICATION=EVERY is specified, XES processing is initiated to drive the connection's list transition exit when a list entry is added to the monitored list and the list not-empty threshold count for the list is exceeded instead of only when the list state transitions from empty to not-empty.
- For REQUEST=MONITOR\_KEYRANGE, when the state of a key-range changes from empty to not-empty, XES drives the connection's list transition exit. When NOTIFICATION=EVERY is specified, XES processing is initiated to drive the connection's list transition exit whenever a list entry is added to the monitored keyrange and the keyrange not-empty threshold count for the keyrange is exceeded instead of only when the keyrange state transitions from empty to not-empty.

**NO**

- For REQUEST=MONITOR\_EVENTQ, when the state of the user's event queue changes from empty to not-empty, XES does not drive the connection's list transition exit.
- For REQUEST=MONITOR\_LIST, when the state of a list changes, XES does not drive the connection's list transition exit.
- For REQUEST=MONITOR\_KEYRANGE, when the state of the key-range changes from empty to not-empty, XES does not drive the connection's list transition exit.

**,ENDINDEX=endindex**

For REQUEST=MONITOR\_SUBLIST, use this input parameter to specify the 1-origin index number of the last IXLYMSRI entry, which is requested to be processed. The valid range is from the STARTINDEX value to 1024, inclusive. The specified index must not imply a larger buffer size than the user provided for the BUFFER or BUFLIST.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a halfword field that contains the 1-origin index number of the last entry to be processed that contains the entry identifier.

**,ENTRYKEY=entrykey**

Depending on the IXLLSTC request type, use this input parameter to specify an unsigned 128-bit entry key to be used in conjunction with LISTNUM to designate:

- For REQUEST=READ\_EMCONTROLS, a sublist whose EMCs are to be read.
- For REQUEST=MONITOR\_SUBLIST, a sublist whose monitoring is to be started or stopped.

Specify ENTRYKEY only for structures that support keyed entries. The use of ENTRYKEY requires that the list structure be allocated in a coupling facility of CFLEVEL=3 or higher.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-byte field that contains the unsigned 128-bit entry key.

**,KEYRANGE=NO\_CHANGE**

**,KEYRANGE=SET**

For REQUEST=WRITE\_LCONTROLS, use this input parameter to specify whether the key-range is to be updated.

**NO\_CHANGE**

The key-range currently defined for the list will remain unchanged.

**SET**

The key-range defined for the list is to be set to the range specified by KEYRANGESTART and KEYRANGEEND.

SET is valid for keyed list structures allocated in a coupling facility of CFLEVEL=9 or higher. SET is ignored and the request processed as if NO\_CHANGE were specified when the structure is allocated in a coupling facility of CFLEVEL=8 or lower, or if the structure does not support keyed list entries.

The default key-range start and end key values are initialized to binary zeros for all lists when the structure is allocated.

**,KEYRANGEEND=keyrangeend**

Use this input parameter to specify an unsigned 128-bit key-range end key value to be associated with the list. The end key value must be greater than or equal to the start key value.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-byte field that contains the key-range end key value.

**,KEYRANGESTART=keyrangestart**

Use this input parameter to specify an unsigned 128-bit key-range start key value to be associated with the list.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-byte field that contains the key-range start key value.

**,KEYRANGESTATE=NO\_CHANGE**

**,KEYRANGESTATE=DEFINE**

For REQUEST=WRITE\_LCONTROLS, use this input parameter to specify whether the threshold counts that define the empty or not-empty state of a key-range are to be updated.

**NO\_CHANGE**

The threshold counts will remain unchanged.

**DEFINE**

The threshold counts that define the empty or not-empty state of a key-range are to be set to the values indicated by KREMPY and KRNOTEMPTY.

DEFINE is valid for keyed list structures allocated in a coupling facility of CFLEVEL=9 or higher. DEFINE is ignored and the request processed as if NO\_CHANGE were specified when the structure is allocated in a coupling facility of CFLEVEL=8 or lower, or if the structure does not support keyed list entries.



A key-range is either in the empty state or the not-empty state. Use a REQUEST=MONITOR\_KEYRANGE request to register interest in monitoring transitions between key-range states.

The default key-range empty or not-empty threshold counts are initialized to zero for all lists when the structure is allocated.

A request to define the key-range empty or not-empty threshold counts can timeout. The caller is expected to reissue the request until it completes without timing out.

#### **,KEYRNOTIFYDELAY=NO\_CHANGE**

#### **,KEYRNOTIFYDELAY=ON**

#### **,KEYRNOTIFYDELAY=OFF**

For REQUEST=WRITE\_LCONTROLS, use this input parameter to specify whether the key-range notification delay value for the structure is applied when notifying monitoring instances of empty to not-empty state transitions for the list's (specified by LISTNUM) keyrange.

**Note:** The default setting for KEYRNOTIFYDELAY for all lists when the structure is allocated, is ON.

KEYRNOTIFYDELAY is applicable when the structure is allocated with keys in a CFLEVEL=22 or higher coupling facility and the structure definition statement for the structure in the CFRM active policy specifies a non-zero KEYRNOTIFYDELAY value.

If KEYRNOTIFYDELAY is specified and the operational level of the coupling facility in which the structure is allocated (see ConaCFacilityCFLevel in the IXLCONN Connect Answer Area) is less than CFLEVEL=22, the WRITE\_LCONTROLS request fails with return code, Ix1RetCodeEnvError, and reason code, Ix1RsnCodeBadRegCflevel.

#### **NO\_CHANGE**

The key-range notification delay value for the list remains unchanged.

#### **ON**

The key-range notification delay value for the structure is applied when notifying monitoring instances of empty to not-empty state transitions for this list's keyrange.

#### **OFF**

The key-range notification delay value for the structure is not applied when notifying monitoring instances of empty to not-empty state transitions for this list's keyrange.

#### **,KEYTYPE=ENTRY**

#### **,KEYTYPE=SECONDARY**

Depending on the IXLLSTC request type, use this input parameter to specify:

- For REQUEST=READ\_EQCONTROLS, whether to return information about the event queue for state transitions of sublists identified by list entry key or to return information about the event queue for state transitions of sublists identified by secondary key.
- For REQUEST=DEQ\_EVENTQ, whether to dequeue EMCs for state transitions of sublists identified by list entry key or to dequeue EMCs for state transitions of sublists identified by secondary key.
- For REQUEST=MONITOR\_EVENTQ, whether to monitor the event queue for state transitions of sublists identified by list entry key or to monitor the event queue for state transitions of sublists identified by secondary key.

#### **ENTRY**

- For REQUEST=READ\_EQCONTROLS, return information about the event queue for state transitions of sublists identified by list entry key.
- For REQUEST=DEQ\_EVENTQ, dequeue the EMCs for state transitions of sublists identified by list entry.
- For REQUEST=MONITOR\_EVENTQ, monitor the event queue for state transitions of sublists identified by list entry key.

KEYTYPE=ENTRY is valid only when the structure is allocated in a coupling facility of CFLEVEL=3 or higher.

**SECONDARY**

- For REQUEST=READ\_EQCONTROLS, return information about the event queue for state transitions of sublists identified by secondary key.
- For REQUEST=DEQ\_EVENTQ, dequeue the EMCs for state transitions of sublists identified by secondary key.
- For REQUEST=MONITOR\_EVENTQ, monitor the event queue for state transitions of sublists identified by secondary key.

KEYTYPE=SECONDARY is valid only when the structure is allocated in a coupling facility of CFLEVEL=9 or higher.

**,KREMPY=*krempty***

Use this input parameter to specify the key-range empty threshold count to be associated with the list.

A key-range is in the empty state if the number of list entries in the key-range is either less than or equal to the KREMPY threshold or zero. Upon completion of the key-range state definition, the key-range is also considered to be in the empty state if the number of list entries in the key-range is less than or equal to the KRNOTEMPTY threshold. Once the key-range state becomes empty, it remains empty until the number of list entries in the key-range becomes greater than the KRNOTEMPTY threshold.

The KREMPY keyword is valid only for a keyed list structure allocated in a coupling facility of CFLEVEL=9 or higher.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword field that contains the key-range empty threshold count to be associated with the list.

**,KRNOTEMPTY=*krnotempty***

Use this input parameter to specify the key-range not-empty threshold count to be associated with the list. The key-range not-empty count must be greater than or equal to the key-range empty count.

A key-range is in the not-empty state if the number of list entries in the key-range is greater than the KRNOTEMPTY threshold. Once the key-range state is not-empty, it remains not-empty until the number of list entries in the key-range either becomes less than or equal to the KREMPY threshold or becomes zero.

The KRNOTEMPTY keyword is valid only for a keyed list structure allocated in a coupling facility of CFLEVEL=9 or higher.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword field that contains the key-range not-empty threshold to be associated with the list.

**,LISTDESC=NO LISTDESC****,LISTDESC=*listdesc***

For REQUEST=WRITE\_LCONTROLS, use this input parameter to specify the user-defined description to be associated with the list. If LISTDESC is not specified, the user description for the designated list will remain unchanged.

**Note:** The default list description for all lists when the structure is allocated is binary zeros.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 32-byte field that contains the user-defined description.

**,LISTEMPTY=*listempty***

Use this input parameter to specify the list empty threshold count to be associated with the list.

A list is in the empty state if the number of list entries on the list is either zero or less than or equal to the LISTEMPTY threshold.

The LISTEMPTY keyword is valid only for a list structure allocated in a coupling facility of CFLEVEL=9 or higher.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword field that contains the list empty threshold count.

**,LISTKEY=NO LISTKEY**

**,LISTKEY=listkey**

For REQUEST=WRITE\_LCONTROLS, use this input parameter to specify the list key to be associated with the list. The list key value may be assigned to list entries when they are created or moved.

If LISTKEY is not specified, the list key for the designated list will remain unchanged. LISTKEY is ignored when the structure does not support keyed entries.

**Note:** The default list key for all lists when the structure is allocated is binary zeros.

The LISTKEY keyword is only meaningful for list structures allocated in a coupling facility of CFLEVEL=1 or higher.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-byte field that contains the list key to be associated with the list.

**,LISTLIMIT=NO LISTLIMIT**

**,LISTLIMIT=listlimit**

For REQUEST=WRITE\_LCONTROLS, use this input parameter to specify the list limit bounding the number of entries or elements that can reside on the list. If LISTLIMIT is not specified, the list limit for the designated list will remain unchanged.

**Note:** The default list limit for all lists when the structure is allocated is the total number of list entries or elements in the structure.

When an IXLALTER is processed for a list structure, if the list limit for a list is equal to the default value then it will be automatically adjusted to the new number of entries or elements. If a list limit is not equal to the default value then the alter process will not adjust it and it is the responsibility of the user, if desired, to set a new limit taking into account the current entry and element counts for the altered structure.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword field that contains the list limit for the list.

**,LISTNOTEMPTY=listnotempty**

Use this input parameter to specify the list not-empty threshold count to be associated with the list. The list not-empty threshold count must be greater than or equal to the list empty threshold count.

A list is in the not-empty state if the number of list entries on the list is greater than the LISTNOTEMPTY threshold.

The LISTNOTEMPTY keyword is valid only for a list structure allocated in a coupling facility of CFLEVEL=9 or higher.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword field that contains the list not-empty threshold count.

**,LISTNOTIFYDELAY=NO CHANGE**

**,LISTNOTIFYDELAY=ON**

**,LISTNOTIFYDELAY=OFF**

For REQUEST=WRITE\_LCONTROLS, use this input parameter to specify whether the list notification delay value for the structure is applied when notifying monitoring instances of empty to not-empty state transitions for the list that is specified by LISTNUM.

**Note:** The default setting for LISTNOTIFYDELAY for all lists when the structure is allocated, is ON.

LISTNOTIFYDELAY is applicable when the structure is allocated with keys in a CFLEVEL=22 or higher coupling facility and the structure definition statement for the structure in the CFRM active policy specifies a non-zero LISTNOTIFYDELAY value.

If LISTNOTIFYDELAY is specified and the operational level of the coupling facility in which the structure is allocated (see ConaCFacilityCFLevel in the IXLCONN Connect Answer Area) is less than

CFLEVEL=22, the WRITE\_LCONTROLS request fails with return code, `Ix1RetCodeEnvError`, and reason code, `Ix1RsnCodeBadRegCflevel`.

**NO\_CHANGE**

The list notification delay value for the list remains unchanged.

**ON**

The list notification delay value for the structure is applied when notifying monitoring instances of empty to not-empty state transitions for this list.

**OFF**

The list notification delay value for the structure is not applied when notifying monitoring instances of empty to not-empty state transitions for this list.

**,LISTNUM=NO\_LISTNUM****,LISTNUM=listnum**

Depending on the IXLLSTC request type, use this input parameter to specify the number of the list to be processed.

- For REQUEST=READ\_LCONTROLS, the number of the list to be processed.
- For REQUEST=WRITE\_LCONTROLS, the number of the list to be processed.
- For REQUEST=READ\_EMCONTROLS, the partial designation of the sublist to be processed.

LISTNUM is used in conjunction with either ENTRYKEY or SECONDARYKEY to designate a sublist for which the user's event monitor controls are to be read.

- For REQUEST=MONITOR\_LIST, the number of the list to be monitored.
- For REQUEST=MONITOR\_KEYRANGE, the number of the list whose key-range is to be monitored.
- For REQUEST=MONITOR\_SUBLIST, the partial designation of the sublist to be processed.

LISTNUM may be used in conjunction with ENTRYKEY or SECONDARYKEY to designate a sublist for which the user wishes to start or stop sublist monitoring.

**To Code:** Specify the RS-type name or address (using a 2 register from 2 to 12) of a fullword field that contains the appropriate list or sublist value.

**,LISTSTATE=NO\_CHANGE****,LISTSTATE=DEFINE**

For REQUEST=WRITE\_LCONTROLS, use this input parameter to specify whether the threshold counts that define the empty or not-empty state of a list are to be updated.

**NO\_CHANGE**

The threshold counts will remain unchanged.

**DEFINE**

The threshold counts are to be set to the values indicated by LISTEMPTY and LISTNOTEMPTY.

Upon completion of the definition of the list empty or not-empty thresholds, the list state may change. If the number of list entries is less than the new list empty thresholds, the list state is empty. If the number of list entries is greater than the new list not-empty threshold, the list state is not-empty. When the number of list entries is greater than or equal to the new list empty threshold and less than or equal to the new list not-empty threshold, the list state remains unchanged.

DEFINE is valid for coupling facilities of CFLEVEL=9 or higher. DEFINE is ignored and the request processed as if NO\_CHANGE were specified when the structure is allocated in a coupling facility of CFLEVEL=8 or lower.

A list is either in the empty state or the not-empty state. Use a REQUEST=MONITOR\_LIST request to register interest in monitoring transitions between list states.

The default list empty or not-empty threshold counts are initialized to zero for all lists when the structure is allocated. For structures allocated in a coupling facility of CFLEVEL=8 or lower, the list-empty and list-not-empty threshold counts are always zero.

**,LOCKCOMP=NO LOCKCOMP****,LOCKCOMP=lockcomp**

This parameter has slightly different meanings based on the value specified for the LOCKOPER parameter. Generally, this input parameter specifies a connection identifier to be verified as the owner of the lock specified by LOCKINDEX. This verification is a prerequisite to the successful completion of this request.

When LOCKCOMP is specified, the completion of the request is conditional on there being no contention for the lock. If contention exists, the request will fail.

The connection identifier is available from the IXLCONN answer area, mapped by IXLYCONA, in field CONACONID.

The effect of LOCKCOMP is based on the LOCKOPER value specified. See the description of the LOCKOPER values to see how each request is affected by this parameter.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 1-byte field that contains the connection identifier.

**,LOCKDATA=NO LOCKDATA****,LOCKDATA=lockdata**

Use this input parameter to specify user information that is to be passed to the Notify exit for the connection when the lock is held because of a LOCKOPER=SET operation and another connection issues one of the following requests:

- An IXLLSTC LOCK request specifying LOCKOPER=SET with LOCKMODE=UNCOND
- An IXLLSTE request specifying LOCKOPER=SET with LOCKMODE=UNCOND
- An IXLLSTM request specifying LOCKOPER=NOTHELD

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-byte field that contains the user-specified information.

**,LOCKINDEX=lockindex**

For REQUEST=LOCK, use this input parameter to specify the index of the lock within the lock table for the list structure that is to be operated on as specified by the LOCKOPER keyword. The index value must fall within the range 0 to the number of lock table entries minus one, inclusive.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword field that contains the lock index.

**,LOCKMODE=UNCOND****,LOCKMODE=COND**

Use this input parameter to specify how contention on the lock is to be handled.

**UNCOND**

The lock operation will be performed unconditionally. If the specified lock is held, the IXLLSTC request will be held up until the operation can be processed.

- If MODE=SYNCSUSPEND is specified, the caller is suspended until the lock becomes available.
- If any MODE value other than SYNCSUSPEND is specified, the request is processed asynchronously and the caller is notified by the means indicated on the MODE keyword.

**COND**

The lock operation is performed conditionally. If the lock is held, the IXLLSTC request is terminated with no resultant change to the structure, and indicative return and reason codes are returned to the caller.

**,LOCKOPER=SET****,LOCKOPER=RESET****,LOCKOPER=TEST****,LOCKOPER=READNEXT**

For REQUEST=LOCK, use this input parameter to specify the type of operation to be performed on the lock specified by LOCKINDEX.

**SET**

- When LOCKCOMP is not specified, requests ownership of the lock for the connection specified by CONTOKEN.
- When LOCKCOMP is specified, requests that ownership of the lock be transferred from the connection specified by LOCKCOMP to the connection specified by CONTOKEN. Any outstanding requests awaiting contention resolution on this lock are ignored.

**RESET**

- When LOCKCOMP is not specified, requests that ownership of the lock be released if held by the connection specified by CONTOKEN.
- When LOCKCOMP is specified, requests that ownership of the lock be released if it is held by the connection specified by LOCKCOMP.

The RESET operation will always be done unconditionally when LOCKCOMP is not specified and conditionally when LOCKCOMP is specified.

**TEST**

- When LOCKCOMP is not specified, requests that the lock be tested to see if it is owned by the connection specified by CONTOKEN.
- When LOCKCOMP is specified, requests that the lock be tested to see if it is owned by the LOCKCOMP connection.

In both cases return code X'0' indicates that the lock was held and return code X'4' indicates that the lock is either not held or is held by a different connection.

The lock state remains unchanged as a result of this option.

**READNEXT**

- When LOCKCOMP is not specified, requests that the lock index and connection ID of the owner of the next owned lock starting at the lock index specified by LOCKINDEX be returned.
- When LOCKCOMP is specified, requests that the lock index of the next lock owned by the LOCKCOMP connection, starting at the lock index specified by LOCKINDEX, be returned.

The lock state remains unchanged as a result of this option.

A request specifying LOCKOPER=READNEXT may complete prematurely if coupling facility model-dependent timeout criteria is exceeded. In this event, indicative return and reason codes are provided, and the index of the next lock to be processed is returned in the answer area specified by ANSAREA. This lock index can be specified on a subsequent LOCKOPER=READNEXT request to resume processing with the appropriate lock entry.

**,MAXLISTKEY=NO\_MAXLISTKEY****,MAXLISTKEY=maxlistkey**

For REQUEST=WRITE\_LCONTROLS, use this input parameter to specify the maximum list key value to be associated with the list. This value specifies an upper bound for the list key value.

If MAXLISTKEY is not specified, the maximum list key for the designated list will remain unchanged. MAXLISTKEY is ignored when the structure does not support keyed entries.

**Note:** The default maximum list key for all lists when the structure is allocated is binary zeros.

The MAXLISTKEY keyword is only meaningful for list structures allocated in a coupling facility of CFLEVEL=1 or higher.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-byte field that contains the maximum list key value to be associated with the list.

```
,MF=S
,MF=(L,mfctrl)
,MF=(L,mfctrl,mfattr)
,MF=(L,mfctrl,0D)
,MF=(M,mfctrl)
,MF=(M,mfctrl,COMPLETE)
,MF=(M,mfctrl,NOCHECK)
,MF=(E,mfctrl)
,MF=(E,mfctrl,COMPLETE)
,MF=(E,mfctrl,NOCHECK)
```

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

#### **,mfctrl**

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

#### **,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

#### **,COMPLETE**

#### **,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

#### **COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if SMILE=*var* were an optional parameter and the default is SMILE=NO\_SMILE then it would not be documented. However, if the default was SMILE=:), then it would be documented because a value would be the default.

#### **NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

,**MODE=SYNCSUSPEND**  
 ,**MODE=SYNCECB**  
 ,**MODE=SYNCEXIT**  
 ,**MODE=SYNCTOKEN**  
 ,**MODE=ASYNCECB**  
 ,**MODE=ASYNCEXIT**  
 ,**MODE=ASYNCTOKEN**  
 ,**MODE=ASYNCSNORESPONSE**

Use this input parameter to specify whether the request is to be performed synchronously or asynchronously.

#### **SYNCSUSPEND**

The request is performed synchronously. Control is not returned to the caller until request processing is complete and the final disposition determined.

If necessary, the caller will be suspended until the request completes. The caller must be executing in an enabled state to use this option.

#### **SYNCECB**

The request will be attempted synchronously. If the request cannot be completed synchronously, control is returned to the caller prior to completion of the request, and the ECB specified by REQECB is posted when the request has completed.

If the request does not complete synchronously, latent XES binds to the storage locations specified by BUFFER, BUFLIST, MOSVECTOR, and ANSAREA persist until the REQECB ECB is posted.

#### **SYNCEXIT**

The request will be attempted synchronously. If the request cannot be completed synchronously, control is returned to the caller prior to completion of the request. When the request completes, the connection's Complete Exit will be called.

If the request does not complete synchronously, latent XES binds to the storage locations specified by BUFFER, BUFLIST, MOSVECTOR, and ANSAREA persist until the connection's Complete Exit is called.

#### **SYNCTOKEN**

The request will be attempted synchronously. If the request cannot be completed synchronously, control is returned to the caller prior to completion of the request and a token that uniquely identifies the request is returned. This token must be specified on a subsequent invocation of IXLFCOMP to force completion of the request and determine its final disposition.

If the request does not complete synchronously, latent XES binds to the storage locations specified by BUFFER, BUFLIST, MOSVECTOR, and ANSAREA persist until a subsequent corresponding IXLFCOMP request indicates completion of the original request.

#### **ASYNCECB**

The request is to be initiated and control is to be returned to the caller prior to completion of the request. When the request completes, the ECB specified by REQECB will be posted.

Latent XES binds to the storage locations specified by BUFFER, BUFLIST, MOSVECTOR, and ANSAREA persist until the REQECB ECB is posted.

#### **ASYNCEXIT**

The request is to be initiated and control is to be returned to the caller prior to completion of the request. When the request completes, the connection's Complete Exit will be called.

Latent XES binds to the storage locations specified by BUFFER, BUFLIST, MOSVECTOR, and ANSAREA persist until the connection's Complete Exit is called.

#### **ASYNCTOKEN**

The request is to be initiated, a token generated that uniquely identifies the request on this system, and control returned to the caller prior to completion of the requested operation. The



token must be specified on a subsequent invocation of IXLFCOMP to force completion of the request and determine its final disposition.

Latent XES binds to the storage locations specified by BUFFER, BUFLIST, MOSVECTOR, and ANSAREA persist until a subsequent corresponding IXLFCOMP request indicates completion of the original request.

### **ASYNCRESPONSE**

The request is to be initiated and control returned to the caller prior to completion of the requested operation. No asynchronous request token is returned, hence no external mechanism exists to force completion of the request.

MODE=ASYNCRESPONSE is mutually exclusive with LOCK, READ\_LCONTROLS, DEQ\_EVENTQ, and MONITOR\_SUBLISTS requests. It is also mutually exclusive with REQUEST=MONITOR\_LIST and REQUEST=MONITOR\_KEYRANGE when ACTION=START is specified. Any other request can specify MODE=ASYNCRESPONSE.

### **,MONITORTYPE=NOTEEMPTY**

### **,MONITORTYPE=NOTFULL**

For REQUEST=MONITOR\_LIST, use this input parameter to specify the type of monitoring requested. A user can monitor for either empty/not-empty or full/not-full list transitions, but not both for the same list.

A user can dynamically change the type of list monitoring that is performed for a list, without stopping monitoring, by re-issuing a MONITOR\_LIST ACTION=START request for the same list with the opposite list notification monitor type.

The macro, IXCYQUAA, defines the QuReqRfNotFullMonitoring bit in the QuReqFeatures string that can be used to test if list not-full monitoring support and the IXLLSTC MONITORTYPE keyword are supported on the system. Use IXCQUERY REQINFO=FEATURES to retrieve the QuReqFeatures string.

### **NOTEEMPTY**

Requests that the list that is specified by LISTNUM be monitored for empty/not-empty transitions. While list monitoring is in effect, the IXLVECTR service can be used to determine whether the list is considered empty or not-empty according to the thresholds established with a WRITE\_LCONTROLS request.

### **NOTFULL**

Requests that the list that is specified by LISTNUM be monitored for full/not-full transitions. While list monitoring is in effect, the IXLVECTR service can be used to determine whether the list is considered full or not-full according to the list limits established with a WRITE\_LCONTROLS request.

Monitoring for a full/not-full transition is valid only when the structure is allocated in a CFLEVEL=22 or higher coupling facility.

If MONITORTYPE=NOTFULL is specified and the operational level of the coupling facility in which the structure is allocated (see ConaCFacilityCFLevel) is less than CFLEVEL=22, the MONITOR\_LIST request fails with return code, Ix1RetCodeEnvError, and reason code, Ix1RsnCodeBadReqCflevel.

### **,MOSVECTOR=*mosvector***

Use this input parameter to specify a 1-origin bit string in which each bit represents the monitored object state (empty or not-empty) of a particular sublist at the time that the REQUEST=MONITOR\_SUBLISTS request was processed.

The MOSVECTOR bits correspond one-to-one with the IXLYMSRI entries that were passed as input in the BUFFER or BUFLIST. Only the bits corresponding to the IXLYMSRI entries that were processed on the current request (that is, between STARTINDEX and ENDINDEX for a request that completed successfully, or between STARTINDEX and the returned index of the first unprocessed entry minus one for premature completion cases) contain valid monitored object state information for the sublists that are designated by the corresponding IXLYMSRI entries. Bits in the MOSVECTOR that lie outside the valid range are not meaningful.

When a bit in the valid range is on, the sublist designated by the corresponding IXLYMSRI entry was not-empty at the time the request was processed. When a bit in the valid range is off, the sublist designated by the corresponding IXLYMSRI entry was empty at the time the request was processed.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 128-byte field that contains a 1-origin bit string in which each bit represents the monitored object state of a particular sublist.

**,NEWAUTH=NO\_NEWAUTH**

**,NEWAUTH=newauth**

For REQUEST=WRITE\_LCONTROLS, use this input parameter to specify a new value to be established as the list authority of the designated list. If NEWAUTH is not specified, the list authority for the designated list will remain unchanged.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-byte field that contains the list authority value.

**,NOTIFICATION=FIRST**

**,NOTIFICATION=EVERY**

Depending on the IXLLSTC request type, use the input parameter to specify:

- For REQUEST=MONITOR\_KEYRANGE, the conditions under which the coupling facility updates the list notification vector for keyrange monitoring for the list that is identified by LISTNUM.
- For REQUEST=MONITOR\_LIST, the conditions under which the coupling facility updates the list notification vector for empty/not-empty list state transitions for the list that is identified by LISTNUM.
- For REQUEST=MONITOR\_SUBLIST, the condition for which the coupling facility is to queue an EMC to the registered user's event queue.

**FIRST**

**REQUEST=MONITOR\_KEYRANGE**

Indicates that the list notification vector index that is associated with the monitored keyrange of a list that is identified by LISTNUM is updated whenever the keyrange state transitions from the not-empty state to empty state and when the keyrange state transitions from the empty state to not-empty state.

**REQUEST=MONITOR\_LIST**

Indicates that the list notification vector index that is associated with the monitored list is updated whenever the list state transitions from the not-empty state to empty state and when the list state transitions from the empty state to not-empty state.

**REQUEST=MONITOR\_SUBLIST**

Indicates that an EMC is to be queued to the registered user's event queue when the monitored sublist transitions from empty to not-empty.

**EVERY**

**REQUEST=MONITOR\_KEYRANGE**

Indicates that the list notification vector index that is associated with the monitored keyrange of a list that is identified by LISTNUM is updated whenever the keyrange state transitions from the not-empty state to empty state. It is also updated every time a list entry is added to the keyrange and the keyrange not-empty threshold count is exceeded.

NOTIFICATION=EVERY is valid only when the structure is allocated in a CFLEVEL=22 or higher coupling facility.

If NOTIFICATION=EVERY is specified and the operational level of the coupling facility in which the structure is allocated (see ConaCFacilityCFLevel) is less than CFLEVEL=22, the MONITOR\_KEYRANGE request fails with return code, Ix1RetCodeEnvError, and reason code, Ix1RsnCodeBadReqCflevel.

The macro IXCYQUAA defines the QuReqRfAggressiveNotify bit in the QuReqFeatures string that can be used to test if aggressive keyrange monitoring (the IXLLSTC

NOTIFICATION=EVERY keyword for REQUEST=MONITOR\_KEYRANGE) is supported on the system. Use IXCQUERY REQINFO=FEATURES to retrieve the QuReqFeatures string.

### **REQUEST=MONITOR\_LIST**

Indicates that the list notification vector index that is associated with the monitored list is updated whenever the list state transitions from the not-empty state to empty state and every time a list entry is added to the list and the list not-empty threshold count is exceeded.

NOTIFICATION=EVERY is valid only when the structure is allocated in a CFLEVEL=22 or higher coupling facility.

If NOTIFICATION=EVERY is specified and the operational level of the coupling facility in which the structure is allocated (see ConaCFacilityCFLevel) is less than CFLEVEL=22, the MONITOR\_LIST request fails with return code, Ix1RetCodeEnvError, and reason code, Ix1RsnCodeBadReqCflevel.

The macro IXCYQUAA defines the QuReqRfAggressiveNotify bit in the QuReqFeatures string that can be used to test if aggressive list monitoring (the IXLLSTC NOTIFICATION=EVERY keyword for REQUEST=MONITOR\_LIST) is supported on the system. Use IXCQUERY REQINFO=FEATURES to retrieve the QuReqFeatures string.

### **REQUEST=MONITOR\_SUBLIST**

Indicates that an EMC is to be queued to the registered user's event queue whenever a list entry is queued to the monitored sublist and an EMC for the monitored sublist interest is not already queued to the registered user's event queue.

NOTIFICATION=EVERY is valid only when the structure is allocated in a CFLEVEL=9 or higher coupling facility.

### **,PAGEABLE=YES**

### **,PAGEABLE=NO**

Use this input parameter to identify whether the storage areas specified by BUFFER or BUFLIST reside in pageable storage.

#### **YES**

Specify this option to indicate that the BUFFER or BUFLIST buffers reside in pageable virtual storage.

This includes disabled reference (DREF) storage, and may include storage that has the potential to become pageable during the processing of a request. (An example is address space storage owned by any swappable address space, for which a PGSER FIX has been successfully processed, but for which the owning address space gets swapped during processing of a cache or list request.) It does not include implicitly non-pageable storage (such as is obtained from non-pageable subpools).

The system takes responsibility for managing binds to central storage for the duration of the list request, regardless of what address space owns the storage or whether the storage-owning address space is swappable or nonswappable. The storage can be owned by any address space.

#### **NO**

Specify this option to indicate that the BUFFER or BUFLIST buffers reside in non-pageable virtual storage.

This includes implicitly non-pageable storage areas. If the virtual storage may potentially become pageable, the invoker is responsible for ensuring the virtual storage remains non-pageable for the duration of the request. (An example is address space storage owned by any swappable address space, for which a PGSER FIX has been successfully processed, but for which the owning address space gets swapped-out during processing of a list request.)

If MODE=ASYNCTOKEN is specified or MODE=SYNCTOKEN is specified and the request does not complete synchronously, the storage must remain non-pageable until completion of the corresponding IXLFCOMP request. If MODE=ASYNCEXIT is specified or MODE=SYNCEXIT is specified and the request does not complete synchronously, the storage must remain non-pageable until the complete exit is driven for the request. If MODE=ASYNCECB is specified or

MODE=SYNCECB is specified and the request does not complete synchronously, the storage must remain non-pageable until the specified ECB is posted for the request.

The system takes responsibility for managing binds to central storage for the duration of the list request, if and only if the non-pageable storage is owned by either the requester's address space or the connector's address space. If the storage is owned by any other address space, then the invoker is responsible for ensuring that the virtual storage remains non-pageable for the duration of the request (including the case in which the storage is owned by a swappable address space that is swapped during processing of a list request). Subject to this consideration, the storage can be owned by any address space. See [z/OS MVS Programming: Sysplex Services Guide](#).

**,PLISTVER=IMPLIED\_VERSION**

**,PLISTVER=MAX**

**,PLISTVER=plistver**

Use this input parameter to specify the version of the macro. See “[Understanding IXLLSTC Version Support](#)” on page 1374 for a description of the options available with the PLISTVER macro.

**,REQDATA=NO\_REQDATA**

**,REQDATA=reqdata**

Use this input parameter with MODE=SYNCEXIT or MODE=ASYNCEXIT to pass any data you choose to the complete exit. The exit will get control only if the request is processed asynchronously.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the data to be passed to the complete exit.

**,REQECB=reqecb**

Use this output parameter with either MODE=SYNCECB or MODE=ASYNCECB to specify the address of an ECB, which is to be posted when the request completes if the request was processed asynchronously.

Before coding REQECB, you must ensure that:

- You initialize the ECB before you issue the request.
- The ECB resides in either common storage or the home address space where IXLCONN was issued.
- Any tasks that wait for the ECB to be posted reside in the home address space where IXLCONN was issued.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that contains the address of the ECB to be posted when the request completes. The ECB must be aligned on a fullword boundary.

**,REQID=NO\_REQID**

**,REQID=reqid**

Use this input parameter to specify a user-defined request identifier to be associated with the request. You can specify this request identifier on the IXLPURGE macro to cancel a request that has not yet been processed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the user-defined request identifier.

**,REQTOKEN=reqtoken**

Use this output parameter with either MODE=SYNCTOKEN or MODE=ASYNCTOKEN to specify the address of a storage area to receive the request token that is returned when the request will be processed asynchronously. This token, which uniquely identifies the request, must be used as input to the IXLFCOMP macro, which you use to determine if the request has completed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field where the system will put the request token.

**,REQUEST=LOCK**  
**,REQUEST=READ\_LCONTROLS**  
**,REQUEST=WRITE\_LCONTROLS**  
**,REQUEST=READ\_EQCONTROLS**  
**,REQUEST=READ\_EMCONTROLS**  
**,REQUEST=DEQ\_EVENTQ**  
**,REQUEST=MONITOR\_EVENTQ**  
**,REQUEST=MONITOR\_LIST**  
**,REQUEST=MONITOR\_KEYRANGE**  
**,REQUEST=MONITOR\_SUBLIST**  
**,REQUEST=MONITOR\_SUBLISTS**  
**,REQUEST=READ\_STRCOUNTS**

Use this input parameter to specify the type of operation to be performed on the structure.

#### **LOCK**

The lock entry designated by LOCKINDEX is to be operated on as specified by the LOCKOPER keyword. This request type may only be specified for structures that contain a lock table.

REQUEST=LOCK is mutually exclusive with MODE=ASYNCHRESPONSE.

#### **READ\_LCONTROLS**

Read the control information for the list specified by the LISTNUM keyword. The information returned in the answer area specified by ANSAREA consists of the list controls as mapped by the IXLYLAA macro. In addition, the list monitoring information and key-range monitoring information for the list, as mapped by the IXLYLMI macro, is returned in the storage specified by BUFFER or the buffers specified by BUFLIST.

REQUEST=READ\_LCONTROLS is mutually exclusive with MODE=ASYNCHRESPONSE.

#### **WRITE\_LCONTROLS**

Update one or more of the list controls for the list specified by LISTNUM. The list controls that can be updated include:

- List authority (NEWAUTH)
- List limit bounding the number of entries or elements that may reside on the list (LISTLIMIT)
- List descriptor (LISTDESC)
- List key (LISTKEY)
- Maximum list key (MAXLISTKEY)
- List cursor and list cursor direction (SETCURSOR)
- Key-range start and end values (KEYRANGESTART and KEYRANGEEND)
- Key-range empty and key-range not-empty threshold counts (KREMPY and KRNOTEMPTY)
- List empty and list not-empty threshold counts (LISTEMPTY and LISTNOTEMPTY)

#### **READ\_EQCONTROLS**

Read event queue control information for the requesting user's event queue. Event queue processing is used in conjunction with sublist monitoring and requires that the list structure be allocated in a coupling facility of CFLEVEL=3 or higher. For list structures allocated in a coupling facility of CFLEVEL=9 or higher, a sublist can be identified either by entry key or by secondary key. Event monitor controls (EMCs) for sublists identified by entry key are queued to a different event queue than EMCs for sublists identified by secondary key. The different event queues can be monitored independently. The KEYTYPE keyword designates which event queue is the subject of the READ\_EQCONTROLS request.

Information returned in the answer area specified by ANSAREA consists of the following:

- The total number of event monitor controls (EMCs) currently on the event queue
- The number of times that the event queue has transitioned from the empty to the non-empty state
- The vector index number that is being used to monitor the event queue

- An indication of whether or not the user's list transition exit is to be driven when the event queue transitions from the empty to the not-empty state
- An indication of whether the EMCs are for sublists identified by entry key or for sublists identified by secondary key.

**READ\_EMCONTROLS**

Read the EMC control information for the requesting user's registered interest in monitoring a particular sublist designated by the LISTNUM keyword and either ENTRYKEY or SECONDARYKEY.

- For entry keys, this request type is valid only when the structure is allocated in a coupling facility of CFLEVEL=3 or higher. The structure must support keyed list entries (REFOPTION=KEY must be specified on the IXLCONN request when the structure is allocated).
- For secondary keys, this request type is valid only when the structure is allocated in a coupling facility of CFLEVEL=9 or higher. The structure must support secondary keys (KEYTYPE=SECONDARY must be specified on the IXLCONN request when the structure is allocated).

Information returned in the answer area specified by ANSAREA includes the following:

- The user notification controls (UNC)
- The list number and entry key or secondary key of the sublist with which the event monitor control information is associated
- An indication of whether or not the EMC is currently queued to the user's event queue
- An indication of whether the EMCs are for sublists identified by entry key or for sublists identified by secondary key
- An indication of whether the EMC is queued to the event queue for only the first list entry added to the sublist or for every list entry added to the sublist.

**DEQ\_EVENTQ**

Dequeue the event monitor controls from an event queue. The sublist associated with the event queue can be identified either by list entry key or by secondary key. Identification by list entry key requires that the list structure be allocated in a coupling facility of CFLEVEL=3 or higher; identification by secondary key requires that the list structure be allocated in a coupling facility of CFLEVEL=9 or higher. If the structure is allocated with secondary keys, there are two event queues that can be monitored. The KEYTYPE keyword designates which event queue is the subject of the DEQ\_EVENTQ request.

The contents of the dequeued EMCs are returned in the specified BUFFER or BUFLIST area. Each individual EMC is mapped by the IXLYEMC macro. Note that neither the EMCs nor the sublist monitoring interest which they represent is deleted by this request. Sublist monitoring remains active until it is stopped by a subsequent MONITOR\_SUBLIST ACTION=STOP request.

This request type is valid only when the structure is allocated in a coupling facility of CFLEVEL=3 or higher (for entry keys) or CFLEVEL=9 or higher (for secondary keys).

The DEQ\_EVENTQ request may complete prematurely, that is, without having dequeued all of the EMCs from an event queue. In such a case, the user is expected to process the EMCs that were returned on the current request and then re-issue the DEQ\_EVENTQ request to continue the process of dequeuing the remaining EMCs from the event queue.

Information returned in the answer area specified by ANSAREA consists of the number of EMCs that were returned and the number of EMCs that still remain on the user's event queue.

REQUEST=DEQ\_EVENTQ is mutually exclusive with MODE=ASYNCRESPONSE.

**MONITOR\_EVENTQ**

Start or stop list notification vector monitoring of an event queue for a requesting user.

This request type is valid only when the structure is allocated in a coupling facility of CFLEVEL=3 or higher. The user must have a local vector associated with the connection to the structure, and the structure must support keyed list entries. If the structure is allocated in a coupling facility of CFLEVEL=9 or higher and is allocated with secondary keys, there are two event queues that

can be monitored. The KEYTYPE keyword designates which event queue is the subject of the MONITOR\_EVENTQ request.

While event queue monitoring is in effect for an event queue, the IXLVECTR service can be used to determine whether the event queue contains any event monitor controls (EMCs). These EMCs represent not-empty sublists in which the user has registered sublist monitoring interest. (See Chapter 85, “IXLVECTR — Check or Modify Local Cache Vector or List Notification Vector,” on page 1605.) Use event queue monitoring in conjunction with sublist monitoring (MONITOR\_SUBLIST or MONITOR\_SUBLISTS) to process state transitions for monitored sublists. Use the DEQ\_EVENTQ request to dequeue queued EMCs from an event queue and to read the contents of the EMCs.

If a list notification vector index is in use for monitoring, a request to stop monitoring should be performed before using the same vector index to start another monitor. If event queue monitoring is in effect, a request to start monitoring the same event queue with a different VECTORINDEX or DRIVEEXIT specification will cause the old specifications to be immediately replaced by the new specifications. It is not necessary to stop event queue monitoring before requesting to start monitoring the same event queue using the new specifications. However, because the replaced vector index is no longer being used to monitor an event queue, it may be reassigned for other uses (for example, to monitor a list).

When event queue monitoring is stopped, the state of the vector index that was previously being used to monitor the event queue is not defined. However, because the vector index is no longer being used to monitor the event queue, it may be reassigned for other uses (for example, to monitor a list).

## **MONITOR\_LIST**

Start or stop monitoring the list specified by LISTNUM.

While list monitoring is in effect for a list, the IXLVECTR service can be used to determine whether the list is considered empty or not-empty according to the thresholds established with a WRITE\_LCONTROLS request.

The IXLVECTR service can also be used to alter the size of the list notification vector, and thus changed the number of lists, key-ranges, or event queues that can be monitored concurrently. (See Chapter 85, “IXLVECTR — Check or Modify Local Cache Vector or List Notification Vector,” on page 1605.)

If a list notification vector index is in use for monitoring, a request to stop monitoring should be performed before using the same vector index to start another monitor. If a vector index is in use for monitoring a list, a request to start list monitoring for the same list with a different vector index will cause the vector index in use for the list to be replaced by the new vector index. In this case, it is not necessary to stop monitoring using the old index before requesting to start monitoring using the new index.

REQUEST=MONITOR\_LIST with ACTION=START is mutually exclusive with MODE=ASYNCRESPONSE.

REQUEST=WRITE\_LCONTROLS can be used to set the empty or not-empty threshold counts that are used to define the empty or not-empty state of a list for list monitoring.

## **MONITOR\_KEYRANGE**

Start or stop key-range monitoring of the list specified by LISTNUM. This request type is valid only for keyed list structures allocated in a coupling facility of CFLEVEL=9 or higher.

While key-range monitoring is in effect for a list, the IXLVECTR service can be used to determine whether the key-range is considered empty or not-empty according to the thresholds established with a WRITE\_LCONTROLS request that specifies KEYRANGESTATE=DEFINE.

The IXLVECTR service can also be used to alter the size of the list notification vector, and thus change the number of lists, key-ranges, or event queues that can be monitored concurrently. (See Chapter 85, “IXLVECTR — Check or Modify Local Cache Vector or List Notification Vector,” on page 1605.)

If a list notification vector index is in use for monitoring, a request to stop monitoring should be performed before using the same vector index to start another monitor. If a vector index is in use for monitoring a key-range, a request to start key-range monitoring of the same list with a different vector index will cause the vector index in use for the key-range to be replaced by the new vector index. In this case, it is not necessary to stop monitoring using the old index before requesting to start monitoring using the new index.

Only one key-range can be monitored per list. REQUEST=WRITE\_LCONTROLS can be used to set the starting and ending key-range values, and the empty or not-empty threshold counts that define the empty or not-empty state of a key-range for key-range monitoring.

REQUEST=MONITOR\_KEYRANGE with ACTION=START is mutually exclusive with MODE=ASYNCRESPONSE.

A request to start monitoring a key-range for a list can timeout if definition of the key-range is not complete (see REQUEST=WRITE\_LCONTROLS with KEYRANGESTATE=DEFINE). The caller is expected to reissue the request until it completes without timing out.

### **MONITOR\_SUBLIST**

Start or stop monitoring a sublist designated by a list number and either an entry key or secondary key within the list structure.

A request to monitor a keyed sublist is valid only when the structure is allocated in a coupling facility of CFLEVEL=3 or higher. The user must have a local vector associated with the connection to the structure, and the structure must support keyed list entries (REFOPTION=KEY must be specified on IXLCONN when the structure is allocated).

A request to monitor a secondary keyed sublist is valid only when the structure is allocated in a coupling facility of CFLEVEL=9 or higher. The user must have a local vector associated with the connection to the structure, and the structure must support secondary keys (KEYTYPE=SECONDARY must be specified on IXLCONN when the structure is allocated).

### **MONITOR\_SUBLISTS**

Start monitoring a set of sublists designated by list number and either entry key or secondary key in a list structure. MONITOR\_SUBLISTS cannot be used to stop sublist monitoring.

A request to monitor a sublist identified by a list entry key is valid only when the structure is allocated in a coupling facility of CFLEVEL=3 or higher. A request to monitor a sublist identified by a secondary key is valid only when the structure is allocated in a coupling facility of CFLEVEL=9 or higher and the structure supports secondary keys. In both cases, the user must have a local vector associated with the connection to the structure, and the structure must support keyed list entries.

REQUEST=MONITOR\_SUBLISTS is mutually exclusive with MODE=ASYNCRESPONSE.

### **READ\_STRCOUNTS**

Request the information on the maximum and in-use counts be returned.

Information returned in the answer area specified by ANSAREA mapped by LAARSTC in the IXLYLAA consists of the following:

- The current in-use number of elements and maximum number of elements defined for the structure if data elements are defined for the structure.
- The current in-use number of entries and the maximum number of entries defined for the structure.
- The current in-use number of event monitor controls and the total number of event monitor controls defined for the structure if event monitor controls are supported by the structure.
- The number of lock entries defined for the structure if the structure is defined as a serialized list structure.

### **,RETCODE=retcode**

Use this output parameter to specify a field to contain the return code. (The return code is also returned in register 15.)



**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the return code when the request has completed.

**,RSNCODE=rsncode**

Use this output parameter to specify a field to contain the reason code returned, if applicable. (The reason code is also returned in register 0.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the reason code (if any) when the request has completed.

**,SECONDARYKEY=secondarykey**

Depending on the IXLLSTC request type, use this input parameter to specify the unsigned 256-bit secondary key to be used in conjunction with LISTNUM to designate:

- For REQUEST=READ\_EMCONTROLS, a sublist whose EMCs are to be read.
- For REQUEST=MONITOR\_SUBLIST, a sublist whose monitoring is to be started or stopped.

SECONDARYKEY is valid only for structures that are allocated in a coupling facility of CFLEVEL=9 or higher. The structure must support secondary keys.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 32-byte field that contains the unsigned 256-bit secondary key.

**,SETCURSOR=NO\_SETCURSOR**

**,SETCURSOR=HEAD**

**,SETCURSOR=TAIL**

For REQUEST=WRITE\_LCONTROLS, use this input parameter to specify that the list cursor and the list cursor direction are to be set.

The list cursor direction is only used when an IXLLSTE invocation specifies UPDATECURSOR=YES with CURSORUPDATETYPE=NEXTCOND.

**Note:** The default list cursor for all lists when the structure is allocated is binary zeros. The default list cursor direction for all lists when the structure is allocated is a head-to-tail direction.

The SETCURSOR keyword is only meaningful for list structures allocated in a coupling facility of CFLEVEL=1 or higher.

**NO\_SETCURSOR**

The list cursor and the list cursor direction will remain unchanged.

**HEAD**

The list cursor is to be set to the entry identifier for the first entry on the list and the list cursor direction is to be set in a head-to-tail direction. If the list is empty, the list cursor will be reset to binary zeros.

**TAIL**

The list cursor is to be set to the entry identifier for the last entry on the list and the list cursor direction is to be set in a tail-to-head direction. If the list is empty, the list cursor will be reset to binary zeros.

**,STARTINDEX=startindex**

For REQUEST=MONITOR\_SUBLISTS, use this input parameter to specify the 1-origin index number of the first IXLYMSRI entry that is requested to be processed. The valid range is from 1 to the ENDINDEX value, inclusive.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a halfword field that contains the 1-origin index number of the first entry to be processed.

**,UNC=unc**

For REQUEST=MONITOR\_SUBLIST, use this input parameter to specify the user notification controls (UNC) that represent the user's monitoring interest in the designated sublist. The user notification control information resides in the event monitor controls (EMC), which is queued to an event queue when the monitored sublist becomes not-empty, and is returned along with other information from the EMC on a REQUEST=DEQ\_EVENTQ.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-byte field that contains the user notification controls (UNC).

**,VECTORINDEX=vectorindex**

Depending on the IXLLSTC request, use this input parameter to specify a vector index.

- For REQUEST=MONITOR\_EVENTQ, specify the vector index to be associated with the monitored event queue. If the request completes successfully, this local vector index in the vector for the connection specified by CONTOKEN will subsequently reflect the empty or not-empty state of the user's event queue.
- For REQUEST=MONITOR\_LIST, specify the vector index that is to reflect the empty or not-empty state of the monitored list. If the request completes successfully, this local notification vector index in the vector for the connection specified by CONTOKEN will subsequently reflect the empty or not-empty state of the monitored list.
- For REQUEST=MONITOR\_KEYRANGE, specify the vector index that is to reflect the empty or not-empty state of the monitored key-range. If the request completes successfully, this local notification vector index in the vector for the connection specified by CONTOKEN will subsequently reflect the empty or not-empty state of the monitored key-range.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword field that contains the list notification vector index.

## ABEND Codes

---

Abend X'026' (See *z/OS MVS System Codes* for more information on this abend.)

## Return and Reason Codes

---

When the system returns control to your program:

- GPR 15 (and RETCODE, if specified) contains a return code.
- If the return code is not zero, GPR 0 (and RSNCODE, if specified) contains a reason code.

**Note:** The return and reason codes will also be put into the answer area (mapped by IXLYLAA), if there is one, when the request completes.

The IXLYCON macro provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

<b>0</b>	IXLRETCODEOK
<b>4</b>	IXLRETCODEWARNING
<b>8</b>	IXLRETCODEPARMERROR
<b>C</b>	IXLRETCODEENVERROR
<b>10</b>	IXLRETCODECOMPERROR

The following table contains hexadecimal return and reason codes, the equate symbols associated with each reason code, and the meaning and suggested action for each return and reason code.

Table 78. Return and Reason Codes for IXLLSTC Macro

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0	None.	<p><b>Meaning:</b> If you specified a MODE value of ASYNCECB, ASYNCEXIT, ASYNCTOKEN, or ASYNCNORESPONSE, the request has been successfully initiated. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed. For any other MODE value, the request has successfully completed.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=ASYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=ASYNCEXIT, the connection's complete exit will be given control when the request has completed.</li> <li>• If you specified MODE=ASYNCTOKEN, use REQTOKEN when calling IXLFComp to determine when the request has completed.</li> </ul>
4	xxxx0402	<p><b>Equate Symbol:</b> IXLRNCodeASYNCH</p> <p><b>Meaning:</b> The request could not be processed synchronously. It will be processed asynchronously. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed.</p> <ul style="list-style-type: none"> <li>• If MODE=SYNCECB was specified, the ECB specified for REQECB will be posted when the request has finished.</li> <li>• If MODE=SYNCEXIT was specified, the connection's complete exit will be given control when the request has finished.</li> <li>• If MODE=SYNCTOKEN was specified, an asynchronous request token was returned in the area specified by REQTOKEN. This request token may be specified on the IXLFComp macro to determine when the request has finished.</li> </ul> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=SYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=SYNCEXIT, the connection's complete exit will be given control when the request has completed.</li> <li>• If you specified MODE=SYNCTOKEN, use REQTOKEN when calling IXLFComp to determine when the request has completed.</li> </ul>

Table 78. Return and Reason Codes for IXLLSTC Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	xxxx0409	<p><b>Equate Symbol:</b> IXLRSNCODETIMEOUT</p> <p><b>Meaning:</b> The IXLLSTC request completed prematurely because it exceeded the coupling facility model-dependent time-out criteria. The following information has been returned in the answer area:</p> <ul style="list-style-type: none"> <li>For a MONITOR_SUBLISTS request, the index of the first unprocessed IXLYMSRI entry. All prior IXLYMSRI entries were processed.</li> <li>For a DEQ_EVENTQ request, the number of EMCs that were dequeued from the user's event queue, and the number of EMCs that remain on the user's event queue after dequeuing those EMCs.</li> <li>For a LOCK request, the index of the next lock to be processed.</li> </ul> <p>For WRITE_LCONTROLS and MONITOR_KEYRANGE requests, no specific information is returned in the answer area. However, if a WRITE_LCONTROLS request specifies the NEWAUTH keyword, the list authority is replaced with the new value.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>For WRITE_LCONTROLS, issue a new KEYRANGE=SET request that specifies the same key range as the original request. Any such request that specifies the AUTHCOMP keyword should use the new list authority value.</li> <li>For other request types, reissue the request to continue. Be sure to process the information returned from this request before reissuing the request. The data returned from this request will be overwritten if you specify the same buffer address. Continue to reissue the request until the return code indicates that all processing has completed.</li> </ul> <p>For more information about premature completion of an IXLLSTC request, see <a href="#">z/OS MVS Programming: Sysplex Services Guide</a>.</p>
4	xxxx040E	<p><b>Equate Symbol:</b> IXLRSNCODELOCKNOTHELD</p> <p><b>Meaning:</b> A LOCKOPER=TEST request determined that the specified lock was not held for the specified connection. The connection ID of the lock owner is returned in the answer area (field LAACONID).</p> <p><b>Action:</b> None necessary. If this reason code is not expected, determine who holds the lock and see if any clean-up is necessary. The lock may need to be released, or you can wait and try again.</p>

Table 78. Return and Reason Codes for IXLLSTC Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	xxxx0410	<p><b>Equate Symbol:</b> IXLRSNCODELOCKCOND</p> <p><b>Meaning:</b> For a LOCKMODE=COND request, or a request that specified LOCKCOMP, the request could not be completed successfully because the specified lock is not currently held as required. The connection identifier of the lock owner is returned in the answer area (LAACONID field).</p> <p><b>Action:</b> Retry the request, or obtain the lock as required, and retry the request. If you are unable to get the lock, check the ID in the LAACONID field and determine if some recovery is necessary.</p>
4	xxxx0412	<p><b>Equate Symbol:</b> IXLRSNCODELOCKHELDDBSYS</p> <p><b>Meaning:</b> The lock is not held by any connection, but instead is held by the system. For a request that specified either LOCKOPER=READNEXT or LOCKOPER=TEST, the request could not be completed successfully because the specified lock is not generally available.</p> <p><b>Action:</b> Retry the request, or obtain the lock as required, and retry the request.</p>
4	xxxx041F	<p><b>Equate Symbol:</b> IXLRSNCODENOLOCKSHELD</p> <p><b>Meaning:</b> A request specifying LOCKOPER=READNEXT found no locks held from the LOCKINDEX lock to the end of the lock table.</p> <p><b>Action:</b> None necessary.</p>
8	xxxx0801	<p><b>Equate Symbol:</b> IXLRSNCODEBADPARMLIST</p> <p><b>Meaning:</b> Program error. The parameter list for this request is not addressable. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Verify that:</p> <ul style="list-style-type: none"> <li>• The parameter list address is uncorrupted.</li> <li>• The parameter list is addressable in the caller's primary address space.</li> <li>• If you are issuing this macro while disabled, the parameter list resides in either page-fixed or DREF storage.</li> <li>• If you are issuing this macro in AR-mode and you specified the parameter list address using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If you are issuing this macro in AR-mode, SYSSTATE ASCENV=AR must be issued before issuing this macro.</li> </ul>

Table 78. Return and Reason Codes for IXLLSTC Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0804	<p><b>Equate Symbol:</b> IXLRSNCODEBADVERSIONNUM (or IXLRSNCODEBADVERSION#)</p> <p><b>Meaning:</b> The version number in the macro parameter list is not compatible with the level of XES currently being used. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that your program did not overlay the parameter list storage.</li> <li>• Verify that your program was assembled with the correct macro library for the release of MVS on which your program is running.</li> </ul>
8	xxxx080A	<p><b>Equate Symbol:</b> IXLRSNCODEBADCONTOKEN</p> <p><b>Meaning:</b> Program error. The specified connect token (CONTOKEN) was not valid for one of the following reasons:</p> <ol style="list-style-type: none"> <li>1. The user with the connection identifier represented by the token has disconnected from the structure.</li> <li>2. The connector's task (the task that issued IXLCONN) ended.</li> <li>3. The specified token is not the token that was returned from IXLCONN.</li> <li>4. The request was issued from an address space other than the address space in which IXLCONN was issued.</li> <li>5. The connect token was invalidated during rebuild.</li> <li>6. The connect token was invalidated by XES.</li> </ol> <p><b>Note:</b> The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Take the action with the corresponding meaning.</p> <ol style="list-style-type: none"> <li>1. Discontinue use of the structure. Perform recovery and cleanup for the structure and check the protocol for the use of the structure to determine why you did not know of the disconnection.</li> <li>2. Discontinue use of the structure. Perform recovery and cleanup for the structure and check the protocol for the use of the structure to determine why you did not know that the task ended.</li> <li>3. The contoken is returned in the CONACONTOKEN field of the answer area specified on the IXLCONN request.</li> <li>4. Issue your request from the same address space the IXLCONN was issued in.</li> <li>5. Wait for the rebuild to complete, and try again.</li> <li>6. Discontinue use of the structure. Perform recovery and cleanup for the structure.</li> </ol>

Table 78. Return and Reason Codes for IXLLSTC Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0824	<p><b>Equate Symbol:</b> IXLRSNCODEWRONGSTRTYPE</p> <p><b>Meaning:</b> Program error. The connection specified by CONTOKEN is not to a list structure.</p> <p><b>Action:</b> Specify the appropriate CONTOKEN value, and rerun your program. The CONTOKEN is returned in the answer area provided on the IXLCONN invocation. The type of structure built is passed on the TYPE parameter of the IXLCONN macro.</p>
8	xxxx0825	<p><b>Equate Symbol:</b> IXLRSNCODENOENTRY</p> <p><b>Meaning:</b> Program error. The designated event monitor controls object (EMC) does not exist for the user of the designated sublist.</p> <p><b>Action:</b> None necessary. However, if this return code and reason code are unexpected, you should examine the parameters specified for the list number and entry key on the invocation of this macro.</p>
8	xxxx082B	<p><b>Equate Symbol:</b> IXLRSNCODEBADIDINDEX</p> <p><b>Meaning:</b> Program error. The value specified for either STARTINDEX or ENDINDEX was not valid. No entries were processed. The index of the first entry that was not processed is returned in the answer area.</p> <p><b>Action:</b> Ensure that the STARTINDEX and ENDINDEX values are valid and resubmit the request.</p>
8	xxxx0833	<p><b>Equate Symbol:</b> IXLRSNCODEBADPGBLATTR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list is specified as being pageable (PAGEABLE=YES), but is not.</p> <p><b>Action:</b> Change the buffer area(s) to pageable storage, or specify PAGEABLE=NO. See the PAGEABLE parameter description for specification instructions.</p>

Table 78. Return and Reason Codes for IXLLSTC Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0834	<p><b>Equate Symbol:</b> IXLRSNCODEBADNONPGBLATTR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list, is specified as being nonpageable (PAGEABLE=NO), but is either pageable or not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The type of storage in the buffer area corresponds to the value you specified on the PAGEABLE parameter. See the PAGEABLE parameter description for specification instructions.</li> <li>• The correct buffer address was used.</li> <li>• The buffer area was not previously freed.</li> <li>• If you are calling IXLLIST while disabled, the buffers must reside in either page-fixed or DREF storage.</li> <li>• The buffer areas were allocated in a storage key that matches the key specified by the BUFSTGKEY parameter, or, if BUFSTGKEY is omitted, the caller's PSW key.</li> <li>• If BUFLIST was specified and your program is running in AR-mode: <ul style="list-style-type: none"> <li>– If the BUFFER or BUFLIST parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>– The BUFALET specification is correct.</li> <li>– You specified SYSSTATE ASCENV=AR before issuing the IXLLIST macro.</li> </ul> </li> </ul>
8	xxxx0835	<p><b>Equate Symbol:</b> IXLRSNCODEBADDATAADDR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list, is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The correct buffer address was used.</li> <li>• The buffer area was not previously freed.</li> <li>• The buffer area was allocated in a storage key that matches the key specified using the BUFSTGKEY parameter, or, if BUFSTGKEY is omitted, the caller's PSW key.</li> <li>• If BUFLIST was specified and your program is running in AR mode: <ul style="list-style-type: none"> <li>– The BUFALET specification is correct.</li> <li>– You specified SYSSTATE ASCENV=AR before issuing the macro.</li> </ul> </li> </ul>



Table 78. Return and Reason Codes for IXLLSTC Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0836	<p><b>Equate Symbol:</b> IXLRNCODEBADREALADDR</p> <p><b>Meaning:</b> Program error. Real storage addresses were provided in the BUFLIST list, but one of the buffers is not addressable in central storage.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that BUFADDRTYPE was specified as you intended.</li> <li>• Ensure that the buffer addresses specified by BUFLIST are valid.</li> </ul>
8	xxxx0838	<p><b>Equate Symbol:</b> IXLRNCODEBADANSAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by ANSAREA is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The answer area address specified by ANSAREA is valid.</li> <li>• If the caller is running in AR-mode and the ANSAREA parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• The address specified for ANSAREA is addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are calling IXLLSTC while disabled, ANSAREA must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLLSTC in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLSTC macro.</li> </ul>
8	xxxx0839	<p><b>Equate Symbol:</b> IXLRNCODEBADREQTOKENAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by REQTOKEN is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The request token area specified by REQTOKEN is valid.</li> <li>• If the caller is running in AR-mode and the REQTOKEN parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If you are calling IXLLSTC while disabled, REQTOKEN must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLLSTC in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLSTC macro.</li> </ul>

Table 78. Return and Reason Codes for IXLLSTC Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx083D	<p><b>Equate Symbol:</b> IXLRSNCODEBADANSLEN</p> <p><b>Meaning:</b> Program error. The length of the answer area, as specified by ANSLEN, is not sufficient to contain answer area information. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Increase the size of the answer area, and rerun your program. The minimum size of the ANSAREA is in the LAA_LEN field of the IXLYLAA mapping macro.</p>
8	xxxx0842	<p><b>Equate Symbol:</b> IXLRSNCODEPERSISTENTLOCK</p> <p><b>Meaning:</b> Program error. The request specifying LOCKOPER=SET with LOCKMODE=UNCOND, or LOCKOPER=NOTHELD with LOCKMODE=UNCOND failed because the lock is held by a connection that is in the failed-persistent state. The connection identifier of the lock owner is returned in the answer area (field LAACONID).</p> <p><b>Action:</b> Either perform recovery for the connection, or wait until recovery for the connection is performed.</p>
8	xxxx0846	<p><b>Equate Symbol:</b> IXLRSNCODEBADLOCKINDEX</p> <p><b>Meaning:</b> Program error. The specified LOCKINDEX exceeds the size of the lock table for the structure.</p> <p><b>Action:</b> Correct LOCKINDEX to specify an index that is contained within the lock table. The maximum value for the LOCKINDEX is one less than the value specified by the LOCKENTRIES parameter on the IXLCONN macro.</p>
8	xxxx0847	<p><b>Equate Symbol:</b> IXLRSNCODEBADLISTNUMBER</p> <p><b>Meaning:</b> Program error. The specified LISTNUM value exceeds the number of lists for the structure.</p> <p><b>Action:</b> Correct LISTNUM to specify a list number that exists in the structure. The number of lists allocated for a structure is determined on the LISTHEADERS parameter of the IXLCONN macro. The list numbers go from 0 to n-1, where n is the number of lists specified.</p>
8	xxxx0848	<p><b>Equate Symbol:</b> IXLRSNCODEBADRESET</p> <p><b>Meaning:</b> Program error. LOCKOPER=RESET was specified for a lock not currently held by the invoker. The value of the connection ID holding the lock is returned in the answer area (field LAACONID).</p> <p><b>Action:</b> Check your code to ensure that your connection did not already reset the lock.</p>

Table 78. Return and Reason Codes for IXLLSTC Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx084A	<p><b>Equate Symbol:</b> IXLRSNCODENOKEYS</p> <p><b>Meaning:</b> Program error. The structure does not support the use of entry keys. The IXLLSTC request type either required the structure to support entry keys or was a monitor request type that required a keyed structure.</p> <p><b>Action:</b> Ensure that you are connected to the intended structure. The use of keys for a structure is determined by the REFOPTION keyword on the IXLCONN macro.</p>
8	xxxx084B	<p><b>Equate Symbol:</b> IXLRSNCODENOLOCKS</p> <p><b>Meaning:</b> Program error. A locking operation was requested, but the structure does not contain a lock table.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• You are connected to the intended structure.</li> <li>• You intended to perform a locking operation.</li> </ul> <p>The use of locks is determined by the LOCKENTRIES keyword on the IXLCONN macro.</p>
8	xxxx0851	<p><b>Equate Symbol:</b> IXLRSNCODENOSUSPENDISABLE</p> <p><b>Meaning:</b> Program error. The request failed because MODE=SYNCSUSPEND was specified, but the caller is disabled and cannot be suspended.</p> <p><b>Action:</b> Either specify another MODE value, or become enabled (release the CPU lock), then reissue the request.</p>
8	xxxx0852	<p><b>Equate Symbol:</b> IXLRSNCODENOLISTVECTOR</p> <p><b>Meaning:</b> Program error. The request failed because no local vector for monitoring list headers, key ranges, or event queues exists for this connection. Either a list notification vector was not requested for this connection, or the list notification vector has been deleted.</p> <p><b>Action:</b> Either you are not connected to this structure or the VECTORLEN parameter was not specified when the IXLCONN was done for this structure.</p>
8	xxxx0853	<p><b>Equate Symbol:</b> IXLRSNCODEINVLISTVINDEX</p> <p><b>Meaning:</b> Program error. The list notification vector index (VECTORINDEX) specified with ACTION=START was not valid. This might be because the vector index you specified is greater than the number of vector entries in the list notification vector.</p> <p><b>Action:</b> Correct the VECTORINDEX that you are using.</p>

Table 78. Return and Reason Codes for IXLLSTC Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0854	<p><b>Equate Symbol:</b> IXLRSNCODEBADLOCKCOMP</p> <p><b>Meaning:</b> Program error. The connection identifier specified for LOCKCOMP is not valid.</p> <p><b>Action:</b> The connection identifier is returned in the answer area (mapped by IXLYCONA) when the IXLCONN macro was issued.</p>
8	xxxx0859	<p><b>Equate Symbol:</b> IXLRSNCODEBADLISTAUTH</p> <p><b>Meaning:</b> The list authority value for the specified list does not meet the criteria specified by AUTHCOMP. The current list authority (field LAALISTAUTH) and description (field LAALISTDESC) are returned in the answer area.</p> <p><b>Action:</b> None required; however you might want to take some action depending on your application. The list authority is set to binary zeros when the structure is allocated. When IXLLSTC is issued with the NEWAUTH keyword, the list authority is changed if the correct comparison list authority value is specified. Check the answer area to determine the list authority value for the list specified. Verify that the correct list number was specified.</p>
8	xxxx0864	<p><b>Equate Symbol:</b> IXLRSNCODEBADBUFSIZE</p> <p><b>Meaning:</b> Program error. The size of the BUFFER area or the buffer areas specified by BUFLIST is not large enough to contain the data being read. No data is returned.</p> <p><b>Action:</b> If more space is available, specify a larger buffer size, and reissue the request. For READ_LCONTROLS and DEQ_EVENTQ requests, the specified buffer area must be 4096 bytes in length.</p>
8	xxxx0865	<p><b>Equate Symbol:</b> IXLRSNCODEBADBUFSPEC</p> <p><b>Meaning:</b> Program error. There is an error in the buffer specification.</p> <p><b>Action:</b> Check the following:</p> <ul style="list-style-type: none"> <li>• If BUFLIST was specified, check the requirements for BUFLIST, BUFNUM, and BUFINCRNUM.</li> <li>• If BUFFER was specified, check the requirements for BUFFER and BUFSIZE.</li> <li>• Buffer pointer(s) in BUFLIST</li> <li>• Buffer boundaries.</li> </ul>

Table 78. Return and Reason Codes for IXLLSTC Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0866	<p><b>Equate Symbol:</b> IXLRSNCODEBADBUFKEY</p> <p><b>Meaning:</b> Program error. The buffer storage key specified by BUFSTGKEY is incorrect. For requests that write coupling facility data, the data cannot be fetched from the specified buffer area. For requests that read coupling facility data, the data cannot be stored into the specified buffer area.</p> <p><b>Action:</b> Check the following:</p> <ul style="list-style-type: none"> <li>• Determine if the key of the storage being used for the buffers is different from the PSW key.</li> <li>• If BUFSTGKEY was specified, verify that the key was put in the correct bits (see the explanation of this parameter for more information).</li> <li>• If you are calling IXLLSTC in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLSTC macro.</li> </ul>
8	xxxx0867	<p><b>Equate Symbol:</b> IXLRSNCODEBADBUFLIST</p> <p><b>Meaning:</b> Program error. The 128-byte storage area specified by BUFLIST is not addressable.</p> <p><b>Action:</b> Ensure that the address specified by BUFLIST is valid.</p>
8	xxxx0880	<p><b>Equate Symbol:</b> IXLRSNCODEBADMOSVECTOR</p> <p><b>Meaning:</b> Program error. The storage area specified by MOSVECTOR is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The correct storage area address was used.</li> <li>• If you are running in AR-mode and the MOSVECTOR was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If you are calling IXLLSTC in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLSTC macro.</li> </ul>
8	xxxx0891	<p><b>Equate Symbol:</b> IXLRSNCODEBADKEYRANGEEND</p> <p><b>Meaning:</b> Program error. The specified KEYRANGEEND value is not valid. The value of KEYRANGEEND must be greater than or equal to the value of KEYRANGESTART.</p> <p><b>Action:</b> Ensure that the value specified by KEYRANGEEND is valid.</p>
8	xxxx0892	<p><b>Equate Symbol:</b> IXLRSNCODEBADKRNOTEMPTY</p> <p><b>Meaning:</b> Program error. The specified KRNOTEMPTY value is not valid. The value of KRNOTEMPTY must be greater than or equal to the value of KREMPY.</p> <p><b>Action:</b> Ensure that the value specified by KRNOTEMPTY is valid.</p>

Table 78. Return and Reason Codes for IXLLSTC Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0893	<p><b>Equate Symbol:</b> IXLRSNCODEBADLISTNOTEMPTY</p> <p><b>Meaning:</b> Program error. The specified LISTNOTEMPTY value is not valid. The value of LISTNOTEMPTY must be greater than or equal to the value of LISTEMPTY.</p> <p><b>Action:</b> Ensure that the value specified for LISTNOTEMPTY is greater than or equal to the value of LISTEMPTY.</p>
8	xxxx0897	<p><b>Equate Symbol:</b> IXLRSNCODEBADKEYTYPE</p> <p><b>Meaning:</b> Program error. The specified KEYTYPE value is not suitable for the structure. KEYTYPE=SECONDARY is valid only if the structure supports secondary keys. KEYTYPE=ENTRY is valid only if the structure supports entry keys.</p> <p><b>Action:</b> Correct the value of KEYTYPE.</p>
8	xxxx08AD	<p><b>Equate Symbol:</b> IXLRSNCODEBADHIGHSHAREDVRT</p> <p><b>Meaning:</b> Program error. The request specified a high shared virtual storage area (above 2GB).</p> <p><b>Action:</b> None required.</p>
C	xxxx0C06	<p><b>Equate Symbol:</b> IXLRSNCODENOCNN</p> <p><b>Meaning:</b> Environmental error. This system does not have connectivity to the coupling facility that contains the list structure. Possible reasons for this are:</p> <ul style="list-style-type: none"> <li>• The operator issued VARY PATH,OFFLINE.</li> <li>• The operator issued CONFIG CHP,OFFLINE.</li> <li>• Hardware errors to the coupling facility.</li> <li>• Facility or path failure to the coupling facility.</li> </ul> <p><b>Action:</b> Begin rebuilding the structure on a different coupling facility, or disconnect from the structure.</p>
C	xxxx0C13	<p><b>Equate Symbol:</b> IXLRSNCODEREQPURGED</p> <p><b>Meaning:</b> Environmental error. The request was purged prior to completion of the request. Possible reasons include:</p> <ul style="list-style-type: none"> <li>• The connector failed.</li> <li>• The connector disconnected.</li> <li>• The requestor failed.</li> <li>• The request was purged by IXLPURGE.</li> <li>• Requests were purged when the connector provided an IXLEERSP response for the Rebuild Stop or Rebuild Cleanup event.</li> <li>• The secondary address space was no longer valid.</li> </ul> <p><b>Action:</b> None if this is expected. Otherwise, determine why the connector failed.</p>

Table 78. Return and Reason Codes for IXLLSTC Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C14	<p><b>Equate Symbol:</b> IXLRNCODESTATUSUNKNOWN</p> <p><b>Meaning:</b> Environmental error. The request has completed, but the final disposition of the request cannot be determined. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the validity of your data by comparing the expected results with what is in the coupling facility.</li> <li>• Consider running rebuild processing to move the structure to attempt to protect it from further errors of this kind.</li> </ul>
C	xxxx0C17	<p><b>Equate Symbol:</b> IXLRNCODESTRFULL</p> <p><b>Meaning:</b> Environmental error. The MONITOR_SUBLIST or MONITOR_SUBLISTS request attempted to create a new event monitor controls (EMC) object, but the structure is full and cannot accommodate any more EMCs.</p> <p>For a MONITOR_SUBLISTS request, the index of the IXLYMSRI entry that experienced the structure full condition is returned in the answer area.</p> <p><b>Action:</b> Determine why the structure is full.</p> <ul style="list-style-type: none"> <li>• For a MONITOR_SUBLIST request, you should be monitoring the use of EMC objects (LAAMNSL_EMCCNT is the count of EMCs in use when sublist monitoring was established and LAAMNSL_MAXEMCCNT is the maximum number of EMCs for the structure.) Evaluate this data periodically to ensure structure resources are being used efficiently. You might be able to delete existing EMCs to free up space, or, if rebuild is allowed (IXLCONN macro, ALLOWREBLD parameter), rebuild the structure either to make it larger or with a changed EMCSTGPCT to allow more EMCs to be created.</li> <li>• For a MONITOR_SUBLISTS request, you should be monitoring the usage of the structure every time a REQUEST=MONITOR_SUBLISTS is done (LAAMNSLS_EMCCNT is the total count of EMCs in use in the list structure and LAAMNSLS_MAXEMCCNT is the maximum number of EMCs for the list structure.). This data should be evaluated periodically to ensure structure resources are being used efficiently. Field LAAMNSLS_FAILINDEX contains the index of the entry in IXLYMSRI that experienced the structure full condition. IXLYMSRI entries prior to LAAMNSLS_FAILINDEX were processed successfully.</li> </ul>
C	xxxx0C25	<p><b>Equate Symbol:</b> IXLRNCODESTRFAILURE</p> <p><b>Meaning:</b> Environmental error. The list structure failed prior to completion of the request.</p> <p><b>Action:</b> Either rebuild or disconnect from the structure.</p>

Table 78. Return and Reason Codes for IXLLSTC Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C68	<p><b>Equate Symbol:</b> IXLRSNCODEBADREQCFLEVEL</p> <p><b>Meaning:</b> Environmental error. The request type is not permitted for the level of coupling facility in which the target structure is allocated.</p> <p><b>Action:</b> Either disconnect from the structure (using IXLDISC) or initiate a rebuild of the structure, if allowed (using IXLREBLD) in a coupling facility of the correct CFLEVEL.</p>
C	xxxx0CA0	<p><b>Equate Symbol:</b> IXLRSNCODEQUIESCEDSUSPENDFAIL</p> <p><b>Meaning:</b> Environmental error. The request is failed because the structure is quiesced for a system-managed process and SUSPEND=FAIL is specified on the IXLCONN.</p> <p><b>Action:</b> None, if this is expected.</p>
C	xxxxFFFF	<p><b>Equate Symbol:</b> IXLRSNCODENOTAVAILABLE</p> <p><b>Meaning:</b> Environmental error. XES functions are not available. This could occur because the coupling facility hardware necessary to provide XES functions is not present.</p> <p><b>Action:</b> Re-IPL the system, or follow your particular failure management protocol.</p>
10	xxxx10xx	<p><b>Meaning:</b> System error. XES processing failure. The state of the involved structure and the disposition of the request are unpredictable. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Contact the IBM support center.</p>



## Chapter 77. IXLLSTE – XES List Structure Single Entry Services

### Description

The IXLLSTE service enables you to atomically operate on individual list entries in a list structure. Some functions of IXLLSTE require that the structure be allocated in a coupling facility of CFLEVEL=9 or higher. The following services are available for operating on individual list entries:

- Create a new list entry, either by reading it from your storage area or by moving it from an existing list in the structure.
- Read a list entry into your storage area.
- Write a list entry from your storage area to a list structure.
- Move a list entry from one list to another, which may result in the creation of a new list entry.
- Delete a list entry from a list.

The IXLLSTE macro provides list services equivalent to the following IXLLIST request types:

- IXLLIST REQUEST=READ
- IXLLIST REQUEST=WRITE
- IXLLIST REQUEST=MOVE
- IXLLIST REQUEST=DELETE

See the descriptions of the corresponding IXLLIST macro for basic information about the services provided.

List structures allocated in a coupling facility of CFLEVEL=9 or higher can be allocated with certain attributes not available in a lower level coupling facility.

- Users can assign a user-designated entry identifier to a newly created list entry, rather than having the system assign the entry identifier.
- Secondary keys can be specified, thus allowing the user to reference a keyed list entry by entry key, secondary key, or both.

The IXLLSTE macro can be used to exploit these new attributes.

See *z/OS MVS Programming: Sysplex Services Guide* for information about using the IXLLSTE macro.

### Environment

The following are the environment requirements for the caller.

Environment	Environment requirement
Minimum authorization:	Supervisor state and PSW key 0-7
Dispatchable unit mode:	Task or SRB
Cross memory mode:	Any PASN, any HASN, any SASN. The current primary address space must be the same as the primary address space at the time the connection service (IXLCONN) was issued for the structure.
AMODE:	31-bit
ASC mode:	Primary or access register (AR)
Interrupt status:	Enabled or disabled for I/O and external interrupts

Environment	Environment requirement
Locks:	Disabled callers must be legally disabled by holding the CPU lock and cannot hold other disabled locks. Enabled callers must not hold any locks. When MODE=SYNCSUSPEND is specified, the caller must be enabled.
Control parameters:	See “Restrictions” on page 1418

## Programming Requirements

- If your program is in AR mode, issue the SYSSTATE ASCENV=AR macro before IXLLSTE. SYSSTATE ASCENV=AR tells the system to generate code appropriate for AR mode.
- Include the IXLYCON mapping macro in your program. This macro provides a list of equate symbols for users of XES services and exits.
- Include mapping macros IXLYLAA and IXLYLCTL in your program as necessary. Table 79 on page 1418 lists these macros, the information and areas they map, and the particular IXLLSTE requests and parameters they apply to.

Table 79. Mapping Macros for IXLLSTE

Mapping Macro	Information	Area	Request/Parameter
IXLYLAA	Answer area output	ANSAREA area	All requests
IXLYLCTL	List entry controls	Field LAALCTL of IXLYLAA	READ, WRITE, MOVE, DELETE

## Restrictions

- If you specify BUFLIST and PAGEABLE=YES, all of the buffers in the list must be in the same area of storage; you cannot mix common and private storage addresses for the buffers in the list.
- This service cannot be invoked by callers running as a disabled interrupt exit (DIE).
- The caller's parameter list must be addressable in the caller's primary address space.
- If the caller is running in AR ASC mode and specifies a parameter using explicit register notation, the access register corresponding to the general register must appropriately qualify the general register.
- The virtual storage areas specified by the ADJAREA and ANSAREA parameters must be addressable in the caller's primary address space or from the caller's PASN access list.
- The virtual storage areas specified by parameters other than ADJAREA and ANSAREA can be addressable in the caller's primary, secondary, or home address spaces, from the PASN access list, or from the dispatchable unit access list (DU-AL).
- If the caller is disabled then the parameter list and all storage areas addressed by the macro parameters must reside in either nonpageable or disabled reference storage.

## Input Register Information

Before issuing the IXLLSTE macro, the caller does not have to place any information into any register unless using it in register notation for a particular parameter, or using it as a base register.

## Output Register Information

When control returns to the caller, the GPRs contain:

**Register  
Contents**

**0**

Reason code

**1**

Used as work register by the system

**2-13**

Unchanged

**14**

Used as work register by the system

**15**

Return code

When control returns to the caller, the ARs contain:

**Register****Contents****0-1**

Used as work registers by the system

**2-13**

Unchanged

**14-15**

Used as work registers by the system

## Performance Implications

---

See *z/OS MVS Programming: Sysplex Services Guide* for performance information.

## Understanding IXLLSTE Version Support

---

The IXLLSTE macro supports versions 0, 1, and 4 — the version number corresponds to the level of the IXLLIST or IXLLSTE macro in which a keyword or function was introduced. The higher the version number, the more recent the version of the macro. Some IXLLSTE keywords are supported for multiple versions, but have different functions for each version.

- Keywords not specifically noted here are supported by all versions starting with version 0 and higher of the IXLLSTE macro.
- The following IXLLSTE keywords are supported by all versions starting with version 1 and higher of the IXLLSTE macro.
  - LISTKEYINC
- The following keywords and functions are supported by all versions starting with version 4 and higher of the IXLLSTE macro.
  - ASSIGNENTRYID
  - KEYCOMPARE
  - KEYPOSITION
  - LISTLIMIT
  - SECONDARYKEY
  - SKEYCOMPARE
  - SKEYPOSITION
  - SKEYREQTYPE
  - SKEYTARGETDIR

Specify the version of the parameter list that you want generated with the PLISTVER keyword. See Chapter 2, “Specifying a Macro Version Number,” on page 5 for considerations when specifying the version of the parameter list with PLISTVER.

## Summary of Version-Dependent Parameter Functions

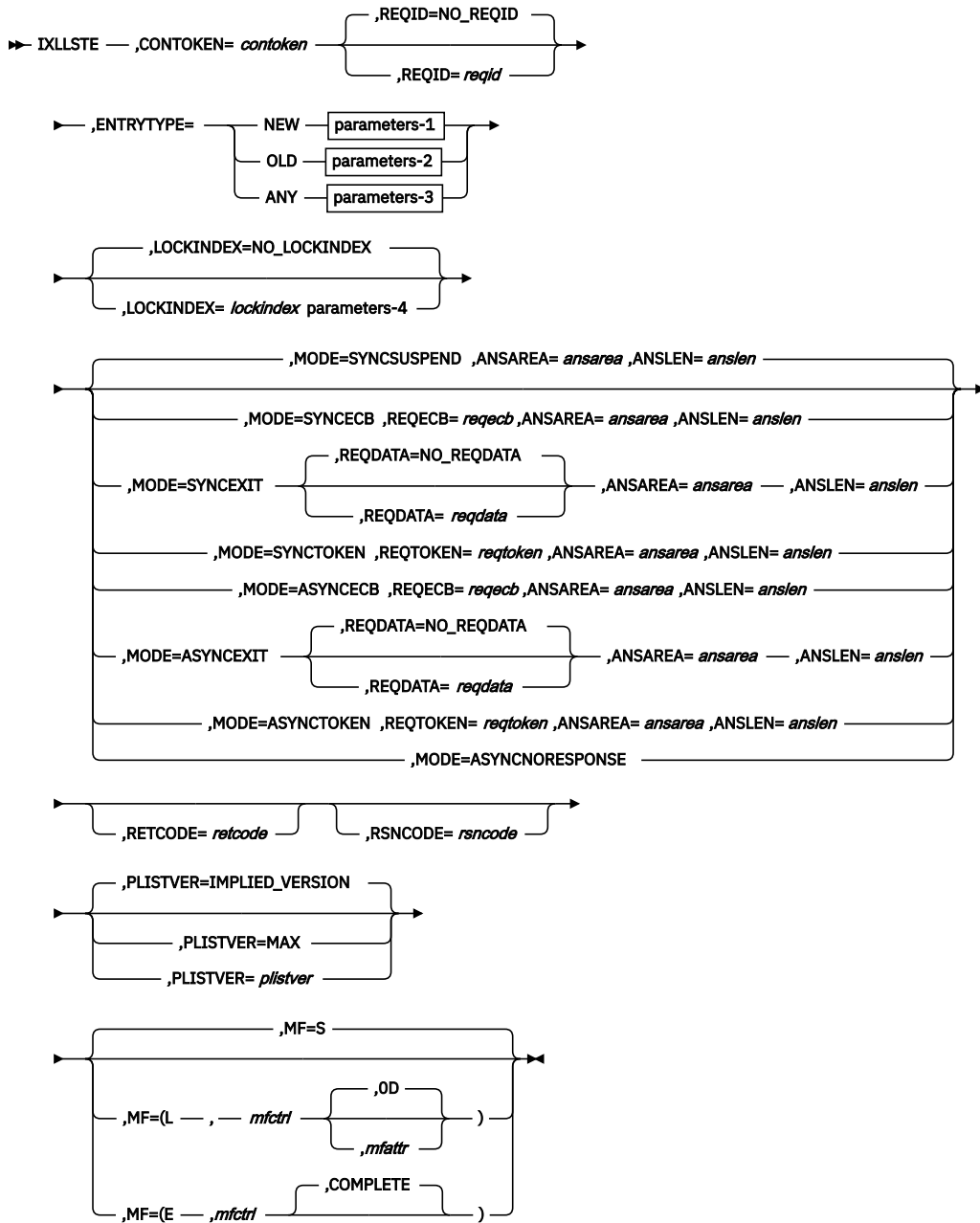
The following table summarizes the allowed use of some of the specifications that can be made depending on the PLISTVER value specified.

<i>Table 80. IXLLSTE Version Support</i>		
<b>Keyword</b>	<b>Version</b>	<b>Notes</b>
ASSIGN	0	NONE, NAME, or KEY may be specified.
	1	NONE, NAME, KEY, or LISTKEY may be specified.
ASSIGNLISTKEY	0	NO may be specified.
	1	NO, CREATE, MOVE, or ANY may be specified.
AUTHCOMPARE	0	NO may be specified.
	1	NO or YES may be specified.
CURSORUPDTYPE	0	NEXT may be specified.
	1	NEXT, NEXTCOND, CURRENT, or CURRENTCOND may be specified.
KEYTYPE	0	ENTRY may be specified.
	1	Same as version 0.
	2	Same as version 1.
	3	Same as version 2.
	4	ENTRY or SECONDARY may be specified.
MOVETOKEY	0	UNCHANGED or TARGETENTRYKEY may be specified.
	1	UNCHANGED, TARGETENTRYKEY, or LISTKEY may be specified.
NEWAUTH	0	NO_NEWAUTH may be specified.
	1	May be specified.
VERSCOMPTYPE	0	EQUAL may be specified.
	1	EQUAL or LESSOREQUAL may be specified.

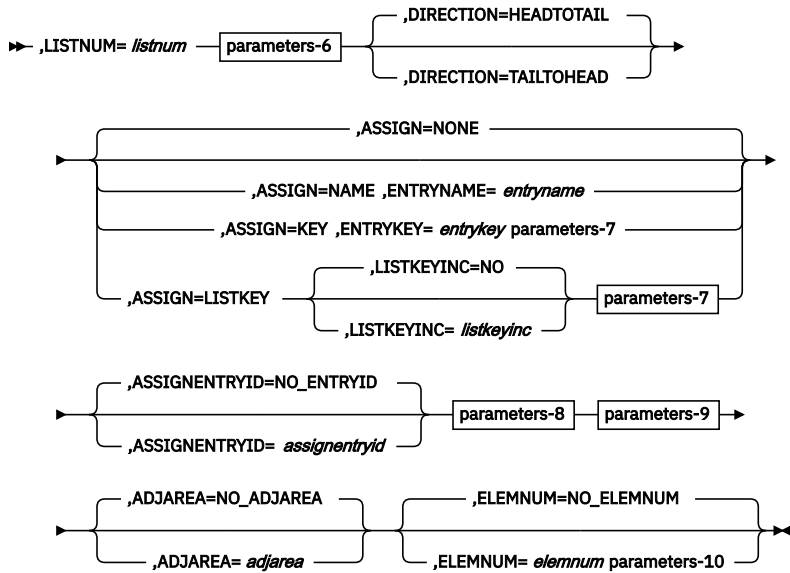
## Syntax Diagram

The syntax diagram for IXLLSTE is as follows:

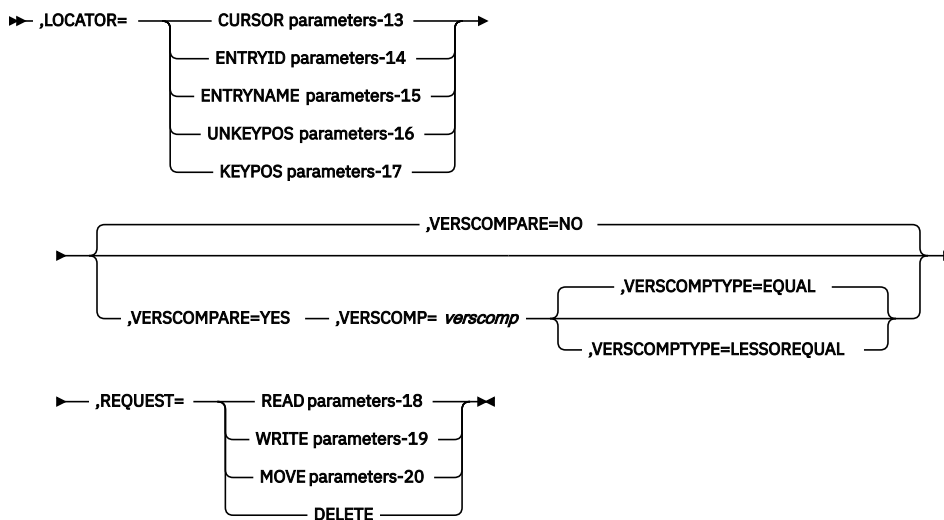
## main diagram



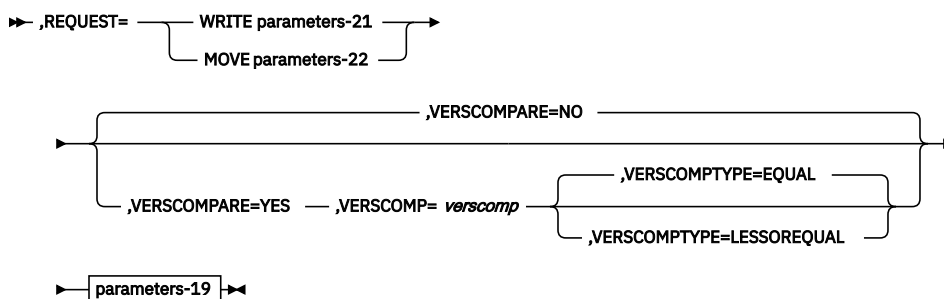
## parameters-1

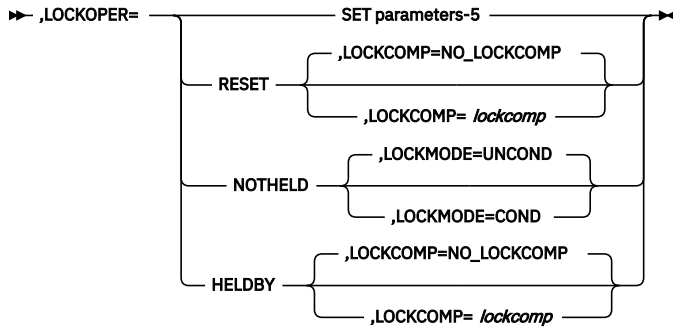
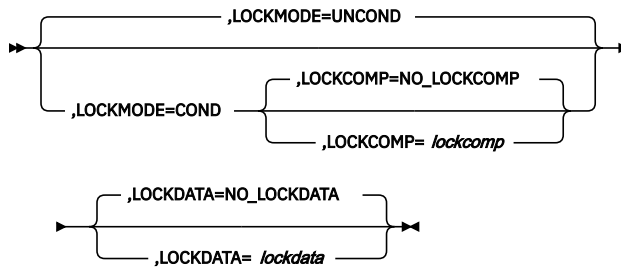
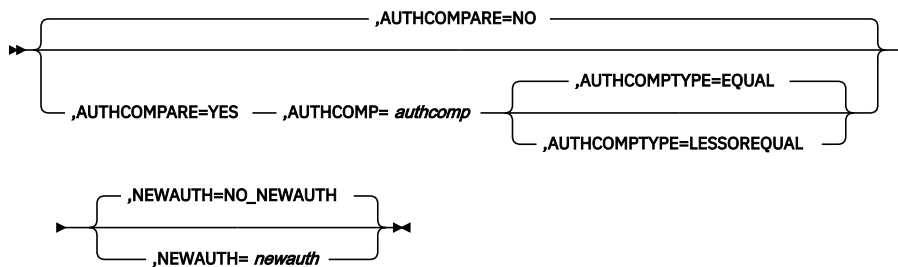
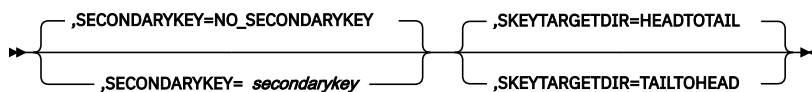
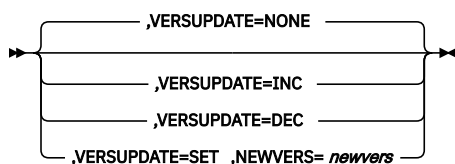


## parameters-2

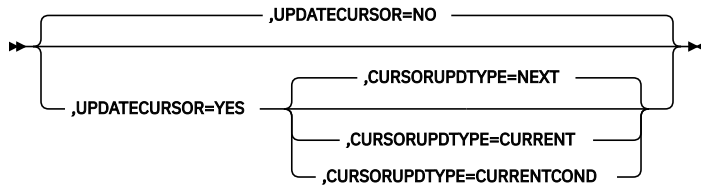


## parameters-3

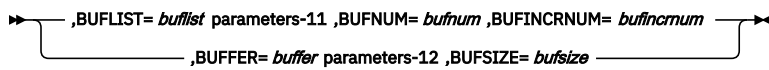


**parameters-4****parameters-5****parameters-6****parameters-7****parameters-8**

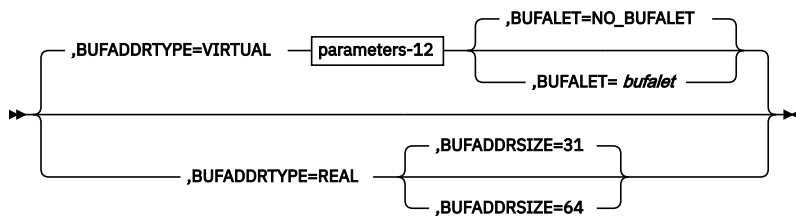
## parameters-9



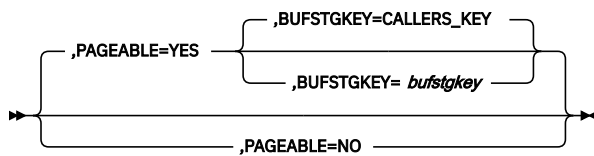
## parameters-10



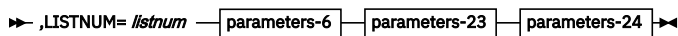
## parameters-11



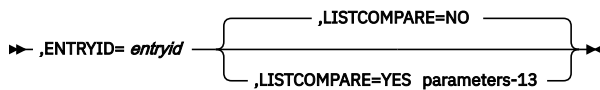
## parameters-12



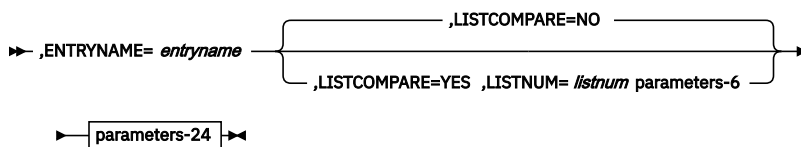
## parameters-13



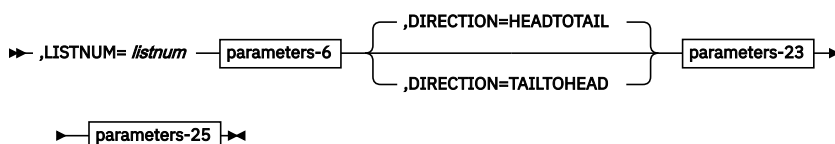
## parameters-14



## parameters-15

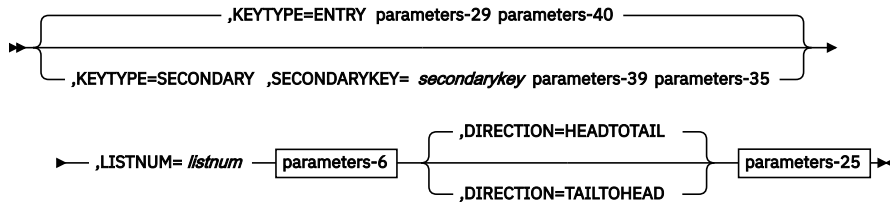


## parameters-16

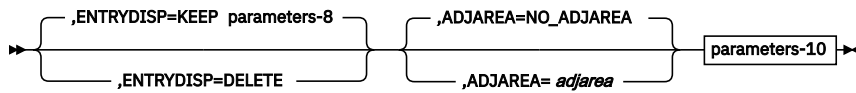




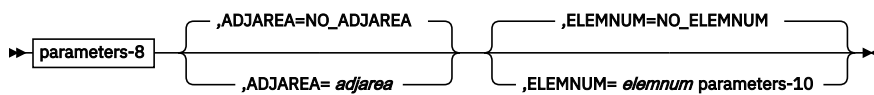
## parameters-17



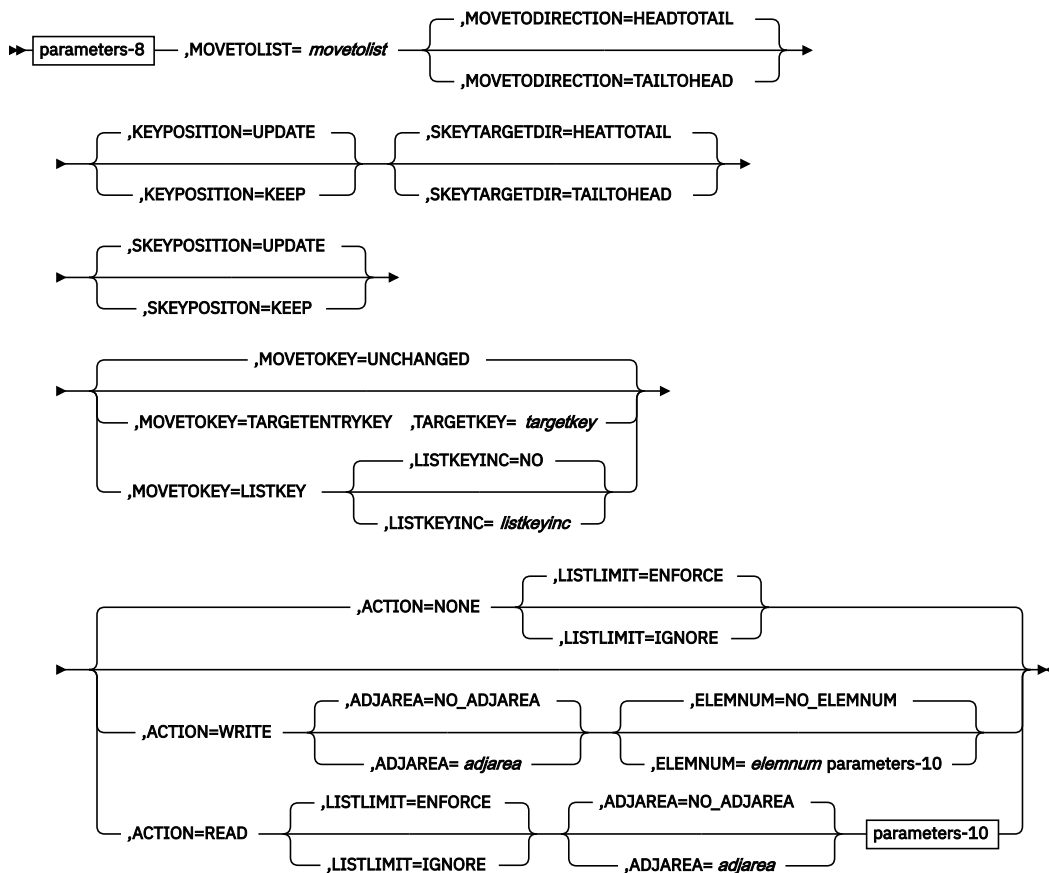
## parameters-18

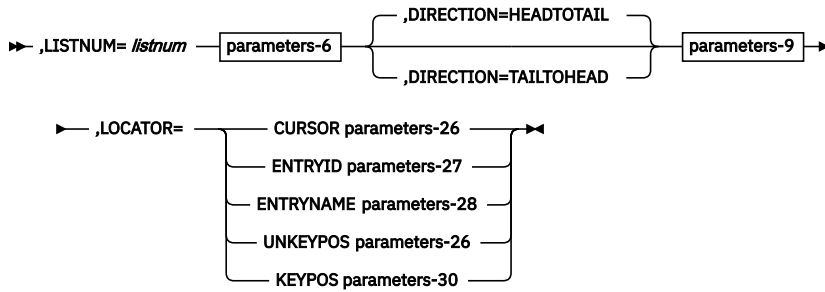
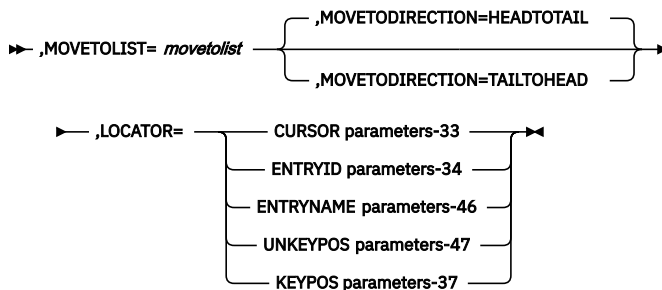
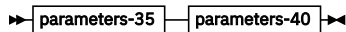
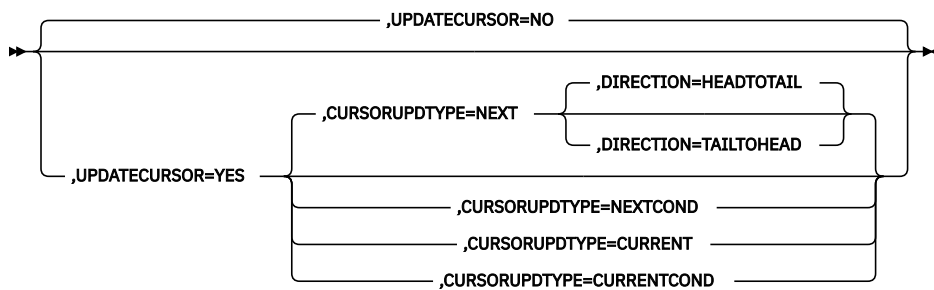
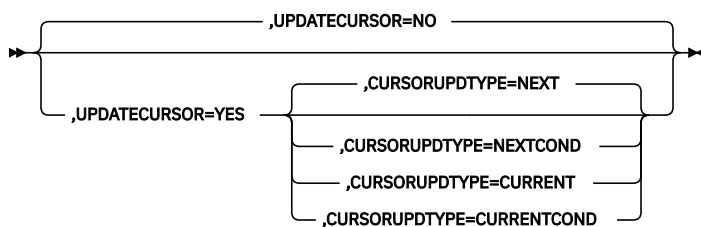


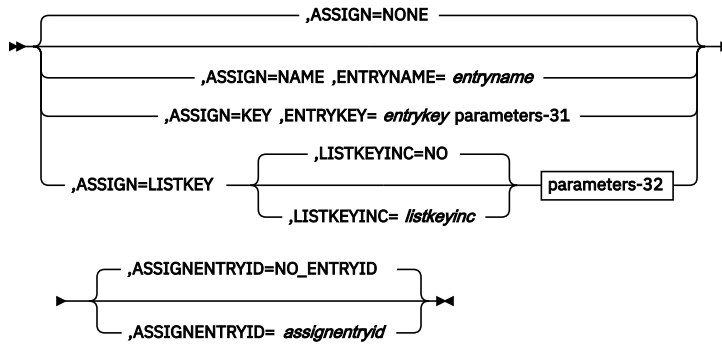
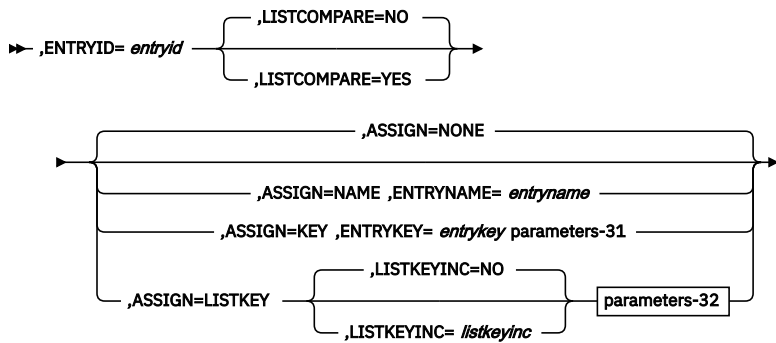
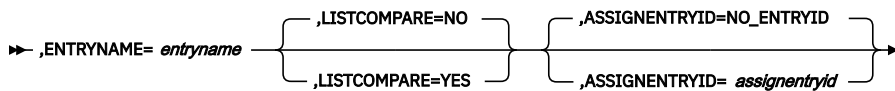
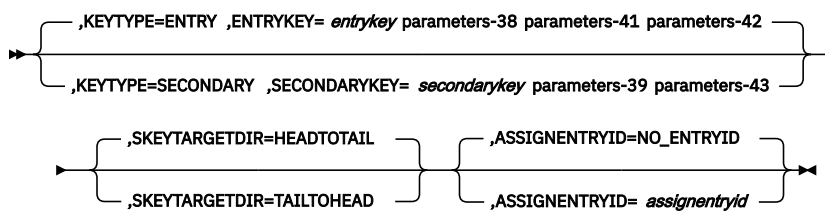
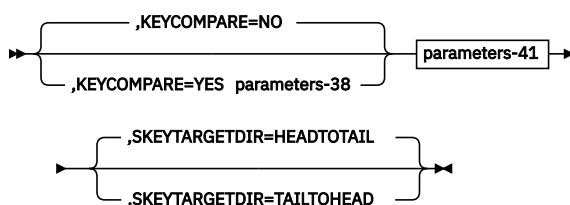
## parameters-19



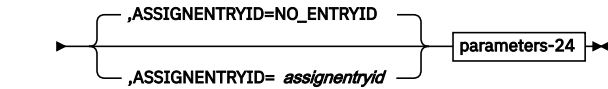
## parameters-20



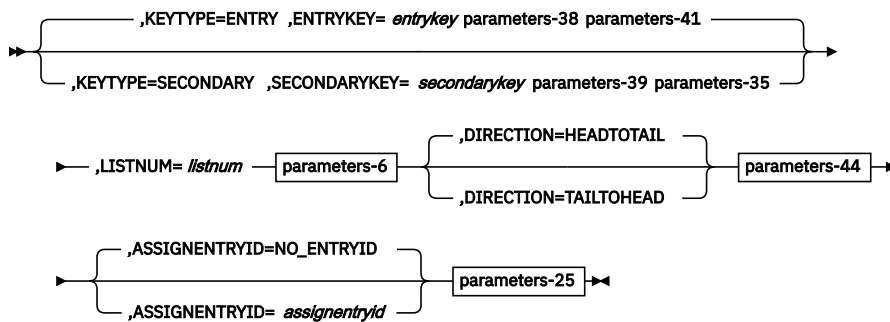
**parameters-21****parameters-22****parameters-23****parameters-24****parameters-25**

**parameters-26****parameters-27****parameters-28****parameters-29****parameters-30****parameters-31**

### parameters-33

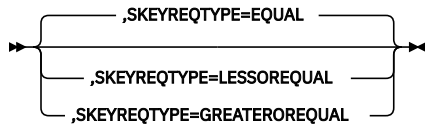
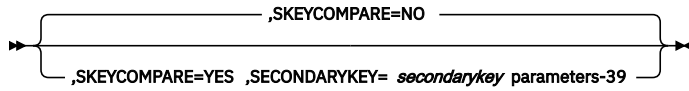
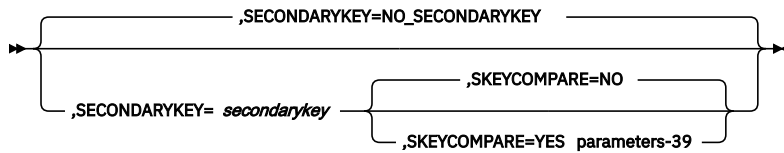
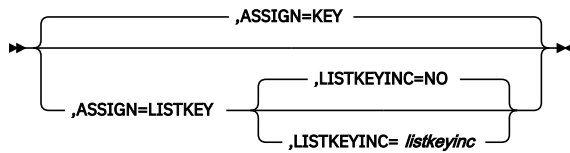
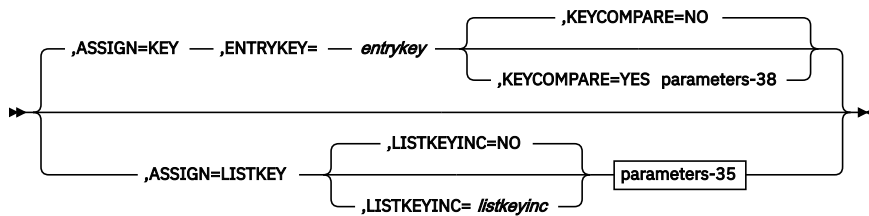
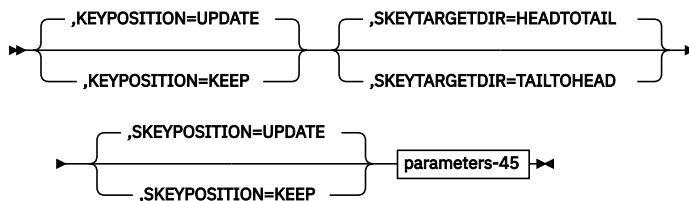


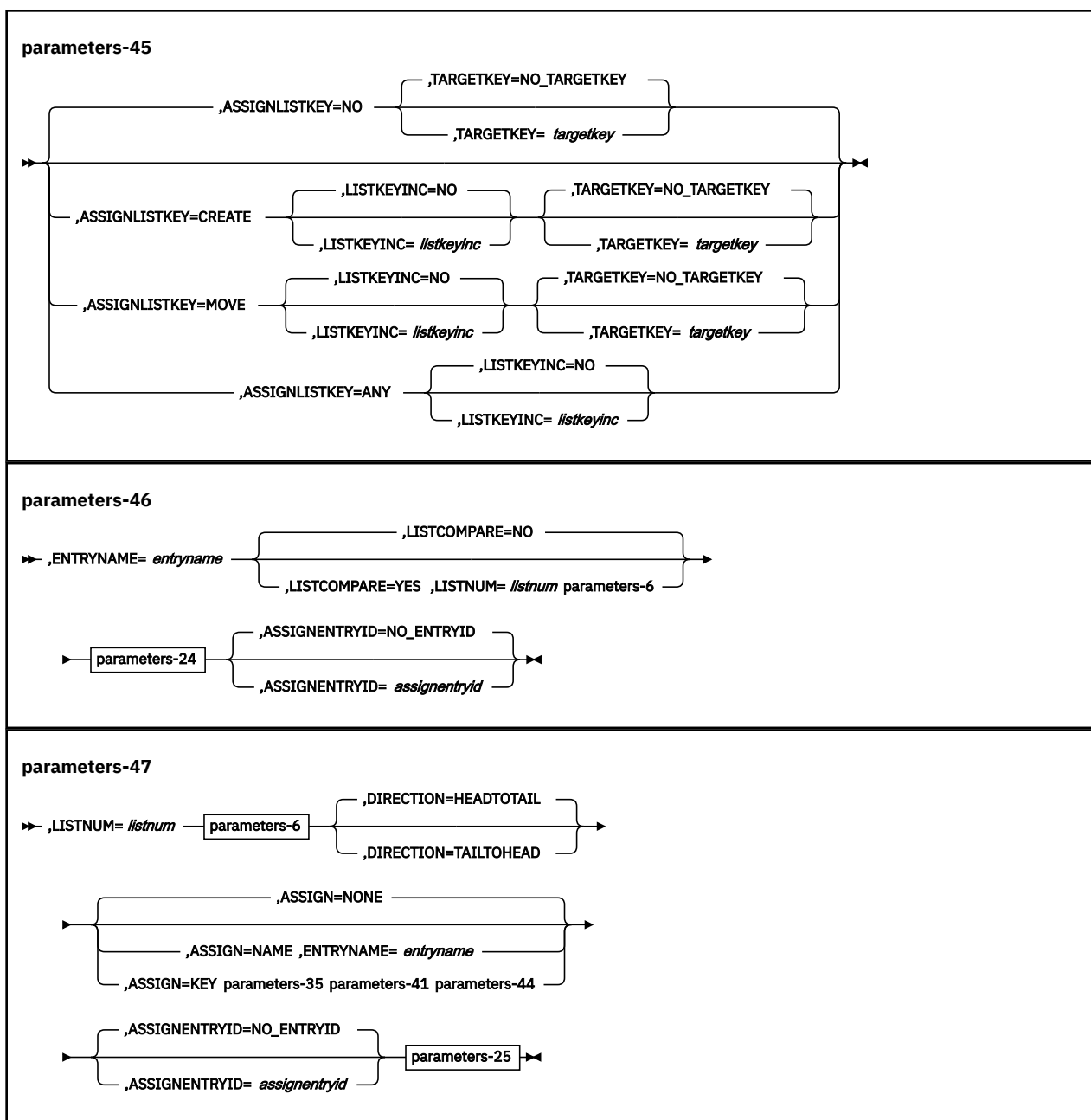
### parameters-35



```

graph LR
    A[KEYREQTYPE] --- B[=EQUAL]
    A --- C[=LESSOREQUAL]
    A --- D[=GREATEROREQUAL]
    B --- E[ ]
    C --- E
    D --- E
    E --- F[ ]
    style E width:0px,height:0px
    style F width:0px,height:0px
  
```

**parameters-39****parameters-40****parameters-41****parameters-42****parameters-43****parameters-44**



## Parameter Descriptions

The parameter descriptions for IXLLSTE are listed in alphabetical order. Default values are underlined:

**,ACTION=NONE**

**,ACTION=WRITE**

**,ACTION=READ**

Use this input parameter to specify an additional action with the move request when ENTRYTYPE=OLD.

**NONE**

No additional action is to be performed.

**WRITE**

In addition to moving the list entry, write the contents of the storage area(s) specified by BUFFER or BUFLIST and/or BUFLIST to the list entry, if the storage areas exist.

**READ**

In addition to moving the list entry, read the entry data and/or adjunct data into the storage areas specified by BUFFER or BUFLIST and/or ADJAREA.

At least one of BUFFER, BUFLIST, or ADJAREA must be specified when ACTION=READ is specified with REQUEST=MOVE.

**,ADJAREA=NO\_ADJAREA****,ADJAREA=adjarea**

Use this input or output parameter to specify a storage area to contain the adjunct data that is read from or written to an entry.

Specify ADJAREA only for structures that support adjunct data. (Adjunct areas for a structure are established through the IXLCONN macro.)

If the structure was allocated to use secondary keys, and the IXLLSTE request results in a new list entry being created, the first 32 bytes of the list entry adjunct data will be used to store the secondary key of the list entry. This allows only the last 32 bytes of list entry adjunct data to be written.

If the structure was allocated to use secondary keys and the IXLLSTE request results in the list entry being moved or updated, the first 32 bytes of the ADJAREA are ignored, allowing only the last 32 bytes of data to be written to the list entry adjunct area. The secondary key of the list entry can only be updated by an IXLLSTM REQUEST=MOVE\_ENTRYLIST request.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 64-byte area that contains or will contain the adjunct data.

**,ANSAREA=ansarea**

Use this input parameter to specify an answer area to contain information returned from the request. The format of the answer area is described by mapping macro IXLYLAA.

Not all fields in the answer area are applicable to all request types. Request type descriptions indicate which answer area fields are applicable for successful request completion cases. Return and reason code descriptions indicate which answer area fields are applicable for non-successful completing requests.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an area (with a length of ANSLEN) that will contain the information returned by the request.

**,ANSLEN=anslen**

Use this input parameter to specify the size of the storage area specified by ANSAREA.

Check the prologue of the IXLYLAA mapping macro for the minimum required size of the answer area, or use the length field (LAA\_LEN) in the LAA.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 2-byte field that contains the size, in bytes, of the answer area.

**,ASSIGN=NONE****,ASSIGN=NAME****,ASSIGN=KEY****,ASSIGN=LISTKEY**

Use this input parameter to specify whether to assign an ENTRYNAME, ENTRYKEY, or list key value to a newly created list entry when ENTRYTYPE=NEW or ANY. If the structure was allocated to use secondary keys, a secondary key value may be assigned to the newly created list entry.

If the list entry is moved or created, use this parameter to assign a TARGETKEY, ENTRYKEY, or list key value.

ASSIGN also allows key comparison to be specified for keyed list entries.

**NONE**

No assignment is to be explicitly made.

If the list entry is created, the following applies:

- If the structure was allocated to use keyed entries, the entry key (and therefore, the target list keyed position) is assigned as follows:
  - If DIRECTION=HEADTOTAIL is specified (explicitly or by default), the list entry is assigned an entry key value of all binary zeros.
  - If DIRECTION=TAILTOHEAD is specified, the list entry is assigned an entry key value of all binary ones.
- If the structure was allocated to use secondary keys, the list entry is assigned a secondary key value of all binary zeros.

If the list entry is moved, the entry key and the secondary key remain unchanged.

### NAME

The entry name specified by ENTRYNAME is to be assigned to the list entry if it is created as a result of the IXLLSTE request.

If a list entry already exists with the specified ENTRYNAME, a new list entry is not created and the IXLLSTE request is ended.

ASSIGN=ENTRYNAME may only be specified when the structure was allocated to use named entries.

### KEY

The entry key (and therefore the target list keyed position) specified for ENTRYKEY is to be assigned to the newly created or moved list entry.

If the list entry is created and the structure was allocated to use secondary keys, the secondary key value specified by SECONDARYKEY will be assigned to the list entry. The secondary key will be stored in the first 32 bytes of the list entry adjunct area. If the user supplies ADJAREA, the first 32 bytes will be ignored. If the user does not supply SECONDARYKEY, the secondary key will be set to all binary zeros.

For keyed list entries, when ENTRYTYPE=NEW is specified, a new list entry is created regardless of whether a list entry already exists with the same ENTRYKEY value or the same SECONDARYKEY value.

### LISTKEY

The current list key value is to be assigned to the newly created list entry.

The list key and maximum list key values may be set with an IXLLSTC REQUEST=WRITE\_LCONTROLS request.

The current list key value can be assigned to a list entry only on structures allocated in a coupling facility of CFLEVEL=1 or higher.

### **,ASSIGNENTRYID=NO\_ENTRYID**

### **,ASSIGNENTRYID=assignentryid**

Use this input parameter to specify the ENTRYID to be assigned to the list entry created as a result of this request when ENTRYTYPE=NEW or ANY. A user provided ENTRYID must be specified if and only if ENTRYIDTYPE=USER was specified on the IXLCONN request.

If the request is attempting to create a list entry and either a list entry already exists with the specified ENTRYID, or ASSIGNENTRYID was not specified, a new list entry is not created and the IXLLSTE request is ended.

If the request is not creating a new list entry, the value specified for ASSIGNENTRYID will be ignored.

ASSIGNENTRYID is valid only when the structure is allocated in a coupling facility of CFLEVEL=8 or higher.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 12-byte field that contains the ENTRYID to be assigned.



**,ASSIGNLISTKEY=NO**  
**,ASSIGNLISTKEY=CREATE**  
**,ASSIGNLISTKEY=MOVE**  
**,ASSIGNLISTKEY=ANY**

Use this input parameter to specify how an entry key is to be assigned to the list entry if it is moved or created as a result of this request when ENTRYTYPE=ANY. If the list entry is created as a result of this request and the structure was allocated to use secondary keys, the secondary key value specified by SECONDARYKEY will be assigned to the list entry. If the user does not supply SECONDARYKEY, the secondary key will be set to all binary zeros.

The ASSIGNLISTKEY specification in conjunction with the TARGETKEY and ENTRYKEY specifications are used to assign the entry key. Note especially that the ENTRYKEY specified for KEYCOMPARE may be used as the entry key assignment for a newly created list entry depending on the ASSIGNLISTKEY specification.

The list key value and maximum list key values may be set with an IXLLSTC WRITE\_LCONTROLS request.

The current list key value can be assigned to a list entry only on structures allocated in a coupling facility of CFLEVEL=1 or higher.

#### **NO**

The list key value is not to be assigned to the list entry, regardless of whether the list entry is moved or created as a result of this request. Instead, the entry key is assigned as follows:

- If the entry is created as a result of this request:
  - The list entry is assigned the TARGETKEY, if specified.
  - If TARGETKEY is not specified, the list entry is assigned the ENTRYKEY, if specified.
  - If neither TARGETKEY nor ENTRYKEY is specified, the entry key is assigned as follows:
    - If DIRECTIONHEADTOTAIL is specified (explicitly or by default), the entry key is assigned all binary zeros.
    - If DIRECTIONTAILTOHEAD is specified, the entry key is assigned all binary ones.
- If the entry is moved as a result of this request:
  - The list entry is assigned the TARGETKEY, if specified.
  - If TARGETKEY is not specified, the entry key remains unchanged.

#### **CREATE**

The list key value of the target list (and therefore, the target list keyed position) is to be assigned to the list entry if it is created as a result of this request.

If the list entry is moved as a result of this request, the entry key is assigned as follows:

- The list entry is assigned the TARGETKEY, if specified.
- If TARGETKEY is not specified, the list entry is assigned the ENTRYKEY, if specified.
- If neither TARGETKEY nor ENTRYKEY is specified, the entry key remains unchanged.

#### **MOVE**

The list key value of the target list (and therefore, the target list keyed position) is to be assigned to the list entry if it is moved as a result of this request.

If a new list entry is created as a result of this request, the entry key is assigned as follows:

- The list entry is assigned the TARGETKEY, if specified.
- If TARGETKEY is not specified, the list entry is assigned the ENTRYKEY, if specified.
- If neither TARGETKEY nor ENTRYKEY is specified, the entry key is assigned as follows:
  - If DIRECTION=HEADTOTAIL is specified (explicitly or by default), the entry key is assigned all binary zeros.
  - If DIRECTION=TAILTOHEAD is specified, the entry key is assigned all binary ones.

**ANY**

The list key value of the target list is to be assigned to the list entry whether the list entry is moved or created as a result of this request.

**,AUTHCOMP=*authcomp***

Use this input parameter to specify an unsigned fixed 128-bit value to be compared to the list authority value for the designated list. If the comparison criterion specified by AUTHCOMPTYPE is not met, the IXLLSTE request is ended.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the 16-byte field that contains the list authority value.

**,AUTHCOMPARE=NO****,AUTHCOMPARE=YES**

Use this input parameter to specify whether list authority comparison is to be performed to determine whether the entry should be created or processed.

The AUTHCOMPARE keyword is only meaningful for list structures allocated in a coupling facility of CFLEVEL=1 or higher.

**NO**

No list authority comparison is to be performed to determine whether the entry is to be created or processed.

**YES**

List authority comparison is to be performed to determine whether the entry is to be created or processed.

**,AUTHCOMPTYPE=EQUAL****,AUTHCOMPTYPE=LESSOREQUAL**

Use this input parameter specify how the list audit comparison is to be performed.

**EQUAL**

The list authority for the list specified by LISTNUM must be equal to the value specified for AUTHCOMP.

**LESSOREQUAL**

The list authority for the list specified by LISTNUM must be less than or equal to the value specified for AUTHCOMP.

**Note:** The AUTHCOMPTYPE parameter is valid only for list structures allocated in a coupling facility with CFLEVEL=1 or higher.

**,BUFADDRSIZE=31****,BUFADDRSIZE=64**

Use this input parameter to specify whether a 31-bit or a 64-bit address is specified by a BUFLIST entry.

**31**

The entry in BUFLIST is 31 bits in size.

**64**

The entry in BUFLIST is 64 bits in size.

**,BUFADDRTYPE=VIRTUAL****,BUFADDRTYPE=REAL**

Use this input parameter to specify whether the buffer addresses specified in the BUFLIST list are virtual storage or real storage addresses.

**VIRTUAL**

The buffer addresses are virtual storage addresses. The virtual storage can be pageable or nonpageable. See the PAGEABLE parameter for information about managing storage binds when specifying virtual storage addresses.

**REAL**

The buffer addresses are real storage addresses.

It is the caller's responsibility to manage the binds between the data buffer virtual storage and the real storage addresses provided. The caller must ensure that the data buffer virtual storage remains bound to the real storage addresses provided until the request completes.

**,BUFALET=NO\_BUFALET**

**,BUFALET=*bufalet***

Use this input parameter to specify an access list entry token (ALET) to be used in referencing all of the buffers specified by BUFLIST.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 4-byte field that contains the ALET.

**,BUFFER=*buffer***

Use this output or input parameter to hold entry data to be read from or written to the designated entry. The BUFSIZE keyword specifies the size of the buffer. Only 31-bit addressable virtual storage areas (below 2GB) are supported by the BUFFER specification. High virtual storage areas (above 2GB) can only be specified via the BUFLIST specification.

You can define the buffer size to be a total size of up to 65536 bytes. Depending on the size you select, the following restrictions apply:

- If you specify a buffer size of less than or equal to 4096 bytes, you must ensure that the buffer:
  - Is 256, 512, 1024, 2048, or 4096 bytes.
  - Starts on a 256-byte boundary.
  - Does not cross a 4096-byte boundary.
- If you specify a buffer size of greater than 4096 bytes, you must ensure that the buffer:
  - Is a multiple of 4096 bytes.
  - Is less than or equal to 65536 bytes.
  - Starts on a 4096-byte boundary.

If BUFFER or BUFLIST is not specified, entry data is not read or written, as appropriate.

BUFFER is only functional for structures that support entry data. If the structure does not support entry data, the contents of BUFFER will not affect this request.

See the BUFSIZE parameter description for defining the size of the buffer.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an area (with a length of BUFSIZE) that contains the entry data.

**,BUFINCRNUM=*bufincrnum***

Use this input parameter to specify the number of 256-byte segments comprising each buffer in the BUFLIST list.

Valid BUFINCRNUM values are 1, 2, 4, 8, or 16, which correspond to BUFLIST buffer sizes of 256, 512, 1024, 2048, and 4096 bytes respectively.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 1-byte field that contains 1, 2, 4, 8, or 16.

**,BUFLIST=*buflist***

Use this output or input parameter to specify a 128-byte storage area containing a list of addresses of buffers that contain data to be read from or written to the designated entry.

Either 31-bit addressable (below 2GB) or 64-bit addressable (above 2GB) real or virtual storage areas are supported for the BUFLIST specification, depending on the specifications for the BUFADDRTYPE and BUFADDRSIZE keywords. However, pageable high shared virtual storage areas (above 2GB) may not be used.

**The 128-byte storage area must:**

- Consist of 0 to 16 elements.
- Each element must consist of an 8-byte field in which:

- The left (high-order) four bytes are reserved and
- The right (low-order) four bytes contain the address of a buffer.

**The BUFLIST buffers must:**

- Reside in either the same address space or data space.
- Be the same size: either 256, 512, 1024, 2048, or 4096 bytes.
- Start on a 256-byte boundary and not cross a 4096-byte boundary.

**Note:** The buffers do not have to be contiguous in storage. (List services treats BUFLIST buffers as a single, contiguous buffer, even if the buffers are not contiguous.)

If BUFFER or BUFLIST is not specified, entry data is not to be read or written.

BUFLIST is only functional for structures that support entry data. If the structure does not support entry data, the contents of BUFLIST will not affect this request.

See the BUFNUM and BUFINCRNUM parameter descriptions for defining the number and size of buffers.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 128-byte area that contains the list of buffer addresses.

**,BUFNUM=*bufnum***

Use this input parameter to specify the number of buffers in the BUFLIST list. Valid BUFNUM values are from 0 to 16. A value of zero indicates that no data is to be read into the buffers or written to the list entry.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 1-byte field that contains the number of buffers in the buffer list.

**,BUFSIZE=*bufsize***

Use this input parameter to specify the size of the BUFFER area. See the BUFFER parameter description for valid buffer sizes.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a fullword field that contains the size of the buffer (BUFFER) in bytes.

**,BUFSTGKEY=CALLERS\_KEY****,BUFSTGKEY=*bufstgkey***

Use this input parameter to specify a storage key that you define and use when referencing the buffers specified by BUFLIST or the buffer that is specified by BUFFER.

If you do not specify BUFSTGKEY, or if you specify BUFSTGKEY=CALLERS\_KEY, all references to one or more buffers are performed by using the caller's PSW key.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of an 8-bit field that contains the storage key in the format B'kkkkxxxx', where kkkk is the key and xxxx is ignored.

**CONTOKEN=*contoken***

Use this input parameter to specify the connect token that was returned by the IXLCONN service. The CONTOKEN uniquely identifies the user's connection to the list structure.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-byte field that contains the connect token.

**,CURSORUPDTYPE=NEXT****,CURSORUPDTYPE=NEXTCOND****,CURSORUPDTYPE=CURRENT****,CURSORUPDTYPE=CURRENTCOND**

Use this input parameter to specify how the list cursor is to be updated when UPDATECURSOR=YES is specified.

**Note:** The NEXTCOND, CURRENT, and CURRENTCOND values are valid only for list structures allocated in a coupling facility with CFLEVEL=1 or higher.

**NEXT**

Set the list cursor as follows:

- If the list entry is created, the cursor for the list specified by MOVETOLIST or LISTNUM is updated and the direction of the cursor update depends on the value specified for MOVETODIRECTION or DIRECTION.
- For an existing list entry, the cursor for the list on which the designated entry resides before the request is processed is updated and the direction of the cursor update depends on the value specified for DIRECTION.
- When MOVETODIRECTION or DIRECTION=HEADTOTAIL is specified (explicitly or by default), the cursor is set to point to the list entry following the created or processed list entry, except when it is the tail entry of the list, in which case the cursor is set to all binary zeros.
- When MOVETODIRECTION or DIRECTION=TAILTOHEAD is specified, the cursor is set to point to the list entry preceding the created or processed list entry, except when it is the head entry of the list, in which case the cursor is set to all binary zeros.

**NEXTCOND**

Set the list cursor as follows:

- If the list cursor points to the list entry processed for the request, and the list entry is deleted or moved to another list, set the cursor to point to the preceding or following list entry, depending on the DIRECTION specified. If the deleted or moved list entry was the last entry on the list and DIRECTION=HEADTOTAIL, or the deleted or moved list entry was the first entry on the list and DIRECTION=TAILTOHEAD, set the list cursor to binary zeros.
- If the list cursor points to the list entry processed for the request, and the list entry is moved to the same list, the cursor will remain pointing to the moved list entry.

The list cursor direction may be set by specifying SETCURSOR on an IXLLSTC WRITE\_LCONTROLS request.

**CURRENT**

Set the list cursor as follows:

- If the list entry is created, set the list cursor to point to the created list entry.
- If the list entry is deleted or moved to another list, set the list cursor to binary zeros.
- If the list entry is moved to the same list, leave the list cursor to remain pointing to the moved list entry.
- For all other cases where the list entry is processed and remains on the same list, set the list cursor to point to the processed list entry.

**CURRENTCOND**

Set the list cursor as follows:

- If the list entry is created, and if the list cursor does not point to a list entry, set the list entry to point to the created list entry.
- If the list entry for the list specified for LISTNUM does not point to a list entry, and the list entry is moved to another list, the list cursor will remain binary zeros.
- If the list cursor does not point to a list entry, and the list entry is moved from and to the same list, set the cursor to point to the moved list entry.
- For all other cases where the list cursor does not point to a list entry, and the list entry is processed and remains on this same list, set the list cursor to point to the processed list entry.

If the list entry is deleted or moved to another list, then the list cursor will remain binary zeros.

**,DIRECTION=HEADTOTAIL****,DIRECTION=TAILTOHEAD**

Use this input parameter to specify the position on the list or sublist at which:

- To place the newly created list entry,

- To designate a list entry to be processed, or
- To specify the direction relative to the current entry in which the cursor is to be updated to point to the next entry on the list when CURSORUPDTYPE=NEXT is specified with UPDATECURSOR=YES.

This parameter is used in conjunction with LISTNUM, and with the assigned entry key value when keyed entries are being used.

### HEADTAIL

Place the newly created list entry (ENTRYTYPE=NEW) according to the following:

- If the structure was not allocated to use keyed entries, the designated position is at the head of the list designated by LISTNUM.
- If the structure was allocated to use keyed entries, and ASSIGN=NONE is specified, the designated position is at the head of the list.
- If the structure was allocated to use keyed entries, and ASSIGN=KEY is specified, the designated position is at the head of the sublist designated by ENTRYKEY and LISTNUM.
- If the structure was allocated to use keyed entries, and ASSIGN=LISTKEY is specified, the designated position is at the head of the sublist designated by the list key value and LISTNUM.

Process the existing list entry (ENTRYTYPE=OLD) according to the following:

- If the structure was allocated with keyed entries and LOCATOR=KEYPOS, process the list entry at the head of the sublist designated by LISTNUM, ENTRYKEY, and KEYREQTYPE.
- If the structure was allocated with secondary keys and LOCATOR=KEYPOS, process the list entry at the head of the sublist designated by LISTNUM, SECONDARYKEY, and SKEYREQTYPE.
- If LOCATOR=UNKEYPOS is specified, process the list entry at the head of the list specified by LISTNUM.

Process the list entry (ENTRYTYPE=ANY) according to the following:

- If LOCATOR=UNKEYPOS is specified, process the list entry at the head of the list designated by LISTNUM.
- If the structure was allocated to use keyed entries and LOCATOR=KEYPOS is specified, process the list entry at the head of the sublist designated by LISTNUM, ENTRYKEY, and KEYREQTYPE.
- If the structure was allocated with secondary keys and LOCATOR=KEYPOS is specified, process the list entry at the head of the sublist designated by LISTNUM, SECONDARYKEY, and SKEYREQTYPE.
- If a list entry is created as a result of this request, and the structure was not allocated to use keyed entries, place the newly created entry at the head of the list designated by LISTNUM.
- If a list entry is created as a result of this request, and the structure was allocated to use keyed entries, place the newly created entry at the head of the sublist designated by both LISTNUM and the assigned entry key.

With UPDATECURSOR=YES, process the list cursor as follows:

- If CURSORUPDTYPE=NEXT is specified, set the cursor to point to the list entry following the processed list entry, except when the processed list entry is at the tail of the list. In that case, the cursor is set to all binary zeros.
- If CURSORUPDATE=NEXTCOND, set the cursor as follows:
  - If the list cursor points to the list entry processed for the request, and the list entry is deleted or moved to another list, set the cursor to point to the following list entry.
  - If the deleted or moved list entry was the last entry on the list, set the cursor to all binary zeros.
  - If the list cursor points to the list entry processed for the request, and the list entry is moved to the same list, the cursor will remain pointing to the moved list entry.
- If CURSORUPDATE=CURRENT, set the cursor as follows:

- If the list entry is deleted or moved to another list, reset the list cursor to binary zeros.
- If the list entry is moved to the same list, the cursor will remain pointing to the moved list entry.
- For all other cases where the list entry is processed and remains on the same list, the list cursor will be set to point to the processed list entry.
- If CURSORUPDATE=CURRENTCOND, set the cursor as follows:
  - If the list cursor for the list specified by LISTNUM does not point to a list entry, and the list entry is moved to another list, the list cursor will remain binary zeros.
  - If the list cursor for the list specified for LISTNUM does not point to a list entry, and the list entry is moved from and to this same list, the cursor will be set to point to the moved list entry.
  - For all other cases where the list cursor for the list specified for LISTNUM does not point to a list entry, and the list entry is processed and remains on this same list, the list cursor will be set to point to the processed list entry.

### TAILTOHEAD

Place the newly created list entry (ENTRYTYPE=NEW) according to the following:

- If the structure was not allocated to use keyed entries, the designated position is at the tail of the list designated by LISTNUM.
- If the structure was allocated to use keyed entries, and ASSIGN=NONE is specified, the designated position is at the tail of the list designated by LISTNUM.
- If the structure was allocated to use keyed entries, and ASSIGN=KEY is specified, the designated position is at the tail of the sublist designated by ENTRYKEY and LISTNUM.
- If the structure was allocated to use keyed entries, and ASSIGN=LISTKEY is specified, the designated position is at the tail of the sublist designated by the list key value and LISTNUM.

Process the existing list entry (ENTRYTYPE=OLD) according to the following:

- If the structure was allocated to use keyed entries and LOCATOR=KEYPOS, process the list entry at the tail of the sublist designated by LISTNUM, ENTRYKEY, and KEYREQTYPE.
- If the structure was allocated to use secondary keys and LOCATOR=KEYPOS, process the list entry at the tail of the sublist designated by LISTNUM, SECONDARYKEY, and SKEYREQTYPE.
- If LOCATOR=UNKEYPOS is specified, process the list entry at the tail of the list designated by LISTNUM.

Process the list entry (ENTRYTYPE=ANY) according to the following:

- If LOCATOR=UNKEYPOS is specified, process the list entry at the tail of the list designated by LISTNUM.
- If the structure was allocated to use keyed entries and LOCATOR=KEYPOS is specified, process the list entry at the tail of the sublist designated by LISTNUM, ENTRYKEY, and KEYREQTYPE.
- If the structure was allocated with secondary keys and LOCATOR=KEYPOS is specified, process the list entry at the tail of the sublist designated by LISTNUM, SECONDARYKEY, and SKEYREQTYPE.
- If a list entry is created as a result of this request, and the structure was not allocated to use keyed entries, place the newly created entry at the tail of the list designated by LISTNUM.
- If a list entry is created as a result of this request, and the structure was allocated to use keyed entries, place the newly created entry at the tail of the sublist designated by both LISTNUM and the assigned entry key.

With UPDATECURSOR=YES, process the list cursor as follows:

- If CURSORUPDTYPE=NEXT is specified, set the cursor to point to the list entry preceding the processed list entry, except when the processed list entry is at the head of the list. In that case, the cursor is set to all binary zeros.

- If CURSORUPDATE=NEXTCOND, set the cursor as follows:
  - If the list cursor points to the list entry processed for the request, and the list entry is deleted or moved to another list, set the cursor to point to the preceding list entry.
  - If the deleted or moved list entry was the first entry on the list, set the cursor to all binary zeros.
  - If the list cursor points to the list entry processed for the request, and the list entry is moved to the same list, the cursor will remain pointing to the moved list entry.
- If CURSORUPDATE=CURRENT, set the cursor as follows:
  - If the list entry is deleted or moved to another list, reset the list cursor to binary zeros.
  - If the list entry is moved to the same list, the cursor will remain pointing to the moved list entry.
  - For all other cases where the list entry is processed and remains on the same list, the list cursor will be set to point to the processed list entry.
- If CURSORUPDATE=CURRENTCOND, set the cursor as follows:
  - If the list cursor for the list specified by LISTNUM does not point to a list entry, and the list entry is moved to another list, the list cursor will remain binary zeros.
  - If the list cursor for the list specified for LISTNUM does not point to a list entry, and the list entry is moved from and to this same list, the cursor will be set to point to the moved list entry.
  - For all other cases where the list cursor for the list specified for LISTNUM does not point to a list entry, and the list entry is processed and remains on this same list, the list cursor will be set to point to the processed list entry.

**,ENTRYDISP=KEEP****,ENTRYDISP=DELETE**

Use this input parameter to specify whether an existing list entry when ENTRYTYPE=OLD, having been read, is to remain on the list or is to be deleted.

**KEEP**

The list entry is to remain on the list after the read is performed.

**DELETE**

The list entry is to be deleted after the read is performed.

**,ELEMNUM=*elemnum***

Use this input parameter to specify the number of elements to be allocated to the existing or new data entry. Valid ELEMNUM values are from zero to the MAXELEMNUM value that was specified on the IXLCONN macro.

Considerations for specifying ELEMNUM are:

- If this request creates a new entry, the number of elements is set to the ELEMNUM value.
- If the entry exists, the number of elements is updated to the ELEMNUM value specified.

**Note:** If the data from the buffers exceeds the amount of space in the elements, the data is truncated. If the data from the buffers is less than the amount of space in the elements, the remaining space is padded with binary zeros.

If you specify an ELEMNUM value of zero:

- No entry data is written to the new entry.
- Any entry data in the existing entry will be deleted.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 1-byte field that contains the number of elements.

**,ENTRYID=*entryid***

Use this input parameter to specify:



- An entry identifier to be used to identify an existing list entry to be processed.
- An entry identifier to be assigned to a newly created list entry.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 12-byte field that contains the entry identifier.

**,ENTRYKEY=entrykey**

Use this input parameter to designate an unsigned fixed 128-bit entry key, which can be used to:

- Specify an entry key to be used in conjunction with LISTNUM, DIRECTION, and KEYREQTYPE to locate the entry to be processed.
- Specifies an entry key to be assigned to the list entry if it is created. If there exists a sublist of one or more entries with a matching key on the list, the target position is at the head or tail of the sublist, as specified by DIRECTION. If all existing list entries have a key greater than that specified by ENTRYKEY, the target position is at the head of the list. Similarly, if the specified ENTRYKEY exceeds all of the entry keys, the target position is at the tail of the list. If none of the list entries has a matching key, and ENTRYKEY is neither the greatest nor least among the entry keys, the target position is determined according to key sequence for the list.
- For structures allocated in a coupling facility of CFLEVEL=9 or higher, when the request specifies KEYCOMPARE=YES, specifies the entry key to be compared to the entry key of the designated entry.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-byte field that contains the entry key.

**,ENTRYNAME=entryname**

Use this input parameter to either:

- Assign an entry name to a new entry to be created.
- Designate an existing entry to be processed.

Specify ENTRYNAME only for structures that support named entries. Each entry name is unique within the structure.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-byte field that contains the entry name.

**,ENTRYTYPE=NEW**

**,ENTRYTYPE=OLD**

**,ENTRYTYPE=ANY**

Use this input parameter to specify the state in which the list entry is expected to be prior to performing the requested operation.

**NEW**

Write to a new list entry the contents of either the storage area specified by BUFFER or the buffers specified by BUFLIST, and the storage area specified by ADJAREA. LISTNUM designates the list number on which the created entry is to reside. The value of ASSIGN specifies whether to assign an ENTRYNAME, ENTRYKEY, or list key value to the newly created list entry. If the structure was allocated to use secondary keys, a secondary key value may be assigned to the newly created list entry.

- If the structure supports entry data and BUFFER and BUFLIST are not specified, the created entry will not contain any entry data.
- If the structure supports adjunct data and ADJAREA is not specified, the created entry will contain binary zeros for adjunct data.

When the amount of space indicated by ELEMNUM is greater than the amount of data specified for BUFFER or BUFLIST, the entry will be padded with zeros. When the amount of space indicated by ELEMNUM is less than the amount of data specified for BUFFER or BUFLIST, the input data will be truncated.

Specifying AUTHCOMPARE in conjunction with AUTHCOMP causes the list authority for the list designated by LISTNUM to precede processing. If the list authority comparison criterion is not met, the IXLLSTE request is terminated.

If LOCKINDEX is specified, the lock entry designated by LOCKINDEX will be operated on as specified by the LOCKOPER keyword.

The new list entry will be created as follows:

- If the structure was not allocated to use named entries or keyed entries, the newly created entry will be placed on the list specified by LISTNUM at the head or tail as specified by DIRECTION.
- If the structure was allocated to use named entries, the entry name specified for ENTRYNAME is assigned to the newly created list entry, provided a list entry does not already exist with the same entry name. The newly created entry will be placed on the list specified by LISTNUM at the head or tail as specified by DIRECTION.
- If the structure was allocated to use keyed entries, an entry key (and therefore, the target list keyed position) is assigned to the newly created list entry as follows:
  - If ASSIGN=NONE is specified, then:
    - If DIRECTION=HEADTOTAIL is specified (explicitly or by default), the newly created list entry is assigned an entry key value of all binary zeros.
    - If DIRECTION=TAILTOHEAD is specified, the newly created list entry is assigned an entry key value of all binary ones.
  - If ASSIGN=KEY is specified, the newly created list entry is assigned the value specified for ENTRYKEY.
  - If ASSIGN=LISTKEY is specified, the newly created list entry is assigned the list key value.
  - If the assigned entry key is zero, the newly created list entry is placed at the head of the list specified by LISTNUM.
  - If the assigned entry key is non-zero, the newly created list entry is placed on the list specified by LISTNUM at the head or tail of the sublist composed of list entries whose entry keys are equal to the assigned entry key. The newly created list entry is placed at the head or tail of this sublist as specified by DIRECTION. If a sublist of entries with entry keys equal to the assigned entry key does not yet exist, the newly created list entry is placed on the list in key sequence.
- If the structure was allocated to use user-provided ENTRYIDs, the newly created list entry is assigned the ENTRYID specified by ASSIGNENTRYID, provided a list entry does not already exist with the same ENTRYID.
- If the structure supports secondary keys, the secondary key specified by SECONDARYKEY will be assigned to the newly created list entry. If the user does not supply SECONDARYKEY, the newly created list entry is assigned a secondary key of all binary zeros.
  - The newly created list entry is placed on the list specified by LISTNUM at the head or tail, relative to secondary key ordering, of the sublist composed of list entries whose secondary keys are equal to the assigned secondary key. The newly created list entry is placed at the head or tail of this sublist as specified by SKEYTARGETDIR. If a sublist of entries with secondary keys equal to the assigned secondary key does not yet exist, the target position relative to secondary key ordering is determined according to secondary key sequence for the list.

When the request completes successfully, the list entry controls, the number of entries or elements residing on the list, and the total number of allocated entries in the structure are returned in the answer area specified by ANSAREA.

#### **OLD**

Perform a read, write, move, or delete operation on an existing entry designated by the LOCATOR keyword. If the entry is not found, the requested operation is not performed and no change is made to the list structure.

Specifying AUTHCOMPARE in conjunction with AUTHCOMP causes the list authority for the list designated by LISTNUM to precede processing. If the list authority comparison criterion is not met, the IXLLSTE request is terminated.

If LOCKINDEX is specified, the lock entry designated by LOCKINDEX will be operated on as specified by the LOCKOPER keyword.

If the lock comparison criterion is not met, the IXLLSTE request is terminated.

In order for the request to be performed, any and all requested comparison criteria must be met. Namely, authority comparison, list number comparison, version number comparison, lock comparison, and key comparison must each succeed if requested.

#### **ANY**

Perform the specified operation on the designated entry if it exists. If the designated entry does not exist, a new entry will be created.

Specifying AUTHCOMPARE in conjunction with AUTHCOMP causes the list authority for the list designated by LISTNUM to precede processing. If the list authority comparison criterion is not met, the IXLLSTE request is terminated.

If LOCKINDEX is specified, the lock entry designated by LOCKINDEX will be operated on as specified by the LOCKOPER keyword.

In order for the request to be performed on an existing list entry, any and all requested comparison criteria must be met. Namely, list number comparison, version number comparison, lock comparison, and key comparison must each succeed if requested.

#### **,KEYCOMPARE=NO**

#### **,KEYCOMPARE=YES**

Use this input parameter to specify whether key comparison should be performed to determine whether the list entry should be processed.

#### **NO**

No key comparison should be performed to determine whether the list entry should be processed.

#### **YES**

Key comparison should be performed to determine whether the list entry should be processed. If the designated list entry exists but the requested key comparison criterion is not met, the IXLLSTE request is terminated.

KEYCOMPARE=YES is only meaningful when the structure is allocated in a coupling facility of CFLEVEL=9 or higher. KEYCOMPARE=YES will be ignored if the target structure was not allocated with keyed list entries.

#### **,KEYPOSITION=UPDATE**

#### **,KEYPOSITION=KEEP**

Use this input parameter to specify whether the list entry is moved from its current position on the sublist. This keyword has effect only if the list number specified by MOVETOLIST is the same as the list number on which the list entry currently resides.

The KEYPOSITION keyword is only meaningful for list structures allocated in a coupling facility of CFLEVEL=9 or higher.

#### **UPDATE**

The list entry should be moved from its current position on the sublist to a position on the sublist as specified by MOVETODIRECTION and MOVETOKEY.

#### **KEEP**

The list entry should keep its current position based on entry key ordering on the sublist if and only if the list number specified by MOVETOLIST matches the current list number that contains the list entry, and the list entry is not changed by the move operation when MOVETOKEY=UNCHANGED is specified (explicitly or by default).

**,KEYREQTYPE=EQUAL****,KEYREQTYPE=LESSOREQUAL****,KEYREQTYPE=GREATEROREQUAL**

Use this input parameter to specify how key comparison is to be performed on the designated keyed list entry or, when LOCATOR=KEYPOS is specified, to designate the list entry to be processed.

**EQUAL**

- The designated list entry must have a key equal to the value specified for ENTRYKEY. If the entry key is not equal to the value specified for ENTRYKEY, the IXLLSTE request is terminated.
- When LOCATOR=KEYPOS, specifies that if a list entry exists with an entry key equal to the value specified for ENTRYKEY, that list entry is to be processed. If more than one such entry exists, the list entry at either the head or tail of the sublist as specified by DIRECTION is designated for processing. If no such entry exists, the IXLLSTE request is terminated.

**LESSOREQUAL**

- The designated list entry must have an entry key that is equal to or less than the value specified for ENTRYKEY. If the entry key is not equal to or less than the value specified for ENTRYKEY, the IXLLSTE request is terminated.
- When LOCATOR=KEYPOS, specifies that:
  - If a list entry exists with an entry key equal to the value specified for ENTRYKEY, that entry is designated for processing. If more than one such entry exists, the list entry at the head or tail of the sublist as specified by DIRECTION is designated for processing.
  - If no list entries exist with an equal entry key, the list entry with the highest entry key less than the value specified for ENTRYKEY is designated for processing. If more than one such entry exists, the list entry at the head or tail of the sublist as specified by DIRECTION is designated for processing.
  - If no list entries exist with an entry key equal to or less than the value specified for ENTRYKEY, the IXLLSTE request is terminated.

**GREATEROREQUAL**

- The designated list entry must have an entry key that is equal to or greater than the value specified for ENTRYKEY. If the entry key is not equal to or greater than the value specified for ENTRYKEY, the IXLLSTE request is terminated.
- When LOCATOR=KEYPOS, specifies that:
  - If a list entry exists with an entry key equal to the value specified for ENTRYKEY, that entry is designated for processing. If more than one such entry exists, the list entry at the head or tail of the sublist as specified by DIRECTION is designated for processing.
  - If no list entries exist with an equal entry key, the list entry with the lowest entry key greater than the value specified for ENTRYKEY is designated for processing. If more than one such entry exists, the list entry at the head or tail of the sublist as specified by DIRECTION is designated for processing.
  - If no list entries exist with an entry key equal to or greater than the value specified for ENTRYKEY, the IXLLSTE request is terminated.

**,KEYTYPE=ENTRY****,KEYTYPE=SECONDARY**

Use this input parameter to specify whether to locate the list entry using the list entry key or the secondary key.

**ENTRY**

The list entry will be located by list entry key.

**SECONDARY**

The list entry will be located by secondary key.

KEYTYPE=SECONDARY is valid only when the structure is allocated in a coupling facility of CFLEVEL=9 or higher.

**,LISTCOMPARE=NO****,LISTCOMPARE=YES**

Use this input parameter to specify whether list number comparison should be performed to determine whether the list entry should be processed.

**NO**

No list number comparison should be performed to determine whether the list entry should be processed.

**YES**

List number comparison should be performed to determine whether the list entry should be processed.

The list number value specified for LISTNUM is compared with the list number on which the designated list entry resides. If the designated list entry exists but does not reside on the list specified by LISTNUM, the IXLLSTE request is terminated.

**,LISTKEYINC=NO LISTKEYINC****,LISTKEYINC=*listkeyinc***

Use this input parameter to specify a value to be added to the list key after the entry key is set to the list key value. If the entry key is not set to the list key value, the list key value will not be changed. If the result of adding the value specified by LISTKEYINC to the list key value is greater than the maximum list key value, the system terminates the IXLLSTE request.

**Note:** The LISTKEYINC parameter is valid only for list structures allocated in a coupling facility of CFLEVEL=1 or higher.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the fullword field that contains the value to be added to the list key after the entry key is set to the list key value.

**,LISTLIMIT=ENFORCE****,LISTLIMIT=IGNORE**

Use this input parameter to specify whether the current list limits set for the target list should be enforced or ignored on a MOVE request when ENTRYTYPE=OLD. This has effect if the listlimit for the target list has been modified by an IXLLSTC WRITE\_LCONTROLS request and is not equal to the default list limit.

**ENFORCE**

The move request will be failed if the list-entry count would exceed the current list-entry count limit of the target list or the list-element count would exceed the list-element count limit of the target list as a result of the move request.

**IGNORE**

The move request will proceed even if the list-entry count would exceed the current list-entry count limit of the target list or the list-element count would exceed the list-element count of the target list as a result of this move request.

**,LISTNUM=NO LISTNUM****,LISTNUM=*listnum***

Use this input parameter to specify the number of the list on which the entry to be processed currently resides. Use LISTNUM in the following ways:

- Designate the list number (when ENTRYTYPE=NEW) on which the newly created list entry should be placed.
- Check if the entry exists on the list (when ENTRYTYPE=ANY) before proceeding with the request.
- Confirm that the entry exists on the list (when ENTRYTYPE=OLD) before proceeding with the request.

**To Code:** Specify the RS-type name or address (using a 2 register from 2 to 12) of a 4-byte field that contains the number of the list.

**,LOCATOR=CURSOR**  
**,LOCATOR=ENTRYID**  
**,LOCATOR=ENTRYNAME**  
**,LOCATOR=UNKEYPOS**  
**,LOCATOR=KEYPOS**

Use this input parameter to specify the location mechanism to be used to designate the list entry to be processed.

**CURSOR**

The list cursor should be used to designate the list entry to be processed.

**ENTRYID**

The ENTRYID is to be used to designate the list entry to be processed.

**ENTRYNAME**

The entry name specified by ENTRYNAME is to be used to designate the list entry to be processed.

LOCATOR=ENTRYNAME may only be specified when the structure was allocated to use named entries.

**UNKEYPOS**

LISTNUM and DIRECTION are to be used to designate the list entry at the head or tail of the list to be processed.

**KEYPOS**

LISTNUM, DIRECTION, and the key specified by KEYTYPE are to be used to designate the list entry to be processed.

**,LOCKCOMP=NO\_LOCKCOMP**

**,LOCKCOMP=lockcomp**

This parameter has slightly different meanings based on the value that is specified for the LOCKOPER parameter. Generally, this input parameter specifies a connection identifier to be verified as the owner of the lock specified by LOCKINDEX. This verification is a prerequisite to the successful completion of the request.

When LOCKCOMP is specified, the completion of the request is conditional on there being no contention for the lock. If contention exists, the request fails .

The connection identifier is available from the IXLCONN answer area, which is mapped which is by IXLYCONA, in field CONACONID.

The effect of LOCKCOMP is based on the LOCKOPER value specified. See the description under LOCKOPER to see how each request is affected by this parameter.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 1-byte field that contains the connection identifier.

**,LOCKDATA=NO\_LOCKDATA**

**,LOCKDATA=lockdata**

Use this input parameter to specify information that is to be passed to your notify exit when another user requests the LOCKINDEX lock after you have obtained the lock by using LOCKOPER=SET. This user-defined information will be passed to your notify exit when the other user issues a request specifying either one of the following:

- LOCKOPER=SET
- LOCKOPER=NOTHELD with LOCKMODE=UNCOND

**To Code:** Specify the RS-type name or address (by using a register 2 - 12 2 to ) of an 8-byte field that contains the user-defined information.

**,LOCKINDEX=NO\_LOCKINDEX**

**,LOCKINDEX=lockindex**

Use this input parameter to specify the index of the lock to be operated on within the lock table for the list structure.

When specified, the designated lock is operated on as specified by the LOCKOPER keyword. The specified value must fall within the range 0 to the number of lock table entries minus one, inclusive.

**Note:** You cannot code LOCKINDEX with MODE=ASYNCSUSPEND.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a fullword field that contains the lock index.

#### **,LOCKMODE=UNCOND**

#### **,LOCKMODE=COND**

Use this input parameter to specify how contention for the lock that is specified by LOCKINDEX should be handled if your request causes lock contention.

##### **UNCOND**

If the specified lock is held by another user, your request is either:

- Suspended until the lock becomes available (if MODE=SYNCSUSPEND is specified).
- Performed asynchronously with notification to you when the request completes (if any MODE value other than SYNCSUSPEND is specified).

##### **COND**

The lock operation is performed conditionally; that is, only if there is no contention for the lock. If the specified lock is held by another user, the request is terminated with no change to the structure, and return and reason codes describing the termination are returned to the caller.

#### **,LOCKOPER=SET**

#### **,LOCKOPER=RESET**

#### **,LOCKOPER=NOTHELD**

#### **,LOCKOPER=HELD**

Use this input parameter to specify the type of operation to be performed on the lock specified by LOCKINDEX.

##### **SET**

Set the lock.

- When LOCKCOMP is not specified, your connection gets the lock, providing no other connection holds the lock. If another connection holds the lock, the request either continues once the lock is released or fails, depending on the LOCKMODE value you specify.
- When LOCKCOMP is specified, if the connection specified by LOCKCOMP holds the lock, the lock is transferred to your connection.

##### **RESET**

Reset the lock.

- When LOCKCOMP is not specified, the lock is released if it is held by your connection.
- When LOCKCOMP is specified, the lock is released if it is held by the connection that is specified by LOCKCOMP.

##### **NOTHELD**

The request is performed only if the lock is not held by any connection. This option also ensures that the lock remains free for the duration of the request. If another connection holds the lock, your task is suspended until the lock is released or your request fails, depending on the LOCKMODE value you specify. See the LOCKMODE description for how to handle possible lock contention.

##### **HELD**

Processing is determined by which connector is holding the lock.

- When LOCKCOMP is not specified, the request is performed only if your connection holds the lock.
- When LOCKCOMP is specified, the request is performed only if the lock is held by the connection that is specified by LOCKCOMP.

**,MF=S**  
**,MF=(L,mfctrl)**  
**,MF=(L,mfctrl,mfattr)**  
**,MF=(L,mfctrl,0D)**  
**,MF=(M,mfctrl)**  
**,MF=(M,mfctrl,COMPLETE)**  
**,MF=(M,mfctrl,NOCHECK)**  
**,MF=(E,mfctrl)**  
**,MF=(E,mfctrl,COMPLETE)**  
**,MF=(E,mfctrl,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

#### **,mfctrl**

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

#### **,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

#### **,COMPLETE**

#### **,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

#### **COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if SMILE=*var* were an optional parameter and the default is SMILE=NO\_SMILE then it would not be documented. However, if the default was SMILE=:~), then it would be documented because a value would be the default.

#### **NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.



**,MODE=SYNCSUSPEND**  
**,MODE=SYNCECB**  
**,MODE=SYNCEXIT**  
**,MODE=SYNCTOKEN**  
**,MODE=ASYNCECB**  
**,MODE=ASYNCEXIT**  
**,MODE=ASYNCTOKEN**  
**,MODE=ASYNCSNORESPONSE**

Use this input parameter to specify whether the request is to be performed synchronously or asynchronously.

#### **SYNCSUSPEND**

The request is performed synchronously. Control is not returned to the caller until request processing is complete and the final disposition determined.

If necessary, the caller will be suspended until the request completes. The caller must be executing in an enabled state to use this option.

#### **SYNCECB**

The request will be attempted synchronously. If the request cannot be completed synchronously, control is returned to the caller prior to completion of the request, and the ECB specified by REQECB is posted when the request has completed.

If the request does not complete synchronously, latent XES binds to the storage locations specified by BUFFER, BUFLIST, MOSVECTOR, and ANSAREA persist until the REQECB ECB is posted.

#### **SYNCEXIT**

The request will be attempted synchronously. If the request cannot be completed synchronously, control is returned to the caller prior to completion of the request. When the request completes, the connection's Complete Exit will be called.

If the request does not complete synchronously, latent XES binds to the storage locations specified by BUFFER, BUFLIST, MOSVECTOR, and ANSAREA persist until the connection's Complete Exit is called.

#### **SYNCTOKEN**

The request will be attempted synchronously. If the request cannot be completed synchronously, control is returned to the caller prior to completion of the request and a token that uniquely identifies the request is returned. This token must be specified on a subsequent invocation of IXLFCOMP to force completion of the request and determine its final disposition.

If the request does not complete synchronously, latent XES binds to the storage locations specified by BUFFER, BUFLIST, MOSVECTOR, and ANSAREA persist until a subsequent corresponding IXLFCOMP request indicates completion of the original request.

#### **ASYNCECB**

The request is to be initiated and control is to be returned to the caller prior to completion of the request. When the request completes, the ECB specified by REQECB will be posted.

Latent XES binds to the storage locations specified by BUFFER, BUFLIST, MOSVECTOR, and ANSAREA persist until the REQECB ECB is posted.

#### **ASYNCEXIT**

The request is to be initiated and control is to be returned to the caller prior to completion of the request. When the request completes, the connection's Complete Exit will be called.

Latent XES binds to the storage locations specified by BUFFER, BUFLIST, MOSVECTOR, and ANSAREA persist until the connection's Complete Exit is called.

#### **ASYNCTOKEN**

The request is to be initiated, a token generated that uniquely identifies the request on this system, and control returned to the caller prior to completion of the requested operation. The

token must be specified on a subsequent invocation of IXLFCOMP to force completion of the request and determine its final disposition.

Latent XES binds to the storage locations specified by BUFFER, BUFLIST, MOSVECTOR, and ANSAREA persist until a subsequent corresponding IXLFCOMP request indicates completion of the original request.

### **ASYNCSNORESPONSE**

The request is to be initiated and control returned to the caller prior to completion of the requested operation. No asynchronous request token is returned, hence no external mechanism exists to force completion of the request.

MODE=ASYNCSNORESPONSE is mutually exclusive with LOCKINDEX, BUFFER, and BUFLIST. Any request which does not perform a locking operation and does not use a BUFFER area or BUFLIST buffers may specify MODE=ASYNCSNORESPONSE.

### **,MOVETODIRECTION=HEADTOTAIL**

### **,MOVETODIRECTION=TAILTOHEAD**

Use this input parameter in conjunction with MOVETOLIST and the entry key assigned to the list entry when keyed entries are being used, to designate the target position for the moved or created list entry.

#### **HEADTOTAIL**

One of the following:

- If the list entry is moved or created as a result of this request, and the structure was not allocated to use keyed entries, the list entry will be placed at the head of the list designated by MOVETOLIST.
- If the list entry is moved or created as a result of this request, and the structure was allocated to use keyed entries, the entry will be placed at the head of the sublist designated by MOVETOLIST and the assigned entry key.

#### **TAILTOHEAD**

One of the following:

- If the list entry is moved or created as a result of this request, and the structure was not allocated to use keyed entries, the list entry will be placed at the tail of the list designated by MOVETOLIST.
- If the list entry is moved or created as a result of this request, and the structure was allocated to use keyed entries, the entry will be placed at the tail of the sublist designated by MOVETOLIST and the assigned entry key.

### **,MOVETOKEY=NO MOVETOKEY**

### **,MOVETOKEY=movekey**

Use this input parameter to assign a new key, and hence the position on the MOVETOLIST, for the existing entry being moved when ENTRYTYPE=OLD. The existing entry's key will be updated to the MOVETOKEY key value. If MOVETOKEY is not specified, ENTRYKEY will remain unchanged.

#### **Note:**

1. If there is a sublist of one or more entries with a matching key on the list then the target position is the head or tail of the sublist, as specified by the MOVETODIRECTION parameter.
2. If none of the list entries have a matching key, and MOVETOKEY is neither the greatest nor least among the list entry keys, then the target position is according to the appropriate key sequence for the list.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-byte field that contains the key.

### **,MOVETOLIST=movealist**

Use this input parameter to specify the number of the target list where the existing entry being moved or the new entry being created will be placed.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 4-byte field that contains the number of the target list.

**,NEWAUTH=NO\_NEWAUTH****,NEWAUTH=newauth**

Use this input parameter to specify a list authority value for the list specified by LISTNUM. If NEWAUTH is omitted, the list authority for the designated list is unchanged.

When the structure is allocated, the authority value for each list is initialized to binary zeros.

**Note:** The NEWAUTH parameter is valid only for list structures allocated in a coupling facility with CFLEVEL=1 or higher, except on WRITE\_LCONTROLS requests.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12 to ) of the 16-byte field that contains the list authority value.

**,NEWVERS=newvers**

Use this input parameter with VERSUPDATE=SET to specify an unsigned fixed 64-bit value for the list entry version number to either replace the version number of the existing entry, or initialize the version number of a new entry.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-byte field that contains the entry version number.

**,PAGEABLE=YES****,PAGEABLE=NO**

Use this input parameter to identify whether the storage areas specified by BUFFER or BUFLIST reside in pageable storage.

**YES**

Specify this option to indicate that the BUFFER or BUFLIST buffers reside in pageable virtual storage.

This includes disabled reference (DREF) storage, and may include storage that has the potential to become pageable during the processing of a request. (An example is address space storage owned by any swappable address space, for which a PGSER FIX has been successfully processed, but for which the owning address space gets swapped during processing of a cache or list request.) It does not include implicitly non-pageable storage (such as is obtained from non-pageable subpools).

The system takes responsibility for managing binds to central storage for the duration of the list request, regardless of what address space owns the storage or whether the storage-owning address space is swappable or nonswappable. The storage can be owned by any address space.

**NO**

Specify this option to indicate that the BUFFER or BUFLIST buffers reside in non-pageable virtual storage.

This includes implicitly non-pageable storage areas. If the virtual storage may potentially become pageable, the invoker is responsible for ensuring the virtual storage remains non-pageable for the duration of the request. (An example is address space storage owned by any swappable address space, for which a PGSER FIX has been successfully processed, but for which the owning address space gets swapped-out during processing of a list request.)

If MODE=ASYNCTOKEN is specified or MODE=SYNCTOKEN is specified and the request does not complete synchronously, the storage must remain non-pageable until completion of the corresponding IXLFCOMP request. If MODE=ASYNCEXIT is specified or MODE=SYNCEXIT is specified and the request does not complete synchronously, the storage must remain non-pageable until the complete exit is driven for the request. If MODE=ASYNCECB is specified or MODE=SYNCECB is specified and the request does not complete synchronously, the storage must remain non-pageable until the specified ECB is posted for the request.

The system takes responsibility for managing binds to central storage for the duration of the list request, if and only if the non-pageable storage is owned by either the requester's address space or the connector's address space. If the storage is owned by any other address space, then the invoker is responsible for ensuring that the virtual storage remains non-pageable for the duration of the request (including the case in which the storage is owned by a swappable address space

that is swapped during processing of a list request). Subject to this consideration, the storage can be owned by any address space. See *z/OS MVS Programming: Sysplex Services Guide*.

**,PLISTVER=IMPLIED\_VERSION**

**,PLISTVER=MAX**

**,PLISTVER=plistver**

Use this input parameter to specify the version of the macro. See “Understanding IXLLSTE Version Support” on page 1419 for a description of the options available with the PLISTVER macro.

**,REQDATA=NO\_REQDATA**

**,REQDATA=reqdata**

Use this input parameter with MODE=SYNCEXIT or MODE=ASYNCEXIT to pass any data you choose to the complete exit. The exit will get control only if the request is processed asynchronously.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the data to be passed to the complete exit.

**,REQECB=reqecb**

Use this output parameter with either MODE=SYNCECB or MODE=ASYNCECB to specify the address of an ECB, which is to be posted when the request completes if the request was processed asynchronously.

Before coding REQECB, you must ensure that:

- You initialize the ECB before you issue the request.
- The ECB resides in either common storage or the home address space where IXLCONN was issued.
- Any tasks that wait for the ECB to be posted reside in the home address space where IXLCONN was issued.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that contains the address of the ECB to be posted when the request completes. The ECB must be aligned on a fullword boundary.

**,REQID=NO\_REQID**

**,REQID=reqid**

Use this input parameter to specify a user-defined request identifier to be associated with the request. You can specify this request identifier on the IXPURGE macro to cancel a request that has not yet been processed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the user-defined request identifier.

**,REQTOKEN=reqtoken**

Use this output parameter with either MODE=SYNCTOKEN or MODE=ASYNCTOKEN to specify the address of a storage area to receive the request token that is returned when the request will be processed asynchronously. This token, which uniquely identifies the request, must be used as input to the IXLFComp macro, which you use to determine if the request has completed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field where the system will put the request token.

**,REQUEST=READ**

**,REQUEST=WRITE**

**,REQUEST=MOVE**

**,REQUEST=DELETE**

Use this input parameter to specify the type of operation to be performed on the designated list entry.

**READ**

The entry data and adjunct data and list entry controls for the designated entry are to be read.

Entry data is to be read into the storage area specified by BUFFER or the buffers specified by BUFLIST. Adjunct data is to be read into the storage area specified by ADJAREA. The absence of BUFFER, BUFLIST, and ADJAREA specifies the request is only interested in obtaining the list entry controls in the answer area specified by ANSAREA.

When the request completes successfully, the list entry controls, the number of entries or elements residing on the list, and the total number of allocated entries in the structure are returned in the answer area specified by ANSAREA.

## WRITE

- For requests that specify either ENTRYTYPE=OLD or ENTRYTYPE=ANY and the designated entry already exists:

The contents of the storage area specified by BUFFER or the buffers specified by BUFLIST, and the storage area specified by ADJAREA are to be written to the designated list entry.

The size of list entry may be altered by specifying an ELEMNUM value that is different than the ELEMNUM value used when the entry was created. When the amount of space indicated by ELEMNUM is greater than the amount of data specified for BUFFER or BUFLIST, the entry will be padded with zeros. When the amount of space indicated by ELEMNUM is less than the amount of data specified for BUFFER or BUFLIST, the input data will be truncated.

When the request completes successfully, the list entry controls, the number of entries or elements residing on the list, and the total number of allocated entries in the structure are returned in the answer area specified by ANSAREA.

- For requests that specify ENTRYTYPE=ANY and the designated list entry does not exist, a new list entry will be created as follows:
  - If the structure was not allocated to use named entries or keyed entries, the newly created entry will be placed on the list specified by LISTNUM at the head or tail as specified by DIRECTION.
  - If the structure was allocated to use named entries, the entry name specified for ENTRYNAME is assigned to the newly created list entry, provided a list entry does not already exist with the same entry name. The newly created entry will be placed on the list specified by LISTNUM at the head or tail as specified by DIRECTION.
  - If the structure was allocated to use keyed entries, an entry key (and therefore, the target list keyed position) is assigned to the newly created list entry as follows:
    - If ASSIGN=NONE is specified, then:
      - If DIRECTION=HEADTOTAIL is specified (explicitly or by default), the newly created list entry is assigned an entry key value of all binary zeros.
      - If DIRECTION=TAILTOHEAD is specified, the newly created list entry is assigned an entry key value of all binary ones.
    - If ASSIGN=KEY is specified, the newly created list entry is assigned the value specified for ENTRYKEY.
    - If ASSIGN=LISTKEY is specified, the newly created list entry is assigned the list key value.
    - If the assigned entry key is zero, the newly created list entry is placed at the head of the list specified by LISTNUM.
    - If the assigned entry key is non-zero, the newly created list entry is placed on the list specified by LISTNUM at the head or tail of the sublist composed of list entries whose entry keys are equal to the assigned entry key. The newly created list entry is placed at the head or tail of this sublist as specified by DIRECTION. If a sublist of entries with entry keys equal to the assigned entry key does not yet exist, the newly created list entry is placed on the list in key sequence.
  - If the structure was allocated to use user-provided ENTRYIDs, the newly created list entry is assigned the ENTRYID specified by ASSIGNENTRYID, provided a list entry does not already exist with the same ENTRYID.
  - If the structure supports entry data and BUFFER and BUFLIST are not specified, the newly created list entry will not contain any entry data.
  - If the structure supports adjunct data and ADJAREA is not specified, the newly created list entry will contain binary zeros for adjunct data.

- If the structure supports secondary keys, the secondary key specified by SECONDARYKEY will be assigned to the newly created list entry. If the user does not supply SECONDARYKEY, the newly created list entry is assigned a secondary key of all binary zeros.
- The newly created list entry is placed on the list specified by LISTNUM at the head or tail, relative to secondary key ordering, of the sublist composed of list entries whose secondary keys are equal to the assigned secondary key. The newly created list entry is placed at the head or tail of this sublist as specified by SKEYTARGETDIR. If a sublist of entries with secondary keys equal to the assigned secondary key does not yet exist, the target position relative to secondary key ordering is determined according to secondary key sequence for the list.

## MOVE

- When the request specifies ENTRYTYPE=OLD:
  - The designated list entry is to be moved from its current location and placed at the position specified by MOVETOLIST, MOVETODIRECTION, and if keyed entries are being used, the entry key specified by MOVETOKEY.

When ACTION=READ and ACTION=WRITE, in addition to moving the list entry, the entry data and/or adjunct data is read into the storage areas specified by BUFFER or BUFLIST and, if applicable, ADJAREA.

When the request completes successfully, the list entry controls, the number of entries or elements residing on the target list, and the total number of allocated entries in the structure are returned in the answer specified by ANSAREA.

- If the structure was allocated to use keyed entries, an entry key is assigned to the moved list entry as follows:
  - If TARGETKEY is specified with MOVETOKEY=TARGETKEY, the list entry is assigned the entry key specified by TARGETKEY.
  - If MOVETOKEY=LISTKEY, the list entry is assigned the current list key value of the target list.
  - If MOVETOKEY=UNCHANGED, the entry key of the list entry remains unchanged.
- When the request specifies ENTRYTYPE=ANY and ACTION=WRITE, and the designated list entry exists:
  - In addition to moving the list entry, the data in the storage area(s) specified by BUFFER or BUFLIST and, if applicable ADJAREA, is written to the list entry. If BUFFER or BUFLIST is not specified, entry data is not to be written. If ADJAREA is not specified, structure adjunct data will not be written.
  - The size of an existing list entry may be altered by specifying an ELEMNUM value that is different from the ELEMNUM value used when the entry was created. When the amount of space indicated by ELEMNUM is greater than the amount of data specified for BUFFER or BUFLIST, the entry will be padded with zeros. When the amount of space indicated by ELEMNUM is less than the amount of data specified for BUFFER or BUFLIST, the input data will be truncated.
  - The designated list entry will be moved as follows:
    - If the structure was not allocated to use keyed entries, the list entry will be placed on the list specified by MOVETOLIST at the head or tail as specified by MOVETODIRECTION.
    - The moved list entry is placed on the list specified by MOVETOLIST at the head or tail of the sublist composed of list entries whose entry keys are equal to the assigned entry key. The moved list entry is placed at the head or tail of this sublist as specified by MOVETODIRECTION. If a sublist of entries with entry keys equal to the assigned entry key does not yet exist, the moved list entry is placed on the list in key sequence.
    - The moved list entry is placed on the list specified by MOVETOLIST at the head or tail of the sublist composed of list entries whose secondary keys are equal to the assigned

secondary key. The moved list entry is placed at the head or tail of this sublist as specified by SKEYTARGETDIR. If a sublist of entries with secondary keys equal to the assigned secondary key does not yet exist, the moved list entry is placed on the list in secondary key sequence.

- If the structure was allocated to use keyed entries, an entry key (and therefore, the target list keyed position) is assigned to the list entry as follows:
  - If TARGETKEY=NO\_TARGETKEY is specified (explicitly or by default) with ASSIGNLISTKEY=NO or with ASSIGNLISTKEY=CREATE, the entry key remains unchanged.
  - If TARGETKEY is specified with ASSIGNLISTKEY=NO or with ASSIGNLISTKEY=CREATE, the entry key is assigned the value specified for TARGETKEY.
  - If ASSIGNLISTKEY=MOVE or ASSIGNLISTKEY=ANY is specified, the entry key is assigned the list key value of the target list.
- When the request specifies ENTRYTYPE=ANY and the designated list entry does not exist, a new list entry will be created as follows:
  - If the structure was not allocated to use named entries or keyed entries, the newly created entry will be placed on the list specified by MOVETOLIST at the head or tail as specified by MOVETODIRECTION.
  - If the structure was allocated to use named entries, the entry name specified for ENTRYNAME is assigned to the newly created list entry, provided a list entry does not already exist with the same entry name. The newly created entry will be placed on the list specified by MOVETOLIST at the head or tail as specified by MOVETODIRECTION.
  - If the structure was allocated to use keyed entries, an entry key (and therefore, the target list keyed position) is assigned to the newly created list entry as follows:
    - If ENTRYKEY is not specified, and TARGETKEY=NO\_TARGETKEY is specified (explicitly or by default) with ASSIGNLISTKEY=NO or with ASSIGNLISTKEY=MOVE, then:
      - If MOVETODIRECTION=HEADTOTAIL is specified (explicitly or by default), the newly created list entry is assigned an entry key value of all binary zeros.
      - If MOVETODIRECTION=TAILTOHEAD is specified, the newly created list entry is assigned an entry key value of all binary ones.
    - If ENTRYKEY is specified, and TARGETKEY=NO\_TARGETKEY is specified (explicitly or by default) with ASSIGNLISTKEY=NO or with ASSIGNLISTKEY=MOVE, the newly created list entry is assigned the value specified by ENTRYKEY.
    - If TARGETKEY is specified with ASSIGNLISTKEY=NO or with ASSIGNLISTKEY=MOVE, the newly created list entry is assigned the value specified for TARGETKEY.
    - If ASSIGNLISTKEY=CREATE or ASSIGNLISTKEY=ANY is specified, the newly created list entry is assigned the list key value of the target list.
    - The newly created list entry is placed on the list specified by MOVETOLIST at the head or tail of the sublist composed of list entries whose entry keys are equal to the assigned entry key. The newly created list entry is placed at the head or tail of this sublist as specified by MOVETODIRECTION. If a sublist of entries with entry keys equal to the assigned entry key does not yet exist, the newly created list entry is placed on the list in key sequence.
    - The newly created list entry is placed on the list specified by MOVETOLIST at the head or tail of the sublist composed of list entries whose secondary keys are equal to the secondary key of the moved entry. The newly created list entry is placed at the head or tail of this sublist as specified by SKEYTARGETDIR. If a sublist of entries with secondary keys equal to the assigned secondary key does not yet exist, the newly created list entry is placed on the list in secondary key sequence.
  - If the structure was allocated to use user provided ENTRYIDs, the newly created list entry is assigned the ENTRYID specified by ASSIGNENTRYID, provided a list entry does not already exist with the same ENTRYID.

- If the structure supports entry data and BUFFER and BUFLIST are not specified, the newly created list entry will not contain any entry data.
- If the structure supports adjunct data and ADJAREA is not specified, the newly created list entry will contain binary zeros for adjunct data. If the structure was allocated to use secondary keys, then the first 32 bytes of the adjunct data will be occupied by the secondary key which may or may not be zeros.

**DELETE**

The designated list entry is to be removed from the list on which it resides and returned to the pool of free entries for reuse.

When the request completes successfully, the list entry controls for the deleted entry, the remaining number of entries or elements residing on the list, and the remaining total number of allocated entries in the structure are returned in the answer area specified by ANSAREA.

**,RETcode=retcode**

Use this output parameter to specify a field to contain the return code. (The return code is also returned in register 15.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the return code when the request has completed.

**,RSNcode=rsncode**

Use this output parameter to specify a field to contain the reason code returned, if applicable. (The reason code is also returned in register 0.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the reason code (if any) when the request has completed.

**,SECONDARYKEY=NO\_SECONDARYKEY****,SECONDARYKEY=secondarykey**

Use this input parameter to specify an unsigned fixed 256-bit secondary key to be used to:

- Specify a secondary key to be used in conjunction with LISTNUM, DIRECTION, and SKEYREQTYPE to locate the list entry to be processed.
- Specify a secondary key to be assigned to the list entry if it is created. SKEYTARGETDIR is used to specify the position relative to secondary key ordering on the sublist at which to place the newly created list entry. If there exists a sublist of one or more entries with a matching secondary key on the list, the target position is at the head or tail of the sublist, as specified by SKEYTARGETDIR. If all existing list entries have a secondary key greater than that specified by SECONDARYKEY, the target position relative to secondary key ordering is at the head of the list. Similarly, if the specified SECONDARYKEY exceeds all of the secondary keys, the target position relative to secondary key ordering is at the tail of the list. If none of the list entries has a matching secondary key, and SECONDARYKEY is neither the greatest nor least among the secondary keys, the target position relative to secondary key ordering is determined according to secondary key sequence for the list.
- For structures allocated in a coupling facility of CFLEVEL=9 or higher, when the request specifies SKEYCOMPARE=YES, specifies the secondary key to be compared to the secondary key of the designated entry.

If the user does not supply SECONDARYKEY, the secondary key will be set to all binary zeros.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the 32-byte field that contains the secondary key to be assigned to the list entry.

**,SKEYCOMPARE=NO****,SKEYCOMPARE=YES**

Use this input parameter to specify whether secondary key comparison should be performed to determine whether the list entry should be processed.

**NO**

No secondary key comparison should be performed to determine whether the list entry should be processed.



**YES**

Secondary key comparison should be performed to determine whether the list entry should be processed. If the designated list entry exists but the requested key comparison criterion is not met, the IXLLSTE request is terminated.

SKEYCOMPARE=YES is only meaningful when the structure is allocated in a coupling facility of CFLEVEL=9 or higher.

SKEYCOMPARE=YES will be ignored if the target structure was not allocated with secondary keys.

**,SKEYPOSITION=UPDATE****,SKEYPOSITION=KEEP**

Use this input parameter to specify whether the list entry is moved from its current position on the secondary sublist when REQUEST=MOVE. This keyword has effect only if the list number specified by MOVETOLIST is the same as the list number on which the list entry currently resides.

The SKEYPOSITION keyword is only meaningful for list structures allocated in a coupling facility of CFLEVEL=9 or higher.

**UPDATE**

The list entry should be moved from its current position on the secondary sublist to a position on the secondary sublist as specified by SKEYTARGETDIR.

**KEEP**

The list entry should keep its current position based on secondary key ordering on the secondary sublist if and only if the list number specified by MOVETOLIST matches the current list number that contains the list entry.

**,SKEYREQTYPE=EQUAL****,SKEYREQTYPE=LESSOREQUAL****,SKEYREQTYPE=GREATEROREQUAL**

Use this input parameter to specify how secondary key comparison is to be performed on the designated keyed list entry or, when LOCATOR=KEYPOS is specified, to designate the list entry to be processed.

**EQUAL**

- The designated list entry must have a secondary key equal to the value specified for SECONDARYKEY. If the secondary key of the entry is not equal to the value specified for SECONDARYKEY, the IXLLSTE request is terminated.
- When LOCATOR=KEYPOS, specifies that if a list entry exists with a secondary key equal to the value specified for SECONDARYKEY, that entry is designated for processing. If more than one such entry exists, the list entry at either the head or tail of the sublist as specified by DIRECTION is designated for processing. If no such entry exists, the IXLLSTE request is terminated.

**LESSOREQUAL**

- The designated list entry must have a secondary key equal to or less than the value specified for SECONDARYKEY. If the secondary key of the entry is not equal to or less than the value specified for SECONDARYKEY, the IXLLSTE request is terminated.
- When LOCATOR=KEYPOS, specifies that:
  - If a list entry exists with a secondary key equal to or less than the value specified for SECONDARY key, that entry is designated for processing. If more than one such entry exists, the list entry at the head or tail of the sublist as specified by DIRECTION is designated for processing.
  - If no list entries exist with an equal secondary key, the list entry with the highest secondary key less than the value specified for SECONDARYKEY is designated for processing. If more than one such entry exists, the list entry at the head or tail of the sublist as specified by DIRECTION is designated for processing.
  - If no list entries exist with a secondary key equal to or less than the value specified for SECONDARYKEY, the IXLLSTE request is terminated.

**GREATEROREQUAL**

- The designated list entry must have a secondary key equal to or greater than the value specified for SECONDARYKEY. If the secondary key of the entry is not equal to or greater than the value specified for SECONDARYKEY, the IXLLSTE request is terminated.
- When LOCATOR=KEYPOS, specifies that:
  - If a list entry exists with a secondary key equal to the value specified for SECONDARYKEY, that entry is designated for processing. If more than one such entry exists, the list entry at the head or tail of the sublist as specified by DIRECTION is designated for processing.
  - If no list entries exist with an equal secondary key, the list entry with the lowest secondary key greater than the value specified for SECONDARYKEY is designated for processing. If more than one such entry exists, the list entry at the head or tail of the sublist as specified by DIRECTION is designated for processing.
  - If no list entries exist with a secondary key equal to or greater than the value specified for SECONDARYKEY, the IXLLSTE request is terminated.

**,SKEYTARGETDIR=HEADTOTAIL****,SKEYTARGETDIR=TAILTOHEAD**

Use this input parameter to specify the position relative to secondary key ordering on the sublist at which to place the newly created list entry. This parameter is used in conjunction with LISTNUM and with the assigned secondary key value when the structure was allocated with secondary keys.

SKEYTARGETDIR is only valid when the structure is allocated in a coupling facility of CFLEVEL=9 or higher.

**HEADTOTAIL**

The designated position is at the head of the sublist designated by the secondary key value.

**TAILTOHEAD**

The designated position is at the tail of the sublist designated by the secondary key value.

**,TARGETKEY=NO TARGETKEY****,TARGETKEY=targetkey**

Use this input parameter to specify an unsigned fixed 128-bit entry key to be assigned to the list entry if it is moved or created as a result of this request. The assigned entry key is used in conjunction with MOVETOLIST and MOVETODIRECTION to designate the target keyed position of the list entry.

This keyword is only applicable to a structure that was allocated to used keyed entries, and is otherwise ignored.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-byte area that contains the entry key to be assigned to the list entry.

**,UPDATECURSOR=NO****,UPDATECURSOR=YES**

Use this input parameter to specify whether the list cursor for the list containing the processed list entry should be updated if the request is successful.

If the structure was allocated with keyed entries, the entry key order is always used to determine the next or previous list entry. Note that the secondary key order is not relevant in determining the next or previous list entry.

**NO**

The cursor should not be updated.

Note that the list cursor will be set to binary zeros if it points to an entry that is deleted or moved to another list as a result of this request regardless of how UPDATECURSOR is specified.

**YES**

The cursor should be updated.

If the list entry specified on the request exists, the list cursor for the list on which the entry resides is updated before the list entry is processed for the request.

The CURSORUPDTYPE values of NEXTCOND, CURRENT, and CURRENTCOND are only meaningful for list structures allocated in a coupling facility of CFLEVEL=1 or higher.

**,VERSCOMP=NO\_VERSCOMP**

**,VERSCOMP=verscomp**

Use this input parameter to specify a version number to be compared to the version number of the existing entry. If this request creates a new entry, the VERSCOMP specification is ignored.

The existing entry is moved only if its version number meets the condition specified by the VERSCOMPTYPE parameter. If the entry does not have the specified version number, the request is terminated with no change to the structure.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-byte field that contains the version number.

**,VERSCOMPARE=NO**

**,VERSCOMPARE=YES**

Use this input parameter to specify whether version number comparison should be performed to determine whether the list entry should be processed.

**NO**

No version number comparison should be performed to determine whether the list entry should be processed.

**YES**

Version number comparison should be performed to determine whether the list entry should be processed.

**,VERSCOMPTYPE=EQUAL**

**,VERSCOMPTYPE=LESSOREQUAL**

Use this input parameter to specify how a list entry version comparison as specified by VERSCOMP is to be performed.

**Note:** The VERSCOMPTYPE parameter is valid only for list structures allocated in a coupling facility with CFLEVEL=1 or higher.

**VERSCOMPTYPE=EQUAL**

The version number for the list entry must be equal to the value specified for VERSCOMP.

**VERSCOMPTYPE=LESSOREQUAL**

The version number for the list entry must be less than or equal to the value specified for VERSCOMP.

**,VERSUPDATE=NONE**

**,VERSUPDATE=INC**

**,VERSUPDATE=DEC**

**,VERSUPDATE=SET**

Use this input parameter to specify how the entry version number of the moved entry will be updated, or for those cases where a new entry is created, initialized.

**NONE**

The existing entry's version number is not updated. The new entry's version number is set to binary zeros.

**INC**

The existing entry's version number is increased by one. The new entry's version number is set to binary zeros, except for the low-order bit, which is set to one.

**DEC**

The existing entry's version number is decreased by one. The new entry's version number is set to all binary ones.

**SET**

Both the existing and the new entry's version number is set to the value specified by NEWVERS.

## ABEND Codes

Abend X'026' (See *z/OS MVS System Codes* for more information on this abend.)

## Return and Reason Codes

When the system returns control to your program:

- GPR 15 (and RETCODE, if specified) contains a return code.
- If the return code is not zero, GPR 0 (and RSNCODE, if specified) contains a reason code.

**Note:** The return and reason codes will also be put into the answer area (mapped by IXLYLAA), if there is one, when the request completes.

The IXLYCON macro provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

<b>0</b>	IXLRETCODEOK
<b>4</b>	IXLRETCODEWARNING
<b>8</b>	IXLRETCODEPARMERROR
<b>C</b>	IXLRETCODEENVERROR
<b>10</b>	IXLRETCODECOMPERROR

The following table contains hexadecimal return and reason codes, the equate symbols associated with each reason code, and the meaning and suggested action for each return and reason code.

Table 81. Return and Reason Codes for IXLLSTE Macro		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0	None.	<p><b>Meaning:</b> If you specified a MODE value of ASYNCECB, ASYNCEXIT, ASYNCTOKEN, or ASYNCNORESPONSE, the request has been successfully initiated. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed. For any other MODE value, the request has successfully completed.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=ASYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=ASYNCEXIT, the connection's complete exit will be given control when the request has completed.</li> <li>• If you specified MODE=ASYNCTOKEN, use REQTOKEN when calling IXLFCOMP to determine when the request has completed.</li> </ul>

Table 81. Return and Reason Codes for IXLLSTE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	xxxx0402	<p><b>Equate Symbol:</b> IXLRSNCODEASYNCH</p> <p><b>Meaning:</b> The request could not be processed synchronously. It will be processed asynchronously. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed.</p> <ul style="list-style-type: none"> <li>• If MODE=SYNCECB was specified, the ECB specified for REQECB will be posted when the request has finished.</li> <li>• If MODE=SYNCEXIT was specified, the connection's complete exit will be given control when the request has finished.</li> <li>• If MODE=SYNCTOKEN was specified, an asynchronous request token was returned in the area specified by REQTOKEN. This request token may be specified on the IXLFCOMP macro to determine when the request has finished.</li> </ul> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=SYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=SYNCEXIT, the connection's complete exit will be given control when the request has completed.</li> <li>• If you specified MODE=SYNCTOKEN, use REQTOKEN when calling IXLFCOMP to determine when the request has completed.</li> </ul>
4	xxxx040D	<p><b>Equate Symbol:</b> IXLRSNCODEBADREADADJDATA</p> <p><b>Meaning:</b> Program error. The IXLLSTE READ, MOVE, or DELETE request specified that adjunct data was to be read, but the storage area specified by ADJAREA is not addressable. All other requested data was retrieved. If supported and requested, the entry data was retrieved. If this was a MOVE request, the entry was moved. If this was a DELETE request, the entry was deleted.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the ADJAREA address.</li> <li>• ADJAREA must be addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are calling IXLLSTE while disabled, ADJAREA must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLLSTE in AR-mode and you specified the ADJAREA address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are calling IXLLSTE in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLSTE macro.</li> <li>• If LAALCTL is not zeros, the request completed prematurely. See the action suggested for return code X'4' reason code X'xxxx0409'.</li> </ul> <p>Correct the address specified by ADJAREA, and rerun the request asking for adjunct data only.</p>

Table 81. Return and Reason Codes for IXLLSTE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	xxxx0410	<p><b>Equate Symbol:</b> IXLRSNCODELOCKCOND</p> <p><b>Meaning:</b> For a LOCKOPER=HELDDBY request, or a LOCKMODE=COND request, or a request that specified LOCKCOMP, the request could not be completed successfully because the specified lock is not currently held as required. The connection identifier of the lock owner is returned in the answer area (LAACONID field).</p> <p><b>Action:</b> Retry the request, or obtain the lock as required, and retry the request. If you are unable to get the lock, check the ID in the LAACONID field and determine if some recovery is necessary.</p>
4	xxxx0412	<p><b>Equate Symbol:</b> IXLRSNCODELOCKHELDDBYSYS</p> <p><b>Meaning:</b> For a LOCKOPER=HELDDBY or LOCKMODE=COND request, or a request that specified LOCKCOMP, the request could not be completed successfully because the specified lock is not currently held as required.</p> <p><b>Action:</b> Retry the request, or obtain the lock as required, and retry the request.</p>
8	xxxx0801	<p><b>Equate Symbol:</b> IXLRSNCODEBADPARMLIST</p> <p><b>Meaning:</b> Program error. The parameter list for this request is not addressable. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Verify that:</p> <ul style="list-style-type: none"> <li>• The parameter list address is uncorrupted.</li> <li>• The parameter list is addressable in the caller's primary address space.</li> <li>• If you are issuing this macro while disabled, the parameter list resides in either page-fixed or DREF storage.</li> <li>• If you are issuing this macro in AR-mode and you specified the parameter list address using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If you are issuing this macro in AR-mode, SYSSTATE ASCENV=AR must be issued before issuing this macro.</li> </ul>
8	xxxx0804	<p><b>Equate Symbol:</b> IXLRSNCODEBADVERSIONNUM (or IXLRSNCODEBADVERSION#)</p> <p><b>Meaning:</b> The version number in the macro parameter list is not compatible with the level of XES currently being used. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that your program did not overlay the parameter list storage.</li> <li>• Verify that your program was assembled with the correct macro library for the release of MVS on which your program is running.</li> </ul>

Table 81. Return and Reason Codes for IXLLSTE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx080A	<p><b>Equate Symbol:</b> IXLRSNCODEBADCONTOKEN</p> <p><b>Meaning:</b> Program error. The specified connect token (CONTOKEN) was not valid for one of the following reasons:</p> <ol style="list-style-type: none"> <li>1. The user with the connection identifier represented by the token has disconnected from the structure.</li> <li>2. The connector's task (the task that issued IXLCONN) ended.</li> <li>3. The specified token is not the token that was returned from IXLCONN.</li> <li>4. The request was issued from an address space other than the address space in which IXLCONN was issued.</li> <li>5. The connect token was invalidated during rebuild.</li> <li>6. The connect token was invalidated by XES.</li> </ol> <p><b>Note:</b> The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Take the action with the corresponding meaning.</p> <ol style="list-style-type: none"> <li>1. Discontinue use of the structure. Perform recovery and cleanup for the structure and check the protocol for the use of the structure to determine why you did not know of the disconnection.</li> <li>2. Discontinue use of the structure. Perform recovery and cleanup for the structure and check the protocol for the use of the structure to determine why you did not know that the task ended.</li> <li>3. The contoken is returned in the CONACONTOKEN field of the answer area specified on the IXLCONN request.</li> <li>4. Issue your request from the same address space the IXLCONN was issued in.</li> <li>5. Wait for the rebuild to complete, and try again.</li> <li>6. Discontinue use of the structure. Perform recovery and cleanup for the structure.</li> </ol>
8	xxxx0824	<p><b>Equate Symbol:</b> IXLRSNCODEWRONGSTRTYPE</p> <p><b>Meaning:</b> Program error. The connection specified by CONTOKEN is not to a list structure.</p> <p><b>Action:</b> Specify the appropriate CONTOKEN value, and rerun your program. The CONTOKEN is returned in the answer area provided on the IXLCONN invocation. The type of structure built is passed on the TYPE parameter of the IXLCONN macro.</p>

Table 81. Return and Reason Codes for IXLLSTE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0825	<p><b>Equate Symbol:</b> IXLRNOCODENOENTRY</p> <p><b>Meaning:</b> Program error. The designated list entry does not exist; therefore, no entries were processed.</p> <p><b>Action:</b> None necessary. However, if this return code and reason code are unexpected, you should check the way the list entry was created. Examine the parameters specified for locating the list entry on the invocation of this macro.</p>
8	xxxx0833	<p><b>Equate Symbol:</b> IXLRNOCODEBADPGBLATTR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list is specified as being pageable (PAGEABLE=YES), but is not.</p> <p><b>Action:</b> Change the buffer area(s) to pageable storage, or specify PAGEABLE=NO. See the PAGEABLE parameter description for specification instructions.</p>
8	xxxx0834	<p><b>Equate Symbol:</b> IXLRNOCODEBADNONPGBLATTR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list, is specified as being nonpageable (PAGEABLE=NO), but is either pageable or not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The type of storage in the buffer area corresponds to the value you specified on the PAGEABLE parameter. See the PAGEABLE parameter description for specification instructions.</li> <li>• The correct buffer address was used.</li> <li>• The buffer area was not previously freed.</li> <li>• If you are calling IXLLSTE while disabled, the buffers must reside in either page-fixed or DREF storage.</li> <li>• The buffer areas were allocated in a storage key that matches the key specified by the BUFSTGKEY parameter, or, if BUFSTGKEY is omitted, the caller's PSW key.</li> <li>• If BUFLIST was specified and your program is running in AR-mode: <ul style="list-style-type: none"> <li>– If the BUFFER or BUFLIST parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>– The BUFALET specification is correct.</li> <li>– You specified SYSSTATE ASCENV=AR before issuing the IXLLSTE macro.</li> </ul> </li> </ul>



Table 81. Return and Reason Codes for IXLLSTE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0835	<p><b>Equate Symbol:</b> IXLRNCODEBADDATAADDR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list, is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The correct buffer address was used.</li> <li>• The buffer area was not previously freed.</li> <li>• The buffer area was allocated in a storage key that matches the key specified using the BUFSTGKEY parameter, or, if BUFSTGKEY is omitted, the caller's PSW key.</li> <li>• If BUFLIST was specified and your program is running in AR mode: <ul style="list-style-type: none"> <li>– The BUFALET specification is correct.</li> <li>– You specified SYSSTATE ASCENV=AR before issuing the macro.</li> </ul> </li> </ul>
8	xxxx0836	<p><b>Equate Symbol:</b> IXLRNCODEBADREALADDR</p> <p><b>Meaning:</b> Program error. Real storage addresses were provided in the BUFLIST list, but one of the buffers is not addressable in central storage.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that BUFADDRTYPE was specified as you intended.</li> <li>• Ensure that the buffer addresses specified by BUFLIST are valid.</li> </ul>
8	xxxx0837	<p><b>Equate Symbol:</b> IXLRNCODEBADWRITEADJDATA</p> <p><b>Meaning:</b> Program error. The IXLLSTE WRITE or MOVE request specified that adjunct data was to be written, but the area specified by ADJAREA is not addressable. No entry data was written.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the ADJAREA address.</li> <li>• ADJAREA must be addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are calling IXLLSTE while disabled, ADJAREA must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLLSTE in AR-mode and you specified the ADJAREA address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are calling IXLLSTE in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLSTE macro.</li> </ul>

Table 81. Return and Reason Codes for IXLLSTE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0838	<p><b>Equate Symbol:</b> IXLRNCODEBADANSAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by ANSAREA is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The answer area address specified by ANSAREA is valid.</li> <li>• If the caller is running in AR-mode and the ANSAREA parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• The address specified for ANSAREA is addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are calling IXLLSTE while disabled, ANSAREA must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLLSTE in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLSTE macro.</li> </ul>
8	xxxx0839	<p><b>Equate Symbol:</b> IXLRNCODEBADREQTOKENAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by REQTOKEN is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The request token area specified by REQTOKEN is valid.</li> <li>• If the caller is running in AR-mode and the REQTOKEN parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If you are calling IXLLSTE while disabled, REQTOKEN must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLLSTE in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLSTE macro.</li> </ul>
8	xxxx083D	<p><b>Equate Symbol:</b> IXLRNCODEBADANSLEN</p> <p><b>Meaning:</b> Program error. The length of the answer area, as specified by ANSLEN, is not sufficient to contain answer area information. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Increase the size of the answer area, and rerun your program. The minimum size of the ANSAREA is in the LAA_LEN field of the IXLYLAA mapping macro.</p>

Table 81. Return and Reason Codes for IXLLSTE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx083E	<p><b>Equate Symbol:</b> IXLRNCODEMAXLISTKEY</p> <p><b>Meaning:</b> Program error. The list key to be assigned to an entry that was being created or moved was not valid. Either the list key or the list key plus the list key increment value was greater than the maximum list key.</p> <p><b>Action:</b> Ensure that you specified a correct list key increment. Depending on the protocol you are using for assigning list key values, you might want to issue a WRITE_LCONTROLS request to update either the list key value to a lower value or the maximum list key value to a higher value.</p>
8	xxxx083F	<p><b>Equate Symbol:</b> IXLRNCODEBADENTRYVERSION</p> <p><b>Meaning:</b> The entry designated by the specified list entry controls has a version number that does not meet the criteria specified by VERSCOMP and VERSCOMPTYPE. The list entry controls for the entry are returned in the answer area (field LAALCTL).</p> <p><b>Action:</b> None necessary. However, if this return code and reason code are unexpected, you might want to verify the values specified in VERSCOMP and VERSCOMPTYPE and look at the version returned in the answer area.</p>
8	xxxx0840	<p><b>Equate Symbol:</b> IXLRNCODEBADENTRYLIST</p> <p><b>Meaning:</b> The entry designated by the specified list entry controls does not reside on the list specified by LISTNUM. The list entry controls for the entry are returned in the answer area (field LAALCTL).</p> <p><b>Action:</b> None necessary. However, if you did not expect this return and reason code, you might want to verify what list this entry is on. Maybe the entry was moved, or you designated the wrong list in LISTNUM. Check the protocol for using this list.</p> <p>The information returned in the LAALCTL field of the answer area contains the number of the list that this list entry is on as well as other control information.</p>
8	xxxx0841	<p><b>Equate Symbol:</b> IXLRNCODEBADENTRYNAME</p> <p><b>Meaning:</b> Program error. The name specified by ENTRYNAME is not a unique name within the structure, and therefore entry creation is suppressed.</p> <p><b>Action:</b> Be sure to provide a unique entry name when creating entries to be written to structures that support entry names.</p>

Table 81. Return and Reason Codes for IXLLSTE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0842	<p><b>Equate Symbol:</b> IXLRSNCODEPERSISTENTLOCK</p> <p><b>Meaning:</b> Program error. The request specifying an unconditional SET or NOTHELD lock operation failed because the lock was held by a connection that is in the failed-persistent state. The connection identifier of the lock owner is returned in the answer area (field LAACONID).</p> <p><b>Action:</b> Either perform recovery for the connection, or wait until recovery for the connection is performed.</p>
8	xxxx0845	<p><b>Equate Symbol:</b> IXLRSNCODENONAMES</p> <p><b>Meaning:</b> Program error. A list entry was designated by entry name, but the structure does not support entry names.</p> <p><b>Action:</b> Ensure that you are connected to the intended structure. Ensure that the correct specification was made for REFOPTION on the IXLCONN macro. You may want to issue IXLMG to get more information about the structure.</p>
8	xxxx0846	<p><b>Equate Symbol:</b> IXLRSNCODEBADLOCKINDEX</p> <p><b>Meaning:</b> Program error. The specified LOCKINDEX exceeds the size of the lock table for the structure.</p> <p><b>Action:</b> Correct LOCKINDEX to specify an index that is contained within the lock table. The maximum value for the LOCKINDEX is one less than the value specified by the LOCKENTRIES parameter on the IXLCONN macro.</p>
8	xxxx0847	<p><b>Equate Symbol:</b> IXLRSNCODEBADLISTNUMBER</p> <p><b>Meaning:</b> Program error. The specified LISTNUM value exceeds the number of lists for the structure.</p> <p><b>Action:</b> Correct LISTNUM to specify a list number that exists in the structure. The number of lists allocated for a structure is determined on the LISTHEADERS parameter of the IXLCONN macro. The list numbers go from 0 to n-1, where n is the number of lists specified.</p>
8	xxxx0848	<p><b>Equate Symbol:</b> IXLRSNCODEBADRESET</p> <p><b>Meaning:</b> Program error. LOCKOPER=RESET was specified for a lock not currently held by the invoker. The value of the connection ID holding the lock is returned in the answer area (field LAACONID).</p> <p><b>Action:</b> Check your code to ensure that your connection did not already reset the lock.</p>

Table 81. Return and Reason Codes for IXLLSTE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx084A	<p><b>Equate Symbol:</b> IXLRSCODENOKEYS</p> <p><b>Meaning:</b> Program error. The structure does not support the use of entry keys. The IXLLSTE request type either required the structure to support entry keys or designated a sublist, list entry, or list position by list number and entry key.</p> <p><b>Action:</b> Ensure that you are connected to the intended structure. The use of keys for a structure is determined by the REFOPTION keyword on the IXLCONN macro.</p>
8	xxxx084B	<p><b>Equate Symbol:</b> IXLRSCODENOLOCKS</p> <p><b>Meaning:</b> Program error. A locking operation was requested, but the structure does not contain a lock table.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• You are connected to the intended structure.</li> <li>• You intended to perform a locking operation.</li> </ul> <p>The use of locks is determined by the LOCKENTRIES keyword on the IXLCONN macro.</p>
8	xxxx084E	<p><b>Equate Symbol:</b> IXLRSCODEBADMOVETOLIST</p> <p><b>Meaning:</b> Program error. The list number specified for MOVETOLIST exceeds the number of lists defined for the structure.</p> <p><b>Action:</b> Correct MOVETOLIST to specify a list that exists in the structure. The maximum number of lists in a structure is determined by the LISTHEADERS parameter on the IXLCONN macro.</p>
8	xxxx0851	<p><b>Equate Symbol:</b> IXLRSCODENOSUSPENDISABLE</p> <p><b>Meaning:</b> Program error. The request failed because MODE=SYNCSUSPEND was specified, but the caller is disabled and cannot be suspended.</p> <p><b>Action:</b> Either specify another MODE value, or become enabled (release the CPU lock), then reissue the request.</p>
8	xxxx0854	<p><b>Equate Symbol:</b> IXLRSCODEBADLOCKCOMP</p> <p><b>Meaning:</b> Program error. The connection identifier specified for LOCKCOMP is not valid.</p> <p><b>Action:</b> The connection identifier is returned in the answer area (mapped by IXLYCONA) when the IXLCONN macro was issued.</p>

Table 81. Return and Reason Codes for IXLLSTE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0859	<p><b>Equate Symbol:</b> IXLRSNCODEBADLISTAUTH</p> <p><b>Meaning:</b> The list authority value for the specified list does not meet the criteria specified by AUTHCOMP and AUTHCOMPTYPE. The current list authority (field LAALISTAUTH) and description (field LAALISTDESC) are returned in the answer area.</p> <p><b>Action:</b> None required; however you might want to take some action depending on your application. The list authority is set to binary zeros when the structure is allocated. When IXLLSTE is issued with the NEWAUTH keyword, the list authority is changed if the correct comparison list authority value is specified. Check the answer area to determine the list authority value for the list specified. Verify that the correct list number was specified.</p>
8	xxxx0864	<p><b>Equate Symbol:</b> IXLRSNCODEBADBUFSIZE</p> <p><b>Meaning:</b> Program error. The size of the BUFFER area or the buffer areas specified by BUFLIST is not large enough to contain the data for the first entry to be read in the list. No data is returned.</p> <p><b>Action:</b> If more space is available, specify a larger buffer size, or more buffers, and reissue the request. Check the information returned in the answer area (mapped by IXLYLAA) in the field LAALCTL (mapped by IXLYLCTL).</p>
8	xxxx0865	<p><b>Equate Symbol:</b> IXLRSNCODEBADBUFSPEC</p> <p><b>Meaning:</b> Program error. There is an error in the buffer specification.</p> <p><b>Action:</b> Check the following:</p> <ul style="list-style-type: none"> <li>• If BUFLIST was specified, check the requirements for BUFLIST, BUFNUM, and BUFINCRNUM.</li> <li>• If BUFFER was specified, check the requirements for BUFFER and BUFSIZE.</li> <li>• Buffer pointer(s) in BUFLIST</li> <li>• Buffer boundaries.</li> </ul>

Table 81. Return and Reason Codes for IXLLSTE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0866	<p><b>Equate Symbol:</b> IXLRNCODEBADBUFKEY</p> <p><b>Meaning:</b> Program error. The buffer storage key specified by BUFSTGKEY is incorrect. For requests that write coupling facility data, the data cannot be fetched from the specified buffer area. For requests that read coupling facility data, the data cannot be stored into the specified buffer area.</p> <p><b>Action:</b> Check the following:</p> <ul style="list-style-type: none"> <li>• Determine if the key of the storage being used for the buffers is different from the PSW key.</li> <li>• If BUFSTGKEY was specified, verify that the key was put in the correct bits (see the explanation of this parameter for more information).</li> <li>• If you are calling IXLLSTE in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLSTE macro.</li> </ul>
8	xxxx0867	<p><b>Equate Symbol:</b> IXLRNCODEBADBUFLIST</p> <p><b>Meaning:</b> Program error. The 128-byte storage area specified by BUFLIST is not addressable.</p> <p><b>Action:</b> Ensure that the address specified by BUFLIST is valid.</p>
8	xxxx086A	<p><b>Equate Symbol:</b> IXLRNCODEBADELEMNUM</p> <p><b>Meaning:</b> Program error. The value specified for ELEMNUM is not valid.</p> <p><b>Action:</b> Correct the ELEMNUM specification to be within the allowable range: from zero to whatever MAXELEMNUM value was returned to the connector in the connect answer area.</p>
8	xxxx0890	<p><b>Equate Symbol:</b> IXLRNCODEBADENTRYIDVALUE</p> <p><b>Meaning:</b> Program error. The specified user entry ID is zero.</p> <p><b>Action:</b> Correct the value of the user entry ID.</p>
8	xxxx0894	<p><b>Equate Symbol:</b> IXLRNCODEBADKEYCOMPARE</p> <p><b>Meaning:</b> Program error. The specified key comparison criteria was not satisfied.</p> <p><b>Action:</b> Ensure that the values specified for either ENTRYKEY or SECONDARYKEY are valid.</p>
8	xxxx0896	<p><b>Equate Symbol:</b> IXLRNCODEDUPLICATEENTRYID</p> <p><b>Meaning:</b> Program error. The specified user entry ID already exists in the specified structure.</p> <p><b>Action:</b> Ensure that the value specified for the entry ID is valid.</p>

Table 81. Return and Reason Codes for IXLLSTE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0897	<b>Equate Symbol:</b> IXLRSNCODEBADKEYTYPE <b>Meaning:</b> Program error. The specified KEYTYPE value is not valid for the specified structure. <b>Action:</b> Ensure that the value of KEYTYPE is valid.
8	xxxx0899	<b>Equate Symbol:</b> IXLRSNCODEBADSKYCOMPARE <b>Meaning:</b> Program error. The specified SKYCOMPARE value is not valid for the specified structure. <b>Action:</b> Correct the value of SKYCOMPARE.
8	xxxx089A	<b>Equate Symbol:</b> IXLRSNCODEBADSKYREQTYPE <b>Meaning:</b> Program error. The specified SKYREQTYPE value is not valid for the specified structure. <b>Action:</b> Correct the value of SKYREQTYPE.
8	xxxx089B	<b>Equate Symbol:</b> IXLRSNCODEBADKEYCOMPARETYPE <b>Meaning:</b> Program error. The specified KEYCOMPARE value is not valid for the specified structure. <b>Action:</b> Correct the value of KEYCOMPARE.
8	xxxx089C	<b>Equate Symbol:</b> IXLRSNCODEBADMOVETOKEY <b>Meaning:</b> Program error. The specified MOVETOKEY value is not valid for the specified structure. <b>Action:</b> Ensure that the value of MOVETOKEY is correct.
8	xxxx08AD	<b>Equate Symbol:</b> IXLRSNCODEBADHIGHSHAREDVIRT <b>Meaning:</b> Program error. The request specified a high shared virtual storage area (above 2GB). <b>Action:</b> None required.
C	xxxx0C06	<b>Equate Symbol:</b> IXLRSNCODENOCNN <b>Meaning:</b> Environmental error. This system does not have connectivity to the coupling facility that contains the list structure. Possible reasons for this are: <ul style="list-style-type: none"> <li>• The operator issued VARY PATH,OFFLINE.</li> <li>• The operator issued CONFIG CHP,OFFLINE.</li> <li>• Hardware errors to the coupling facility.</li> <li>• Facility or path failure to the coupling facility.</li> </ul> <b>Action:</b> Begin rebuilding the structure on a different coupling facility, or disconnect from the structure.



Table 81. Return and Reason Codes for IXLLSTE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C13	<p><b>Equate Symbol:</b> IXLRNCDEREQPURGED</p> <p><b>Meaning:</b> Environmental error. The request was purged prior to completion of the request. Possible reasons include:</p> <ul style="list-style-type: none"> <li>• The connector failed.</li> <li>• The connector disconnected.</li> <li>• The requestor failed.</li> <li>• The request was purged by IXLPURGE.</li> <li>• Requests were purged when the connector provided an IXLEERSP response for the Rebuild Stop or Rebuild Cleanup event.</li> <li>• The secondary address space was no longer valid.</li> </ul> <p><b>Action:</b> None if this is expected. Otherwise, determine why the connector failed.</p>
C	xxxx0C14	<p><b>Equate Symbol:</b> IXLRNCDERSTATUSUNKNOWN</p> <p><b>Meaning:</b> Environmental error. The request has completed, but the final disposition of the request cannot be determined. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the validity of your data by comparing the expected results with what is in the coupling facility.</li> <li>• Consider running rebuild processing to move the structure to attempt to protect it from further errors of this kind.</li> </ul>
C	xxxx0C17	<p><b>Equate Symbol:</b> IXLRNCDERSTRFULL</p> <p><b>Meaning:</b> Environmental error. The request attempted to create a new entry or overwrite an existing entry, but the structure is full and cannot accommodate any more entries.</p> <p><b>Action:</b> Determine why the structure is full. You should be monitoring the usage of the structure every time a read or write is done (LAATOTALCNT is the total count of allocated entries in the list structure, and LAATOTALELECNT is the total count of allocated elements in the list structure). This data should be evaluated periodically to ensure structure resources are being used efficiently. You might be able to delete existing entries to free up space, rebuild the structure to make it larger if rebuild is allowed (IXLCONN macro, ALLOWREBLD parameter), or alter the size of the structure (IXLALTER macro).</p>

Table 81. Return and Reason Codes for IXLLSTE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C18	<p><b>Equate Symbol:</b> IXLRSNCODELISTFULL</p> <p><b>Meaning:</b> Environmental error. The list designated as the target of a MOVE request, or as the target for entry creation on a WRITE or MOVE request, cannot accommodate more entries.</p> <p><b>Action:</b> If appropriate, change the maximum number of entries allowed on the list by specifying a new LISTLIMIT on a WRITE_LCONTROLS request. You should be monitoring the usage of the list structure every time a read or write is done. (LAATOTALCNT is the total count of allocated entries in the list structure, and LAATOTALELECNT is the total count of allocated elements in the list structure.) This data should be evaluated periodically to ensure structure resources are being used efficiently.</p>
C	xxxx0C25	<p><b>Equate Symbol:</b> IXLRSNCODESTRFAILURE</p> <p><b>Meaning:</b> Environmental error. The list structure failed prior to completion of the request.</p> <p><b>Action:</b> Either rebuild or disconnect from the structure.</p>
C	xxxx0C68	<p><b>Equate Symbol:</b> IXLRSNCODEBADREQCFLEVEL</p> <p><b>Meaning:</b> Environmental error. The request type is not permitted for the level of coupling facility in which the target structure is allocated.</p> <p><b>Action:</b> Either disconnect from the structure (using IXLDISC) or initiate a rebuild of the structure, if allowed (using IXLREBLD) in a coupling facility of the correct CFLEVEL.</p>
C	xxxx0CA0	<p><b>Equate Symbol:</b> IXLRSNCODEQUIESCEDSUSPENDFAIL</p> <p><b>Meaning:</b> Environmental error. The request is failed because the structure is quiesced for a system-managed process and SUSPEND=FAIL is specified on the IXLCONN.</p> <p><b>Action:</b> None, if this is expected.</p>
C	xxxxFFFF	<p><b>Equate Symbol:</b> IXLRSNCODENOTAVAILABLE</p> <p><b>Meaning:</b> Environmental error. XES functions are not available. This could occur because the coupling facility hardware necessary to provide XES functions is not present.</p> <p><b>Action:</b> Re-IPL the system, or follow your particular failure management protocol.</p>
10	xxxx10xx	<p><b>Meaning:</b> System error. XES processing failure. The state of the involved structure and the disposition of the request are unpredictable. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Contact the IBM support center.</p>

## Chapter 78. IXLLSTM – XES List Structure Multiple Entry Services

### Description

The IXLLSTM service enables you to perform operations to read, move, or delete multiple list entries. Some functions of IXLLSTM require that the list structure be allocated in a coupling facility of CFLEVEL=9 or higher. The following services are available:

- Delete multiple list entries that are designated in a list contained in the storage area specified by BUFFER or BUFLIST.
- Delete multiple list entries that meet a designated criteria from a list.
- Delete multiple list entries from a coupling facility list structure.
- Move multiple list entries that are designated in a list contained in the storage area specified by BUFFER or BUFLIST from the current location to a target location.
- Read multiple entries from a single list into the storage areas specified by BUFFER or BUFLIST and/or ADJAREA and ANSAREA.
- Read multiple entries from the list structure into the storage areas specified by BUFFER or BUFLIST and/or ADJAREA and ANSAREA.

The IXLLSTM macro provides list services equivalent to the following IXLLIST request types:

- IXLLIST REQUEST=DELETE\_ENTRYLIST
- IXLLIST REQUEST=DELETE\_MULT
- IXLLIST REQUEST=READ\_LIST
- IXLLIST REQUEST=READ\_MULT

See the descriptions of the corresponding IXLLIST macro for basic information about the services provided.

List structures allocated in a coupling facility of CFLEVEL=9 or higher can be allocated with certain attributes not available in a lower level coupling facility.

- Users can assign a user-designated entry identifier to a newly created list entry, rather than having the system assign the entry identifier.
- Secondary keys can be specified, thus allowing the user to reference a keyed list entry by entry key, secondary key, or both.

The IXLLSTM macro can be used to exploit these new attributes.

See *z/OS MVS Programming: Sysplex Services Guide* for information about using the IXLLSTM macro.

### Environment

The following are the environment requirements for the caller.

Environment	Environment requirement
Minimum authorization:	Supervisor state and PSW key 0-7
Dispatchable unit mode:	Task or SRB
Cross memory mode:	Any PASN, any HASN, any SASN. The current primary address space must be the same as the primary address space at the time the connection service (IXLCONN) was issued for the structure.

Environment	Environment requirement
AMODE:	31-bit
ASC mode:	Primary or access register (AR)
Interrupt status:	Enabled or disabled for I/O and external interrupts
Locks:	Disabled callers must be legally disabled by holding the CPU lock and cannot hold other disabled locks. Enabled callers must not hold any locks. When MODE=SYNCSUSPEND is specified, the caller must be enabled.
Control parameters:	See “Restrictions” on page 1476

## Programming Requirements

- If your program is in AR mode, issue the SYSSTATE ASCENV=AR macro before IXLLSTM. SYSSTATE ASCENV=AR tells the system to generate code appropriate for AR mode.
- Include the IXLYCON mapping macro in your program. This macro provides a list of equate symbols for users of XES services and exits.
- Include mapping macros IXLYLAA, IXLYLCTL, IXLYDELI, and IXLYMELI in your program as necessary. Table 82 on page 1476 lists these macros, the information and areas they map, and the particular IXLLIST requests and parameters they apply to.

Table 82. Mapping Macros for IXLLSTM			
Mapping Macro	Information	Area	Request/Parameter
IXLYLAA	Answer area output	ANSAREA area	All requests
IXLYLCTL	List entry controls	Field LAALCTL of IXLYLAA	READ_MULT
		Field LAARLMLCTLS of IXLYLAA, BUFLIST buffers, BUFFER area	READ_LIST & READ_MULT (when TYPE=ECONTROLS)
IXLYDELI	Delete Entry List input	BUFLIST buffers, BUFFER area	DELETE_ENTRYLIST
IXLYMELI	Move Entry List input	BUFLIST buffers, BUFFER area	MOVE_ENTRYLIST

## Restrictions

- If you specify BUFLIST and PAGEABLE=YES, all of the buffers in the list must be in the same area of storage; you cannot mix common and private storage addresses for the buffers in the list.
- This service cannot be invoked by callers running as a disabled interrupt exit (DIE).
- The caller's parameter list must be addressable in the caller's primary address space.
- If the caller is running in AR ASC mode and specifies a parameter using explicit register notation, the access register corresponding to the general register must appropriately qualify the general register.
- The virtual storage areas specified by the ADJAREA and ANSAREA parameters must be addressable in the caller's primary address space or from the caller's PASN access list.
- The virtual storage areas specified by parameters other than ADJAREA and ANSAREA can be addressable in the caller's primary, secondary, or home address spaces, from the PASN access list, or from the dispatchable unit access list (DU-AL).
- If the caller is disabled then the parameter list and all storage areas addressed by the macro parameters must reside in either nonpageable or disabled reference storage.

## Input Register Information

---

Before issuing the IXLLSTM macro, the caller does not have to place any information into any register unless using it in register notation for a particular parameter, or using it as a base register.

## Output Register Information

---

When control returns to the caller, the GPRs contain:

### Register Contents

- 0**  
Reason code
- 1**  
Used as work register by the system
- 2-13**  
Unchanged
- 14**  
Used as work register by the system
- 15**  
Return code

When control returns to the caller, the ARs contain:

### Register Contents

- 0-1**  
Used as work registers by the system
- 2-13**  
Unchanged
- 14-15**  
Used as work registers by the system

## Performance Implications

---

See *z/OS MVS Programming: Sysplex Services Guide* for performance information.

## Understanding IXLLSTM Version Support

---

The IXLLSTM macro supports versions 0, 1, 3, and 4 — the version number corresponds to the level of the IXLLIST or IXLLSTM macro in which a keyword or function was introduced. The higher the version number, the more recent the version of the macro. Some IXLLSTM keywords are supported for multiple versions, but have different functions for each version.

- Keywords not specifically noted here are supported by all versions starting with version 0 and higher of the IXLLSTM macro.
- The following IXLLSTM keywords are supported by all versions starting with version 1 and higher of the IXLLSTM macro.
  - KEYCOMPARE
  - LISTKEYINC
- The following keyword is supported by all versions starting with version 3 and higher of the IXLLSTM macro.
  - EXTRESTOKEN

- The following keywords and functions are supported by all versions starting with version 4 and higher of the IXLLSTM macro.
  - LISTKEYAREA
  - MISCOMPARE
  - MOVETOSKEY
  - SECONDARYKEY
  - SKEYCOMPARE
  - SKEYRANGEEND
  - SKEYREQTYPE

Specify the version of the parameter list that you want generated with the PLISTVER keyword. See Chapter 2, “Specifying a Macro Version Number,” on page 5 for considerations when specifying the version of the parameter list with PLISTVER.

## Summary of Version-Dependent Parameter Functions

The following table summarizes the allowed use of some of the specifications that can be made depending on the PLISTVER value specified.

<i>Table 83. IXLLSTM Version Support</i>		
Keyword	Version	Notes
AUTHCOMPARE	0	NO may be specified.
	1	NO or YES may be specified.
REQUEST	0	READ_LIST, READ_MULT, DELETE_MULT, or DELETE_ENTRYLIST may be specified.
	1	Same as version 0
	2	Same as version 1
	3	Same as version 2
	4	READ_LIST, READ_MULT, DELETE_MULT, DELETE_ENTRYLIST, DELETE_LIST, or MOVE_ENTRYLIST may be specified.
VERSCOMPARE	0	NO or YES may be specified.
	1	Same as version 0
	2	Same as version 1
	3	Same as version 2
	4	NO, YES, or BYENTRY may be specified.
VERSCOMPTYPE	0	EQUAL may be specified.
	1	EQUAL or LESSOREQUAL may be specified.
KEYSCANTYPE	0	ENTRY may be specified.
	1	Same as version 0.
	2	Same as version 1.
	3	Same as version 2.
	4	ENTRY or SECONDARY may be specified.

Table 83. IXLLSTM Version Support (continued)

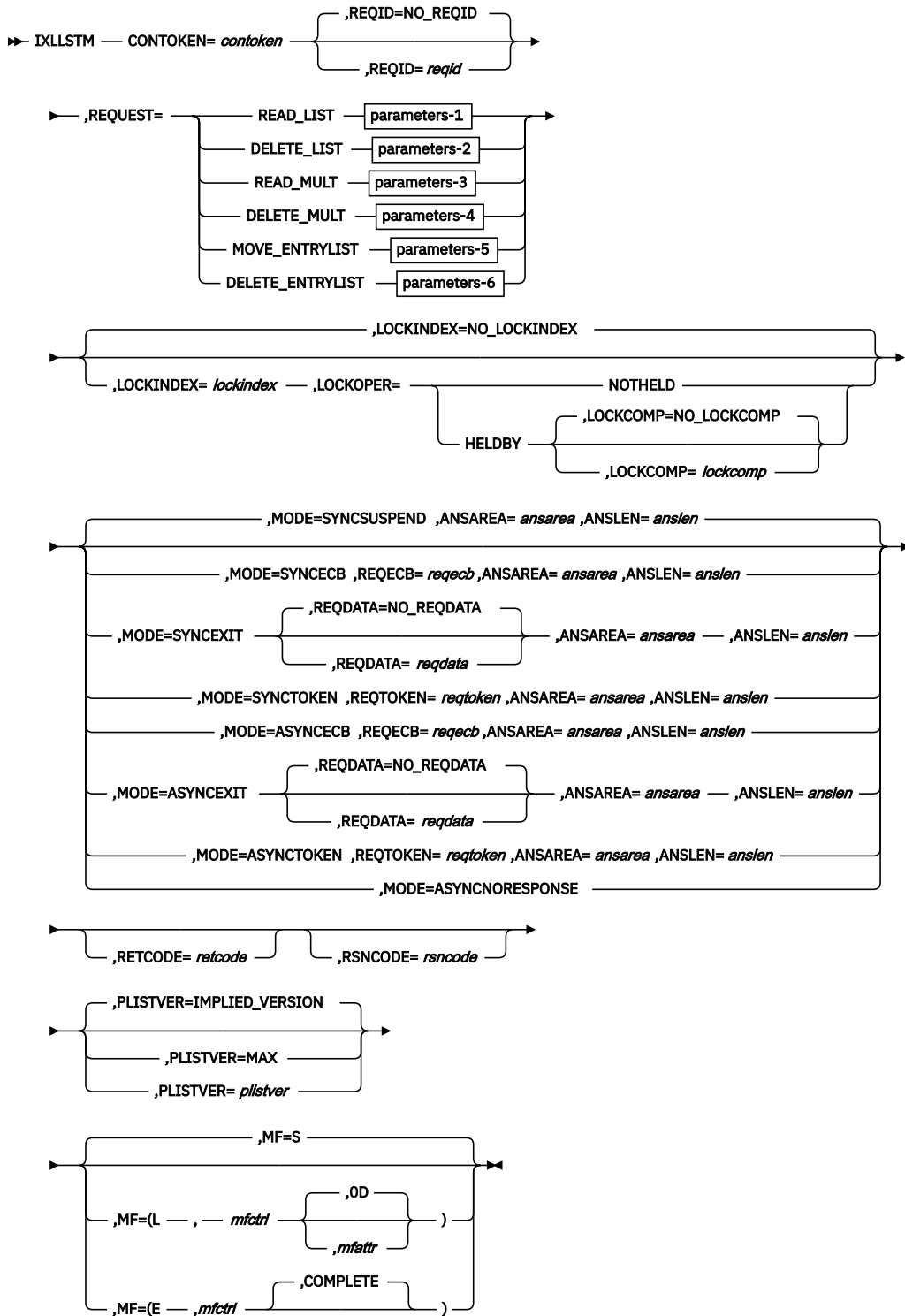
Keyword	Version	Notes
KEYREQTYPE	0	EQUAL may be specified. GREATEROREQUAL and LESSOREQUAL may be specified for only REQUEST=READ_LIST.
	1	Same as version 0
	2	Same as version 1
	3	Same as version 2
	4	EQUAL, LESSOREQUAL, GREATEROREQUAL, or RANGE may be specified.

## Syntax Diagram

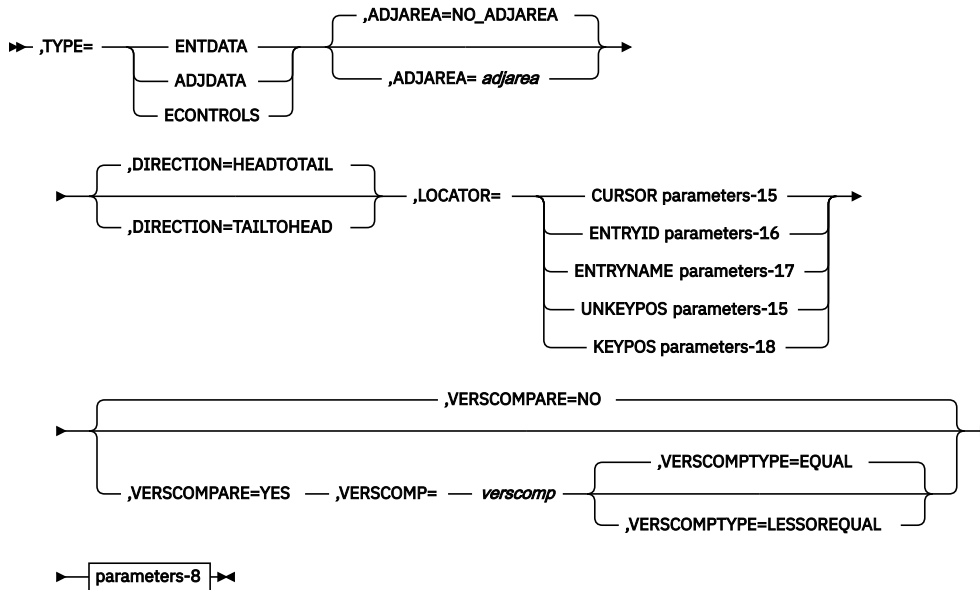
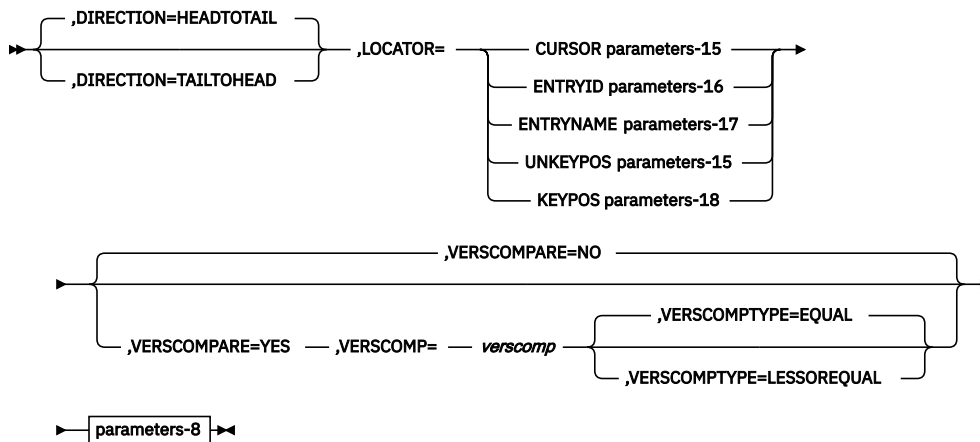
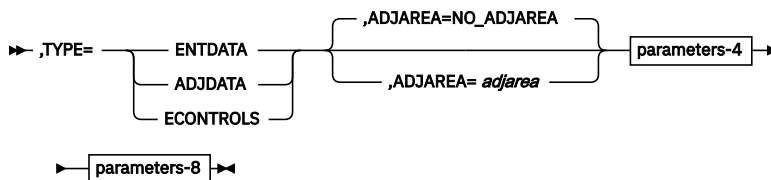
---

The syntax diagram for IXLLSTM is as follows:

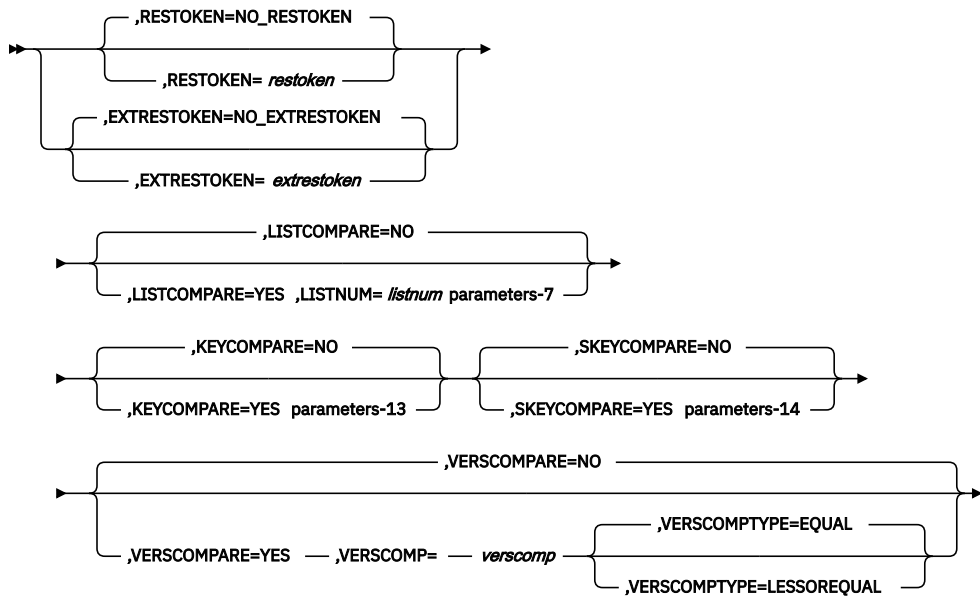
## main diagram



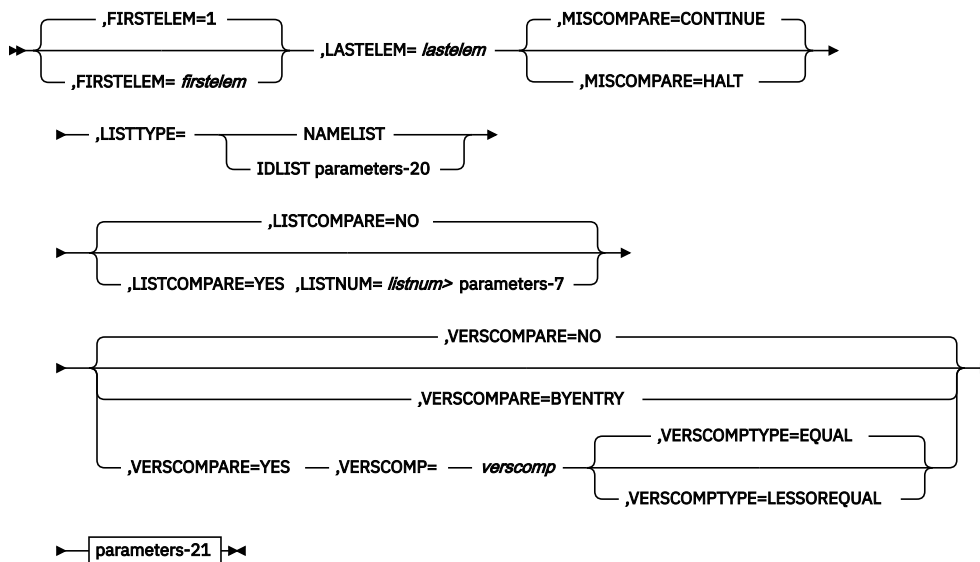


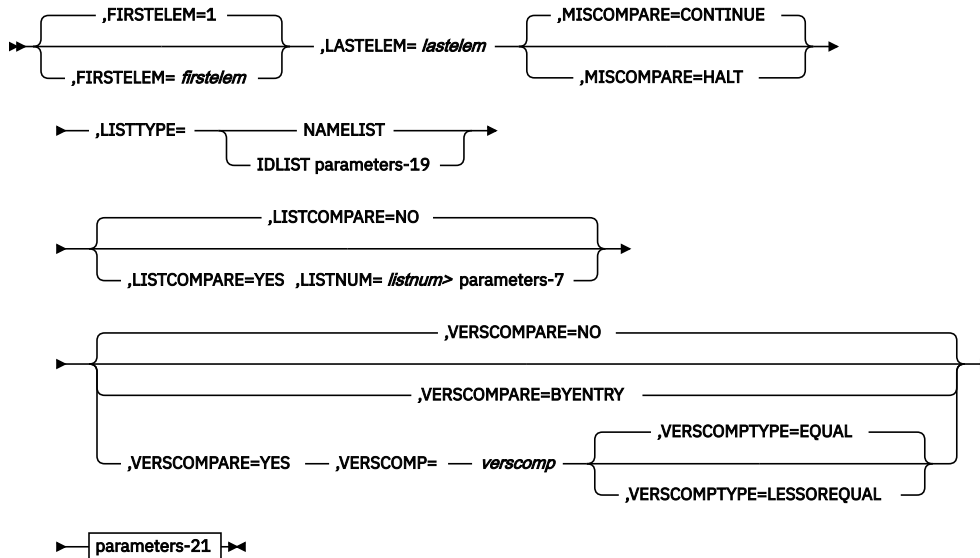
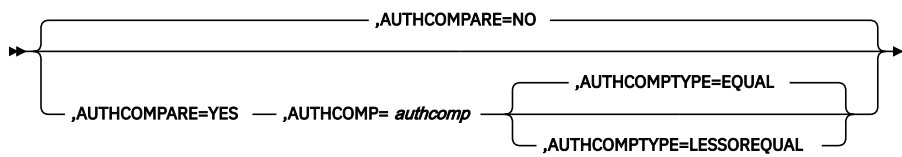
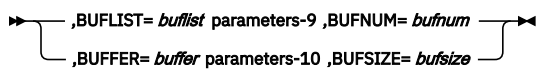
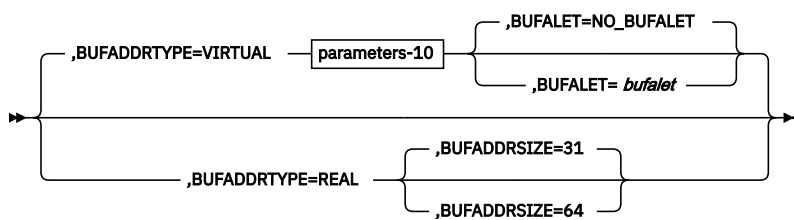
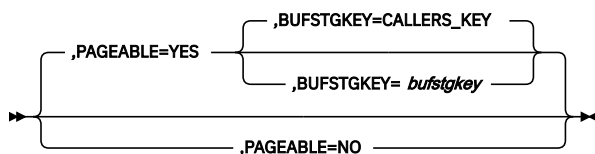
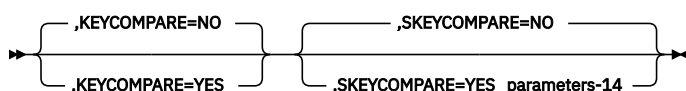
**parameters-1****parameters-2****parameters-3**

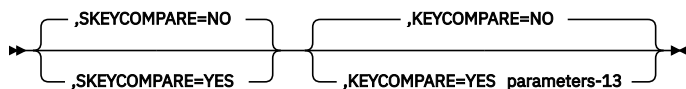
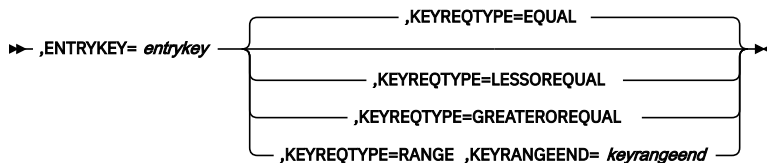
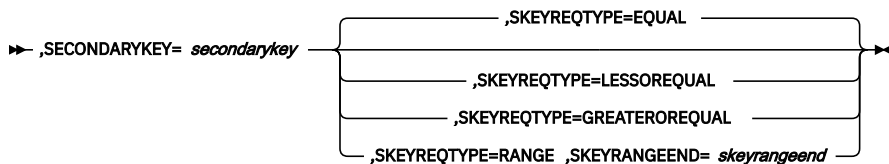
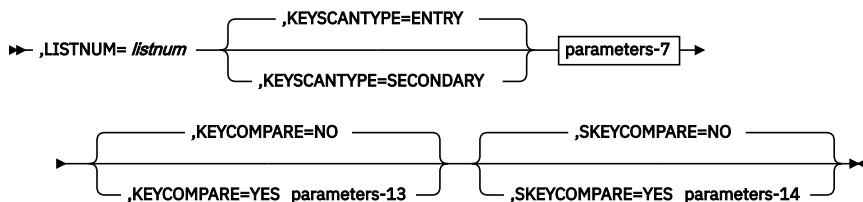
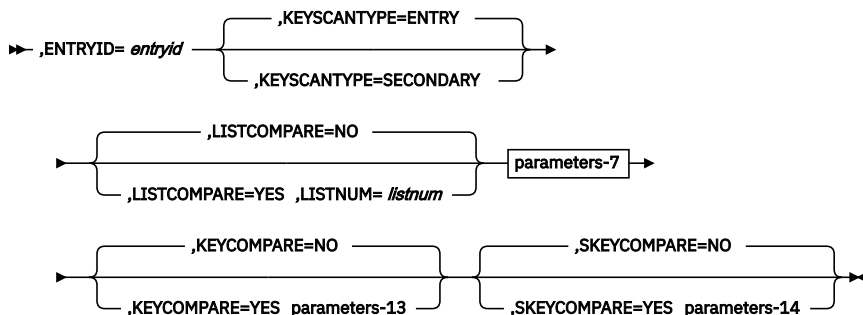
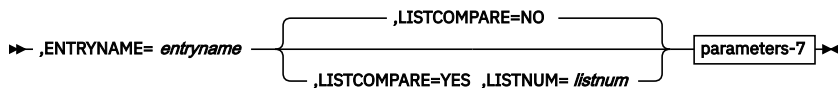
## parameters-4

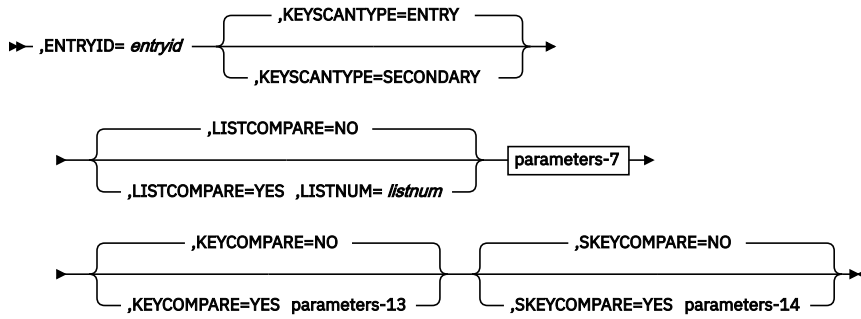
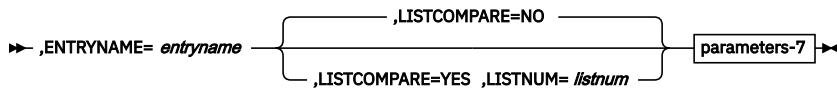
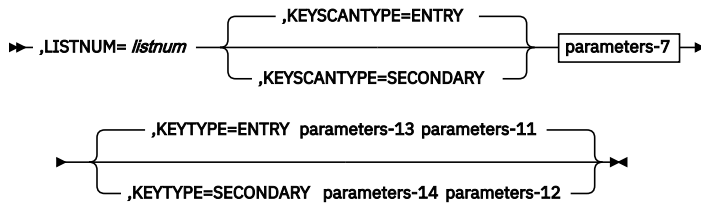
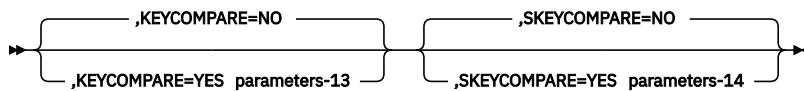
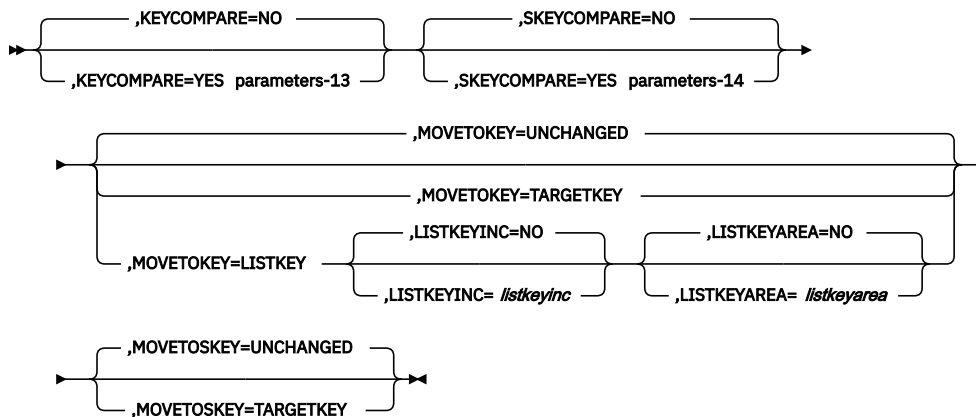


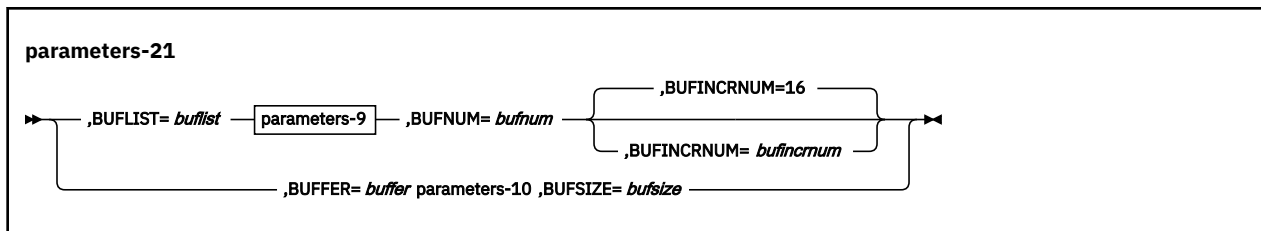
## parameters-5



**parameters-6****parameters-7****parameters-8****parameters-9****parameters-10****parameters-11**

**parameters-12****parameters-13****parameters-14****parameters-15****parameters-16****parameters-17**

**parameters-16****parameters-17****parameters-18****parameters-19****parameters-20**



## Parameter Descriptions

The parameter descriptions for IXLLSTM are listed in alphabetical order. Default values are underlined:

**,ADJAREA=NO\_ADJAREA**

**,ADJAREA=adjarea**

Use this input parameter to specify a storage area to contain the adjunct data that is read from or written to an entry.

Specify ADJAREA only for structures that support adjunct data. (Adjunct areas for a structure are established through the IXLCONN macro.)

If the structure was allocated to use secondary keys, the first 32 bytes of ADJDATA will contain the secondary key of the entry.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 64-byte area that contains or will contain the adjunct data.

**,ANSAREA=ansarea**

Use this input parameter to specify an answer area to contain information returned from the request. The format of the answer area is described by mapping macro IXLYLAA.

Not all fields in the answer area are applicable to all request types. Request type descriptions indicate which answer area fields are applicable for successful request completion cases. Return and reason code description indicates which answer area fields are applicable for non-successful completing requests.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an area (with a length of ANSLEN) that will contain the information returned by the request.

**,ANSLEN=anslen**

Use this input parameter to specify the size of the storage area specified by ANSAREA.

Check the prologue of the IXLYLAA mapping macro for the minimum required size of the answer area, or use the length field (LAA\_LEN) in the LAA.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 2-byte field that contains the size, in bytes, of the answer area.

**,AUTHCOMP=NO\_AUTHCOMP**

**,AUTHCOMP=authcomp**

Use this input parameter to specify a value to be compared to the list authority value of the list specified by LISTNUM. You must supply the LISTNUM parameter.

If the comparison does not meet the condition specified by the AUTHCOMPTYPE parameter (EQUAL or LESSOREQUAL), the request fails.

**Note:** The AUTHCOMP parameter is valid only for list structures allocated in a coupling facility with CFLEVEL=1 or higher.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the 16-byte field that contains the list authority value.

**,AUTHCOMPARE=NO**

**,AUTHCOMPARE=YES**

Use this input parameter to specify whether the list authority comparison is to be used to determine if entries on the list should be processed.

**NO**

No list authority comparison is to be performed before processing any of the list entries.

**YES**

List authority comparison should precede processing of any list entries.

**,AUTHCOMPTYPE=EQUAL****,AUTHCOMPTYPE=LESSOREQUAL**

Use this input parameter specify how the list audit comparison is to be performed.

**EQUAL**

The list authority for the list specified by LISTNUM must be equal to the value specified for AUTHCOMP.

**LESSOREQUAL**

The list authority for the list specified by LISTNUM must be less than or equal to the value specified for AUTHCOMP.

**Note:** The AUTHCOMPTYPE parameter is valid only for list structures allocated in a coupling facility with CFLEVEL=1 or higher.

**,BUFADDRSIZE=31****,BUFADDRSIZE=64**

Use this input parameter to specify whether a 31-bit or a 64-bit address is specified by a BUFLIST entry.

**31**

The entry in BUFLIST is 31 bits in size.

**64**

The entry in BUFLIST is 64 bits in size.

**,BUFADDRTYPE=VIRTUAL****,BUFADDRTYPE=REAL**

Use this input parameter to specify whether the buffer addresses specified in the BUFLIST list are virtual storage or real storage addresses.

**VIRTUAL**

The buffer addresses are virtual storage addresses. The virtual storage can be pageable or nonpageable. See the PAGEABLE parameter for information about managing storage binds when specifying virtual storage addresses.

**REAL**

The buffer addresses are real storage addresses.

It is the caller's responsibility to manage the binds between the data buffer virtual storage and the real storage addresses provided. The caller must ensure that the data buffer virtual storage remains bound to the real storage addresses provided until the request completes.

**,BUFALET=NO\_BUFALET****,BUFALET=*bufalet***

Use this input parameter to specify an access list entry token (ALET) to be used in referencing all of the buffers specified by BUFLIST.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 4-byte field that contains the ALET.

**,BUFFER=*buffer***

Use this parameter to hold data for the request. The BUFSIZE keyword specifies the size of the buffer. For READ\_LIST and READ\_MULT requests, BUFFER is an output parameter. For MOVE\_ENTRYLIST and DELETE\_ENTRYLIST requests, BUFFER is an input parameter. For all IXLLSTM request types, the length of the buffer must be a multiple of 4096 bytes between 4096 and 65536 and the buffer must start on a 4096-byte (page) boundary.

- Upon successful completion of a READ\_LIST or READ\_MULT request, the BUFFER area is used for output and contains, starting at offset zero, an array of elements. One array element is returned for each processed entry. The number of elements returned in the BUFFER area is indicated in

the answer area. The length of an array element can be determined by its make-up: the structure element size, the number of elements in the entry, an adjunct data size (64 bytes), and the length of list entry controls. (The length and contents of list entry controls is defined by mapping macro IXLYLCTL.)

For READ\_LIST and READ\_MULT requests, each array element is constructed as follows, dependent on the request options specified:

- When adjunct data is requested, the adjunct data for the first entry processed is returned in the storage area specified by ADJAREA. The adjunct data for all other entries processed is returned in the BUFFER area.
- When list entry controls are requested, the entry controls for the first entry processed are returned in the answer area specified by ANSAREA. The entry controls for all other entries processed are returned in the BUFFER area.

The format of each array element in the BUFFER, therefore, is as follows:

- When TYPE=ENTDATA is specified, entry data for each list entry processed is contained in the buffer.
- When TYPE=ADJDATA is specified, adjunct data for each list entry processed after the first entry is contained in the buffer. (The adjunct data for the first entry processed is returned in ADJAREA.)
- When TYPE=ECONTROLS is specified, list entry controls for each list entry processed after the first entry is contained in the buffer. (The list entry controls for the first entry is returned in ANSAREA.)
- When TYPE=(ENTDATA,ADJDATA) is specified, entry data for the first list entry processed is contained in the buffer, followed by the entry data and then the adjunct data for each additional list entry processed.
- When TYPE=(ENTDATA,ECONTROLS) is specified, entry data for the first list entry processed is contained in the buffer, followed by the list entry controls and then the entry data for each additional list entry processed.
- When TYPE=(ADJDATA,ECONTROLS) is specified, list entry controls followed by adjunct data for each list entry processed after the first is contained in the buffer.
- When TYPE=(ENTDATA,ADJDATA,ECONTROLS) is specified, entry data for the first list entry processed is contained in the buffer, followed by list entry controls, entry data, and adjunct data (in that order) for each additional list entry processed.
- For MOVE\_ENTRYLIST requests, BUFFER is used for input and should be formatted into 32-byte, 64-byte, or 96-byte elements, where each element is mapped by IXLYMELI and contains the information required to move a list entry. The format and size of an element is determined by the options specified on the MOVE\_ENTRYLIST request.
  - A 32-byte element is required for any one of the following conditions:
    - The structure does not support keyed entries and VERSCOMPARE=NO is specified.
    - The structure does not support keyed entries and VERSCOMPARE=YES is specified.
    - The structure does support keyed entries and MOVETOKEY=UNCHANGED, MOVETOSKEY=UNCHANGED, and VERSCOMPARE=NO are specified.
    - The structure does support keyed entries and MOVETOKEY=UNCHANGED, MOVETOSKEY=UNCHANGED, and VERSCOMPARE=YES are specified.
    - The structure does support keyed entries and MOVETOKEY=LISTKEY, MOVETOSKEY=UNCHANGED, and VERSCOMPARE=NO are specified.
    - The structure does support keyed entries and MOVETOKEY=LISTKEY, MOVETOSKEY=UNCHANGED, and VERSCOMPARE=YES are specified.
  - A 64-byte element is required for the following conditions:
    - VERSCOMPARE=BYENTRY or MOVETOKEY=TARGETKEY is specified.
  - A 96-byte element is required for the following condition:



- MOVETOSKEY=TARGETKEY is specified.
- For DELETE\_ENTRYLIST requests, the BUFFER area is used as input and should be formatted into 12-byte, 16-byte, or 64-byte elements, where each element is mapped by the IXLYDELI macro and contains all the information required to delete a list entry. The format and size of an element is determined by the options specified on the DELETE\_ENTRYLIST request.
  - A 12-byte element is required for any one of the following conditions:
    - VERSCOMPARE=NO and LISTTYPE=IDLIST are specified.
    - VERSCOMPARE=YES and LISTTYPE=IDLIST are specified.
  - A 16-byte element is required for any one of the following conditions:
    - VERSCOMPARE=NO and LISTTYPE=NAMELIST are specified.
    - VERSCOMPARE=YES and LISTTYPE=NAMELIST are specified.
  - A 64-byte element is required when VERSCOMPARE=BYENTRY is specified.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an area (with a length of BUFSIZE) that contains the entry data.

#### **,BUFLIST=buflist**

Use this output or input parameter to specify a list of buffer addresses to hold data for the request. The set of buffers is used as if it were a single contiguous area.

The format of the list is a set of eight-byte elements. The first four (high-order) bytes of each element are reserved. The second four (low-order) bytes of each element contain the address of a buffer.

There may be from 1 to 16 buffers passed in the list. Each buffer in the list must be the same size and must reside in the same address space or data space. Data is fetched from or stored into the buffers in the order specified.

One of BUFFER or BUFLIST is required for all READ\_LIST, READ\_MULT, MOVE\_ENTRYLIST, and DELETE\_ENTRYLIST requests. See the description of the BUFFER keyword for the format of the data contained in the buffers.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 128-byte area that contains the list of buffer addresses.

#### **,BUFNUM=bufnum**

Use this input parameter to specify the number of buffers in the BUFLIST list. Valid BUFNUM values are from 1 to 16.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 1-byte field that contains the number of buffers in the buffer list.

#### **,BUFSIZE=bufsize**

Use this input parameter to specify the size of the BUFFER area. See the BUFFER parameter description for valid buffer sizes.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a fullword field that contains the size of the buffer (BUFFER) in bytes.

#### **,BUFSTGKEY=CALLERS\_KEY**

#### **,BUFSTGKEY=bufstgkey**

Use this input parameter to specify a storage key that you define and use when referencing the buffers specified by BUFLIST or the buffer that is specified by BUFFER.

If you do not specify BUFSTGKEY, or if you specify BUFSTGKEY=CALLERS\_KEY, all references to one or more buffers are performed by using the caller's PSW key.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of an 8-bit field that contains the storage key in the format B'kkkkxxxx', where kkkk is the key and xxxx is ignored.

**CONTOKEN=contoken**

Use this input parameter to specify the connect token that was returned by the IXLCONN service. The connect token uniquely identifies your connection to the list structure, and must be specified on each IXLLSTM invocation.

The connect token is available in the IXLCONN answer area mapped by IXLYCONA.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-byte field that contains the connect token.

**,DIRECTION=HEADTOTAIL****,DIRECTION=TAILTOHEAD**

Use this input parameter to specify the direction of processing for traversing the given list.

**HEADTOTAIL**

Processing should begin at the designated entry and proceed toward the tail of the list.

**TAILTOHEAD**

Processing should begin at the designated entry and proceed toward the head of the list.

**,ENTRYID=entryid**

Use this input parameter to specify the list entry identifier of the entry to be used as the starting point of the READ\_LIST or DELETE\_LIST request.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 12-byte field that contains the entry identifier.

**,ENTRYKEY=entrykey**

Use this input parameter to either:

- Specify the entry key to be used to compare to the entry key of the list entry to determine if the list entry should be processed. For MOVE\_ENTRYLIST and DELETE\_ENTRYLIST requests, if the condition specified by KEYREQTYPE is not met for the current list entry, then no processing is performed for the current entry and processing either continues with the next entry to be considered or is terminated based on the value specified for MISCOMPARE. For all other IXLLSTM requests, if the condition specified by KEYREQTYPE is not met for the current list entry, then no processing is performed for the current entry and processing continues with the next entry to be considered.
- For READ\_LIST and DELETE\_LIST requests, specify the entry key to be used to partially indicate the starting list entry for the request. If DIRECTION=HEADTOTAIL was specified, the designated starting list entry is the head of the sublist. If DIRECTION=TAILTOHEAD was specified, the designated starting list entry is the tail of the sublist.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-byte field that contains the entry key.

**,ENTRYNAME=entryname**

Use this input parameter to specify the list entry name of the entry to be used as the starting point of the READ\_LIST or DELETE\_LIST request.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-byte field that contains the entry name.

**,EXTRESTOKEN=NO\_EXTRESTOKEN****,EXTRESTOKEN=extrestoken**

Use this input parameter to specify a name for an extended restart token specifying an appropriate coupling facility indicator for resuming requests that complete prematurely.

An extended restart token is returned in the LAARESTOKEN answer area specified by ANSAREA when the request terminates prematurely. The extended restart token may be specified on a subsequent READ\_MULT or DELETE\_MULT request to resume the request at an appropriate point.

The RESTOKEN and EXTRESTOKEN keywords are mutually exclusive. Requestors who specify IXLCONN ALLOWAUTO = YES must use the 16-byte extended restart token (EXTRESTOKEN).

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16 byte field that contains the extended restart token.

**Note:** Specifying an extended restart token of all zeros causes the request to consider all entries as unprocessed. Specifying an extended restart token other than one returned from a previous invocation of the request and not fully initialized to all zeros will produce unpredictable request results.

**,FIRSTELEM=1**

**,FIRSTELEM=firstelem**

Use this input parameter to specify the index of the first array element to be processed for MOVE\_ENTRYLIST and DELETE\_ENTRYLIST requests.

The value must reference one of the array elements in the buffers specified by BUFLIST or the buffer specified by BUFFER.

An index value of 1 references the first array element in the BUFFER area or the first BUFLIST buffer.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a halfword field that contains the index of the first array element to be processed.

**,KEYCOMPARE=NO**

**,KEYCOMPARE=YES**

Use this input parameter to specify whether the key value of an existing keyed list entry is to be compared to determine if this entry should be selected for processing.

**NO**

Specify this option if no entry key comparison will be performed to determine if this entry should be processed.

**YES**

Specify this option if entry key comparison is to be performed based on the KEYREQTYPE specified to determine if this entry is selectable for processing.

**Note:**

1. KEYCOMPARE is only meaningful for list structures allocated on CFLEVEL=1 or higher.
2. KEYCOMPARE=YES is ignored if the target structure does not support keyed entries.

**,KEYRANGEEND=keyrangeend**

Use this input parameter to specify the ending value for the range of keys to be compared to the entry key of the designated list entry.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16 character field that contains the key range ending value.

**Note:** KEYRANGEEND is a required keyword when KEYREQTYPE=RANGE is specified.

**,KEYREQTYPE=EQUAL**

**,KEYREQTYPE=LESSOREQUAL**

**,KEYREQTYPE=GREATEROREQUAL**

**,KEYREQTYPE=RANGE**

Use this input parameter to specify how an existing keyed list is located and how entry key comparison is to be performed to determine if the list entry is selectable for processing.

**EQUAL**

The entry must have a key that equals the ENTRYKEY key.

**LESSOREQUAL**

The entry must have a key that is less than or equal to the ENTRYKEY key.

**GREATEROREQUAL**

The entry must have a key that is greater than or equal to the ENTRYKEY key.

**RANGE**

The entry must have a key within the specified range of values. The ENTRYKEY specified will be used as the beginning of the range of values. KEYRANGEEND will be used as the ending value. A

list entry must have an entry key value within the specified entry key range, inclusive, for it to be selectable.

For a READ\_LIST or DELETE\_LIST request when LOCATOR=KEYPOS:

1. When no entries on the list meet the requirements of KEYREQTYPE, the request will be failed with a return code X'8' and reason code IXLRSCODENOENTRY.
2. When LESSOREQUAL or GREATEROREQUAL are specified, if no entries on the list have an entry key value equal to the specified value, but entries exist with an entry key value greater than (if GREATEROREQUAL was specified), or less than (if LESSOREQUAL was specified) the entry key value specified, then the entry with an entry key value closest to the value specified will be selected for the starting list entry. When multiple entries have the same entry key value, DIRECTION is used to resolve whether the first or last entry with the entry key value is selected for the starting list entry.
3. When LESSOREQUAL, GREATEROREQUAL or RANGE is specified, if multiple entries have an entry key value within the specified range, DIRECTION will be used to resolve whether the first or last entry within the entry key range is selected for the starting list entry.
4. When KEYREQTYPE=RANGE is specified, KEYRANGEEND=YES is required.

**,KEYSCANTYPE=ENTRY**

**,KEYSCANTYPE=SECONDARY**

Use this input parameter to specify which key ordering (entry key ordering or secondary key ordering) will be used to scan for entries on the list.

**ENTRY**

Entry key ordering will be used for scanning the list.

**SECONDARY**

Secondary key ordering will be used for scanning the list.

**Note:** KEYSCANTYPE=SECONDARY is only valid when the structure is allocated in a coupling facility with CFLEVEL=9 or higher.

**,KEYTYPE=ENTRY**

**,KEYTYPE=SECONDARY**

Use this input parameter to specify whether to locate the starting list entry using the entry key or the secondary key.

**ENTRY**

The entry key will be used to locate the starting list entry.

**SECONDARY**

The secondary key will be used to locate the starting list entry.

**Note:** KEYTYPE=SECONDARY is only valid when the structure is allocated in a coupling facility with CFLEVEL=9 or higher.

**,LASTELEM=*lastelem***

Use this input parameter to specify the index of the last array element to be processed for MOVE\_ENTRYLIST or DELETE\_ENTRYLIST requests.

The specified value must be greater than or equal to the specified FIRSTELEM value, and must specify one of the array elements passed in the BUFFER area or the BUFLIST buffers.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a halfword field that contains the index of the last array element to be processed.

**,LISTCOMPARE=NO**

**,LISTCOMPARE=YES**

Use this input parameter to specify whether the list number comparison should be used to determine if list entries should be processed.

**LISTCOMPARE=NO**

List number comparison should not precede processing of list entries.

**LISTCOMPARE=YES**

List number comparison should precede processing of list entries.

**,LISTKEYAREA=NO****,LISTKEYAREA=*listkeyarea***

Use this input parameter to specify the address of a virtual storage area in which entry key values will be placed when the current list key value has been assigned to list entries.

List key information will be placed in the LISTKEYAREA when the current list key value has been assigned to the list entry and any of the following conditions exists:

- The request completes successfully.
- The model-dependent timeout has been exceeded.
- A list entry does not exist.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 112-area field that contains an array of entry keys.

**,LISTKEYINC=NO****,LISTKEYINC=*listkeyinc***

Use this input parameter to specify a value to be added to the list key after the entry key is set to the list key value.

If the result of adding the value specified by LISTKEYINC to the target list key value is greater than the maximum list key value, the system fails the request.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the fullword field that contains the value to be added to the list key after the entry key is set to the list key value.

**,LISTNUM=NO LISTNUM****,LISTNUM=*listnum***

Use this input parameter to specify:

- The number of the list on which the starting list entry resides.
- The number of the list to be compared to the number of the list on which the entries to be processed reside.

If the list comparison fails, then the IXLLSTM operation is terminated and the list entry controls along with the appropriate return and reason codes are provided.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 4-byte field that contains the number of the list.

**,LISTTYPE=NAMELIST****,LISTTYPE=IDLIST**

Use this input parameter to specify whether the first field in each array element in the BUFFER area or BUFLIST buffers for MOVE\_ENTRYLIST or DELETE\_ENTRYLIST requests contains an entry name or an EntryID.

**LISTTYPE=NAMELIST**

The first field in each array element in the BUFFER area or BUFLIST buffers contains the entry name of the list entry.

**LISTTYPE=IDLIST**

The first field in each array element in the BUFFER area or BUFLIST buffers contains the EntryID of the list entry. The EntryID may be either system generated or user provided.

**Note:** LISTTYPE=NAMELIST is not allowed for structures that do not support named entries.

**,LOCATOR=CURSOR**  
**,LOCATOR=ENTRYID**  
**,LOCATOR=ENTRYNAME**  
**,LOCATOR=UNKEYPOS**  
**,LOCATOR=KEYPOS**

Use this input parameter to specify how to locate the first list entry for READ\_LIST or DELETE\_LIST requests to be processed.

**CURSOR**

The list cursor is to be used to designate the starting list entry for the request.

**ENTRYID**

The EntryID should be used to designate the starting list entry for the request. EntryIDs may be either assigned or provided by the user. User provided EntryIDs must be specified if ENTRYTYPE=USER is specified on the IXLCONN request.

**ENTRYNAME**

The entry name should be used to designate the starting list entry for the request. ENTRYNAME may only be specified for structures that support named entries.

**UNKEYPOS**

LISTNUM and DIRECTION will be used to designate the starting entry for the request.

**KEYPOS**

LISTNUM, DIRECTION and the key specified by KEYTYPE will be used to designate the starting entry for the request.

**,LOCKCOMP=NO\_LOCKCOMP**

**,LOCKCOMP=lockcomp**

Use this input parameter to specify a connection identifier to be verified as the current lock owner as a prerequisite to successful completion of this request.

When LOCKCOMP is specified the locking operation is always considered to be a conditional operation. That is, if the request experiences lock contention the request will be ended with no resultant change to the structure, and appropriate return and reason codes are provided in the ANSAREA. The connection identifier is available from the IXLCONN answer area.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 1-byte field that contains the connection identifier.

**,LOCKINDEX=NO\_LOCKINDEX**

**,LOCKINDEX=lockindex**

Use this input parameter to specify the index of the lock to be operated on within the lock table for the list structure.

When specified, the designated lock will be operated on as specified by the LOCKOPER keyword. The specified value must fall within the range 0 to the number of lock table entries minus one, inclusive.

LOCKINDEX is mutually exclusive with MODE=ASYNCRESPONSE.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword field that contains the lock index.

**,LOCKOPER=NOTHELD**

**,LOCKOPER=HELD BY**

Use this input parameter to specify the type of operation to be performed on the specified lock.

**NOTHELD**

The state of the lock must be such that the lock is not held for the duration of the requested list entry operation. The lock state remains unchanged as a result of this option.

When NOTHELD is specified, the locking operation is always considered to be a conditional operation. That is, if the specified lock is held then the entire IXLLSTM operation will be ended with no resultant change to the structure, and appropriate return and reason codes are provided in the ANSAREA.

**HELDDBY**

When LOCKCOMP is not specified, the list operation is to be performed only if the lock is currently held by the connection specified by CONTOKEN.

When LOCKCOMP is specified, the list operation is to be performed only if the lock is currently held by the connection specified by LOCKCOMP.

**,MF=S**

**,MF=(L,*mfctrl*)**

**,MF=(L,*mfctrl*,*mfattr*)**

**,MF=(L,*mfctrl*,0D)**

**,MF=(M,*mfctrl*)**

**,MF=(M,*mfctrl*,COMPLETE)**

**,MF=(M,*mfctrl*,NOCHECK)**

**,MF=(E,*mfctrl*)**

**,MF=(E,*mfctrl*,COMPLETE)**

**,MF=(E,*mfctrl*,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

**,*mfctrl***

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

**,*mfattr***

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

**,COMPLETE**

**,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

**COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if SMILE=*var* were an optional parameter and the default is SMILE=NO\_SMILE then it would not be documented. However, if the default was SMILE=:-), then it would be documented because a value would be the default.

**NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

**,MISCOMPARE=CONTINUE****,MISCOMPARE=HALT**

For MOVE\_ENTRYLIST and DELETE\_ENTRYLIST requests, use this input parameter to specify whether processing should continue to the next entry or halt when any of the version number, list number or key comparisons are not successful.

**CONTINUE**

Processing should continue to the next entry if any of the version number, list number or key comparisons, when specified, are not successful.

**HALT**

If a version number, list number or key comparison is specified but is not successful, processing for the command is stopped. List entry controls and the appropriate return and reason codes will be returned in the ANSAREA.

**,MODE=SYNCSUSPEND****,MODE=SYNCECB****,MODE=SYNCEXIT****,MODE=SYNCTOKEN****,MODE=ASYNCECB****,MODE=ASYNCEXIT****,MODE=ASTOEN****,MODE=ASYNCSUSPEND**

Use this input parameter to specify whether the request is to be performed synchronously or asynchronously.

**SYNCSUSPEND**

The request is performed synchronously. Control is not returned to the caller until request processing is complete and the final disposition determined.

If necessary the caller will be suspended until the request completes. To use this option, your program must be enabled for I/O and external interrupts.

**SYNCECB**

The request will be attempted synchronously. If the request cannot be completed synchronously, control is returned to the caller prior to the completion of the request, and the ECB specified by REQECB is posted when the request has completed.

When MODE=SYNCECB is specified and the request does not complete synchronously, latent XES binds to the storage locations specified by BUFFER, BUFLIST, LISTKEYAREA, ADJAREA, and ANSAREA persist until the REQECB ECB is posted.

**SYNCEXIT**

The request will be attempted synchronously. If the request cannot be completed synchronously, control is returned to the caller prior to completion of the request. When the request completes, the connection's Complete exit will be called.

When the request does not complete synchronously, latent XES binds to the storage locations specified by BUFFER, BUFLIST, ADJAREA, LISTKEYAREA, and ANSAREA persist until the connection's Complete exit is called.

**SYNCTOKEN**

The request will be attempted synchronously. If the request cannot be completed synchronously, control is returned to the caller prior to completion of the request and a token that uniquely identifies the request is returned.

When the request does not complete synchronously, latent XES binds to the storage locations specified by BUFFER, BUFLIST, ADJAREA, LISTKEYAREA, and ANSAREA persist until a subsequent corresponding IXLFComp request indicates completion of the original request.

**ASYNCECB**

The request is to be initiated and control is to be returned to the caller prior to completion of the request. When the request completes, the ECB specified by REQECB will be posted.



The latent XES binds to the storage locations specified by BUFFER, BUFLIST, LISTKEYAREA, ADJAREA, and ANSAREA persist until the REQECB ECB is posted.

#### **ASYNCEXIT**

The request is to be initiated and control is to be returned to the caller prior to completion of the request. When the request completes, the connection's Complete exit will be called.

The latent XES binds to the storage locations specified by BUFFER, BUFLIST, LISTKEYAREA, ADJAREA, and ANSAREA persist until the connection's Complete exit is called.

#### **ASYNCTOKEN**

The request is to be initiated, a token generated that uniquely identifies the request on this system, and control returned to the caller prior to completion of the requested operation.

The latent XES binds to the storage locations specified by BUFFER, BUFLIST, LISTKEYAREA, ADJAREA, and ANSAREA persist until a subsequent corresponding IXLFComp request indicates completion of the original request.

#### **ASYNCSNORESPONSE**

The request is to be initiated and control returned to the caller prior to completion of the requested operation. No asynchronous request token is returned; hence no external mechanism exists to force completion of the request.

MODE=ASYNCSNORESPONSE is mutually exclusive with LOCKINDEX, BUFFER, and BUFLIST. Any request that does not perform a locking operation and does not use a BUFFER area or BUFLIST buffers may specify MODE=ASYNCSNORESPONSE.

#### **,MOVETOKEY=UNCHANGED**

#### **,MOVETOKEY=TARGETKEY**

#### **,MOVETOKEY=LISTKEY**

Use this input parameter to specify on a MOVE\_ENTRYLIST request how the key is to be assigned to the list entry when it is moved to the MOVETOLIST. MOVETOKEY may only be specified for structures that support keyed entries.

#### **UNCHANGED**

The current entry key value assigned to the list entry will remain unchanged.

#### **TARGETKEY**

The TARGET\_KEY provided in the array element in the BUFFER or the BUFLIST buffer will be assigned to the list entry when it is moved to the target list.

#### **LISTKEY**

The current list key value will be assigned to the list entry when it is moved to the target list.

#### **,MOVETOSKEY=UNCHANGED**

#### **,MOVETOSKEY=TARGETKEY**

Use this input parameter on MOVE\_ENTRYLIST requests to specify how the secondary key is to be assigned to the list entry when it is moved to the MOVELIST.

#### **UNCHANGED**

The current secondary key entry value assigned to the list entry will remain unchanged.

#### **TARGETKEY**

The TARGET\_SKEY provided in the array element in the BUFFER or the BUFLIST buffers will be assigned to the list entry when it is moved to the target list.

**Note:** MOVETOSKEY can only be specified for structures that support secondary keyed entries.

#### **,PAGEABLE=YES**

#### **,PAGEABLE=NO**

Use this input parameter to identify whether the storage areas specified by BUFFER or BUFLIST reside in pageable storage.

#### **YES**

Specify this option to indicate that the BUFFER or BUFLIST buffers reside in pageable virtual storage.

This includes disabled reference (DREF) storage, and may include storage that has the potential to become pageable during the processing of a request. (An example is address space storage owned by any swappable address space, for which a PGSER FIX has been successfully processed, but for which the owning address space gets swapped during processing of a cache or list request.) It does not include implicitly non-pageable storage (such as is obtained from non-pageable subpools).

The system takes responsibility for managing binds to central storage for the duration of the list request, regardless of what address space owns the storage or whether the storage-owning address space is swappable or nonswappable. The storage can be owned by any address space.

## NO

Specify this option to indicate that the BUFFER or BUFLIST buffers reside in non-pageable virtual storage.

This includes implicitly non-pageable storage areas. If the virtual storage may potentially become pageable, the invoker is responsible for ensuring the virtual storage remains non-pageable for the duration of the request. (An example is address space storage owned by any swappable address space, for which a PGSER FIX has been successfully processed, but for which the owning address space gets swapped-out during processing of a list request.)

If MODE=ASYNCTOKEN is specified or MODE=SYNCTOKEN is specified and the request does not complete synchronously, the storage must remain non-pageable until completion of the corresponding IXLFCOMP request. If MODE=ASYNCEXIT is specified or MODE=SYNCEXIT is specified and the request does not complete synchronously, the storage must remain non-pageable until the complete exit is driven for the request. If MODE=ASYNCECB is specified or MODE=SYNCECB is specified and the request does not complete synchronously, the storage must remain non-pageable until the specified ECB is posted for the request.

The system takes responsibility for managing binds to central storage for the duration of the list request, if and only if the non-pageable storage is owned by either the requester's address space or the connector's address space. If the storage is owned by any other address space, then the invoker is responsible for ensuring that the virtual storage remains non-pageable for the duration of the request (including the case in which the storage is owned by a swappable address space that is swapped during processing of a list request). Subject to this consideration, the storage can be owned by any address space. See *z/OS MVS Programming: Sysplex Services Guide*.

## **,PLISTVER=IMPLIED\_VERSION**

## **,PLISTVER=MAX**

## **,PLISTVER=plistver**

Use this input parameter to specify the version of the macro. See “Understanding IXLLIST Version Support” on page 965 for a description of the options available with the PLISTVER macro.

## **,REQDATA=NO\_REQDATA**

## **,REQDATA=reqdata**

Use this input parameter with MODE=SYNCEXIT or MODE=ASYNCEXIT to pass any data you choose to the complete exit. The exit will get control only if the request is processed asynchronously.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the data to be passed to the complete exit.

## **,REQECB=reqecb**

Use this output parameter with either MODE=SYNCECB or MODE=ASYNCECB to specify the address of an ECB, which is to be posted when the request completes if the request was processed asynchronously.

Before coding REQECB, you must ensure that:

- You initialize the ECB before you issue the request.
- The ECB resides in either common storage or the home address space where IXLCONN was issued.
- Any tasks that wait for the ECB to be posted reside in the home address space where IXLCONN was issued.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that contains the address of the ECB to be posted when the request completes. The ECB must be aligned on a fullword boundary.

**,REQID=NO\_REQID**

**,REQID=*reqid***

Use this input parameter to specify a user-defined request identifier to be associated with the request. You can specify this request identifier on the IXPURGE macro to cancel a request that has not yet been processed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of an 8-byte field that contains the user-defined request identifier.

**,REQTOKEN=*reqtoken***

Use this output parameter with either MODE=SYNCTOKEN or MODE=ASYNCTOKEN to specify the address of a storage area to receive the request token that is returned when the request will be processed asynchronously. This token, which uniquely identifies the request, must be used as input to the IXLFCOMP macro, which you use to determine if the request has completed.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 16-byte field where the system will put the request token.

**,REQUEST=READ\_LIST**

**,REQUEST=DELETE\_LIST**

**,REQUEST=READ\_MULT**

**,REQUEST=DELETE\_MULT**

**,REQUEST=MOVE\_ENTRYLIST**

**,REQUEST=DELETE\_ENTRYLIST**

Use this input parameter to specify the type of operation to be performed on the structure.

**READ\_LIST**

Use READ\_LIST to request that a list scan process be performed such that entries meeting a specified set of criteria will be read. The entry data, adjunct data, list entry controls, or any combination of these for the selected entries on the list might be read into the buffer storage area specified for the request (designated by BUFFER or BUFLIST).

**DELETE\_LIST**

Use DELETE\_LIST to request that a list scan process be performed such that entries meeting a specified set of criteria will be removed from the list on which they reside and returned to the pool of free entries for reuse.

**READ\_MULT**

Use READ\_MULT to request that the entry data, the associated adjunct data, or the list entry controls for all allocated entries that meet the criteria specified be read into the storage area specified by BUFFER or the buffers specified by BUFLIST.

**DELETE\_MULT**

Use DELETE\_MULT to request that all entries that meet the specified criteria be removed from whatever list they reside on and returned to the pool of free entries for reuse.

**MOVE\_ENTRYLIST**

Use MOVE\_ENTRYLIST to request that all specified entries be moved from the current source location to a designated target location. The entries to be moved are found in the list of formatted elements in the storage area specified by BUFFER or the buffers specified by BUFLIST.

**DELETE\_ENTRYLIST**

DELETE\_ENTRYLIST is used to request that all specified entries be removed from whichever list they reside on and returned to the pool of free entries for reuse. The entries to be deleted are found in the list of formatted elements in the storage area specified by BUFFER or the buffers specified by BUFLIST.

**,RESTOKEN=NO\_RESTOKEN**

**,RESTOKEN=*restoken***

Use this input parameter to specify a name for a restart token specifying an appropriate coupling facility indicator for resuming requests the complete prematurely.

A restart token is returned in the LAARESTOKEN answer area specified by ANSAREA when the request terminates prematurely. The restart token may be specified on a subsequent READ\_MULT or DELETE\_MULT request to resume the request at an appropriate point.

The RESTOKEN and EXTRESTOKEN keywords are mutually exclusive. Requestors who specify IXLCONN ALLOWAUTO = YES must use the 16-byte extended restart token (EXTRESTOKEN).

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8 character field that contains the restart token.

**Note:** Specifying a restart token of all zeros causes the request to consider all entries as unprocessed. Specifying a restart token other than one returned from a previous invocation of the request and not fully initialized to all zeros will produce unpredictable request results.

**,RETCODE=retcode**

Use this output parameter to specify a field to contain the return code. (The return code is also returned in register 15.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the return code when the request has completed.

**,RSNCODE=rsncode**

Use this output parameter to specify a field to contain the reason code returned, if applicable. (The reason code is also returned in register 0.)

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a 4-byte field that will contain the reason code (if any) when the request has completed.

**,SECONDARYKEY=xsecondarykey**

Use this input parameter to either:

- Specify the secondary key value to be compared to the secondary key of the list entry to determine if the list entry should be processed. For MOVE\_ENTRYLIST and DELETE\_ENTRYLIST requests, if the condition specified by SKEYREQTYPE is not met for the current list entry, then no processing is performed for the current entry and processing either continues with the next entry to be considered or is terminated based on the value specified for MISCOMPARE. For all other request types, if the condition specified by SKEYREQTYPE is not met for the current list entry, then no processing is performed for the current entry and processing continues with the next entry to be considered.
- Specify the secondary key to be used to partially indicate the starting list entry for the request for READ\_LIST and DELETE\_LIST requests. If DIRECTION=HEADTOTAIL was specified, the designated starting list entry is the head of the sublist. If DIRECTION=TAILTOHEAD was specified, the designated starting list entry is the tail of the sublist.

For a READ\_LIST or DELETE\_LIST request when LOCATOR=KEYPOS:

1. When no entries on the list meet the requirements of SKEYREQTYPE, the request will be failed with a return code X'8' and reason code IXLRSCODENOENTRY.
2. When LESSOREQUAL or GREATEROREQUAL are specified, if no entries on the list have an entry key value equal to the specified value, but entries exist with an entry key value greater than (if GREATEROREQUAL was specified), or less than (if LESSOREQUAL was specified) the entry key value specified, then the entry with an entry key value closest to the value specified will be selected for the starting list entry. When multiple entries have the same entry key value, DIRECTION is used to resolve whether the first or last entry with the entry key value is selected for the starting list entry.
3. When LESSOREQUAL, GREATEROREQUAL or RANGE is specified, if multiple entries have an entry key value within the specified range, DIRECTION will be used to resolve whether the first or last entry within the entry key range is selected for the starting list entry.
4. When KEYREQTYPE=RANGE is specified, KEYRANGEEND=YES is required.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 32 character field that contains the secondary key.

**Note:** SECONDARYKEY is required if SKEYCOMPARE=YES.

**,SKEYCOMPARE=NO**

**,SKEYCOMPARE=YES**

Use this input parameter to specify whether the secondary key value of an existing keyed list entry should be compared to the SKEYREQTYPE to determine if this entry should be selected for processing.

**NO**

Specify this value if no secondary key comparison will be performed to determine if this entry should be processed.

**YES**

Specify this option if secondary key comparison is to be performed based on the SKEYREQTYPE to determine if this entry is to be selected for processing.

**Note:** SKEYCOMPARE=YES is ignored if the target structure was not allocated with secondary keys.

**,SKEYRANGEEND=*keyrangeend***

Use this input parameter to specify the ending value for the range of keys to be compared to the secondary key of the designated list entry.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the 32 character field that contains the secondary key range ending value.

**Note:** SKEYRANGEEND must be specified when SKEYREQTYPE=RANGE.

**,SKEYREQTYPE=EQUAL**

**,SKEYREQTYPE=LESSOREQUAL**

**,SKEYREQTYPE=GREATEROREQUAL**

**,SKEYREQTYPE=RANGE**

Use this input parameter to specify how an existing keyed list entry is located and how key comparison is to be performed to determine if the list entry is selectable for processing.

**EQUAL**

The entry must have a key that equals the SECONDARYKEY key.

**LESSOREQUAL**

The entry must have a key that is less than or equal to the SECONDARYKEY key.

**GREATEROREQUAL**

The entry must have a key that is greater than or equal to the SECONDARYKEY key.

**RANGE**

The entry must have a key within the specified range of values. The SECONDARYKEY specified will be used as the beginning of the range of values. SKEYRANGEEND will be used as the ending value. A list entry must have an secondary key value within the specified entry key range, inclusive, for it to be selectable.

**Note:**

1. When no entries on the list meet the requirements of the SKEYREQTYPE, and a new entry cannot be created, the request will be failed, and the appropriate return and reason codes are provided to the invoker.
2. When LESSOREQUAL or GREATEROREQUAL are specified, if no entries on the list have a secondary key value equal to the specified value, but entries exist with a secondary key value greater than (if GREATEROREQUAL was specified), or less than (if LESSOREQUAL was specified) the entry key value specified, then the entry with a secondary key value closest to the value specified will be selected. When multiple entries have the same secondary key value, LISTPOS is used to resolve whether the first or last entry with the secondary key value is selected.
3. When LESSOREQUAL, GREATEROREQUAL or RANGE is specified, if multiple entries have a secondary key value within the specified range, DIRECTION will be used to resolve whether the first or last entry within the secondary key range is selected.
4. When SKEYREQTYPE=RANGE is specified, SKEYRANGEEND=YES is required.

**,TYPE=ENTDATA**  
**,TYPE=ADJDATA**  
**,TYPE=ECONTROLS**

Use this group of required input(s) to specify the type of information to be read. Any combination of ENTDATA, ADJDATA and ECONTROLS may be specified.

**ENTDATA**

Use ENTDATA to indicate that entry data is to be read.

**ADJDATA**

Use ADJDATA to indicate that adjunct data is to be read.

**ECONTROLS**

Use ECONTROLS to indicate that list entry control information is to be read.

**Note:**

1. ADJDATA is only functional for structures that support adjunct data.
2. For structures that are allocated with secondary keys, the first 32 bytes of the adjunct data will contain the secondary key information.
3. For structures allocated with secondary keys, the secondary key is not read by TYPE=ECONTROLS. The secondary key is read by TYPE=ADJDATA.

**,VERSCOMP=NO\_VERSCOMP**

**,VERSCOMP=verscomp**

Use this input parameter to specify a version number to be compared to the version number of the existing entry. If this request creates a new entry, the VERSCOMP specification is ignored. The existing entry is processed only if its version number meets the condition specified by the VERSCOMPTYPE parameter. If the condition specified by VERSCOMPTYPE is not met for the current list entry, then no processing is performed for the current list entry and processing continues with the next entry to be considered.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-byte field that contains the version number.

**,VERSCOMPARE=NO**

**,VERSCOMPARE=YES**

Use this input parameter to specify if the version number is to be compared to the version number of the existing entry.

**VERSCOMPARE=NO**

No version comparison should be performed to determine if each list entry should be processed.

**VERSCOMPARE=YES**

Version number comparison should precede processing of each list entry.

**,VERSCOMPTYPE=EQUAL**

**,VERSCOMPTYPE=LESSOREQUAL**

Use this input parameter to specify how a list entry version comparison as specified by VERSCOMP is to be performed.

**Note:** The VERSCOMPTYPE parameter is valid only for list structures allocated in a coupling facility with CFLEVEL=1 or higher.

**VERSCOMPTYPE=EQUAL**

The version number for the list entry must be equal to the value specified for VERSCOMP.

**VERSCOMPTYPE=LESSOREQUAL**

The version number for the list entry must be less than or equal to the value specified for VERSCOMP.

## ABEND Codes

Abend X'026' (See [z/OS MVS System Codes](#) for more information on this abend.)

## Return and Reason Codes

When the system returns control to your program:

- GPR 15 (and RETCODE, if specified) contains a return code.
- If the return code is not zero, GPR 0 (and RSNCODE, if specified) contains a reason code.

**Note:** The return and reason codes will also be put into the answer area (mapped by IXLYLAA), if there is one, when the request completes.

The IXLYCON macro provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

<b>0</b>	IXLRETCODEOK
<b>4</b>	IXLRETCODEWARNING
<b>8</b>	IXLRETCODEPARMERROR
<b>C</b>	IXLRETCODEENVERROR
<b>10</b>	IXLRETCODECOMPERROR

The following table contains hexadecimal return and reason codes, the equate symbols associated with each reason code, and the meaning and suggested action for each return and reason code.

Table 84. Return and Reason Codes for IXLLSTM Macro		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0	None.	<p><b>Meaning:</b> If you specified a MODE value of ASYNCECB, ASYNCEXIT, ASYNCTOKEN, or ASYNCNORESPONSE, the request has been successfully initiated. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed. For any other MODE value, the request has successfully completed.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=ASYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=ASYNCEXIT, the connection's complete exit will be given control when the request has completed.</li> <li>• If you specified MODE=ASYNCTOKEN, use REQTOKEN when calling IXLFCOMP to determine when the request has completed.</li> </ul>

Table 84. Return and Reason Codes for IXLLSTM Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	xxxx0402	<p><b>Equate Symbol:</b> IXLRNCODEASYNCH</p> <p><b>Meaning:</b> The request could not be processed synchronously. It will be processed asynchronously. The answer area (ANSAREA) fields will not contain valid information until the asynchronous processing has completed.</p> <ul style="list-style-type: none"> <li>• If MODE=SYNCECB was specified, the ECB specified for REQECB will be posted when the request has finished.</li> <li>• If MODE=SYNCEXIT was specified, the connection's complete exit will be given control when the request has finished.</li> <li>• If MODE=SYNCTOKEN was specified, an asynchronous request token was returned in the area specified by REQTOKEN. This request token may be specified on the IXLFCOMP macro to determine when the request has finished.</li> </ul> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• If you specified MODE=SYNCECB, wait on the ECB you specified in REQECB.</li> <li>• If you specified MODE=SYNCEXIT, the connection's complete exit will be given control when the request has completed.</li> <li>• If you specified MODE=SYNCTOKEN, use REQTOKEN when calling IXLFCOMP to determine when the request has completed.</li> </ul>
4	xxxx0409	<p><b>Equate Symbol:</b> IXLRNCODETIMEOUT</p> <p><b>Meaning:</b> The request completed prematurely because it exceeded the coupling facility model-dependent time-out criteria. The following information has been returned in the answer area:</p> <ul style="list-style-type: none"> <li>• The number of output elements returned in the BUFFER area or BUFLIST buffers (for READ_LIST and READ_MULT).</li> <li>• The number of deleted structure entries (for DELETE_MULT, DELETE_LIST and DELETE_ENTRYLIST).</li> <li>• A restart token or an extended restart token (for READ_MULT and DELETE_MULT).</li> <li>• The list entry controls for the first unprocessed entry in the list sequence (for READ_LIST and DELETE_LIST).</li> <li>• The index of the first unprocessed entry identifier or name (for DELETE_ENTRYLIST and MOVE_ENTRYLIST).</li> </ul> <p><b>Action:</b> Reissue the request to continue.</p> <p>Be sure to process the information returned from this request before reissuing the request. The data returned from this request will be overwritten if you specify the same buffer address. Continue to reissue the request until the return code indicates that all processing has completed.</p> <p>For more information about premature completion of an IXLLSTM request, see <a href="#">z/OS MVS Programming: Sysplex Services Guide</a>.</p>



Table 84. Return and Reason Codes for IXLLSTM Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	xxxx040D	<p><b>Equate Symbol:</b> IXLRSNCODEBADREADADJDATA</p> <p><b>Meaning:</b> Program error. The request specified that adjunct data was to be read, but the storage area specified by ADJAREA is not addressable. All other requested data was retrieved and the following fields in the answer area have been filled in:</p> <ul style="list-style-type: none"> <li>• LAARLRMLCTLS - The first processed entry in the list sequence.</li> <li>• LAAREADCNT - The number of entries processed successfully</li> <li>• LAALCTL (if the request completed prematurely as well) - The list entry controls for the first unprocessed entry in the list sequence.</li> </ul> <p><b>Note:</b> Only the adjunct data for the first entry in the list is placed in the ADJAREA. The buffers should contain the adjunct data for the other entries in the list.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the ADJAREA address.</li> <li>• ADJAREA must be addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are calling IXLLSTM while disabled, ADJAREA must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLLSTM in AR-mode and you specified the ADJAREA address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are calling IXLLSTM in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLSTM macro.</li> <li>• If LAALCTL is not zeros, the request completed prematurely. See the action suggested for return code X'4' reason code X'xxxx0409'.</li> </ul> <p>Correct the address specified by ADJAREA, and rerun the request asking for adjunct data only.</p>

Table 84. Return and Reason Codes for IXLLSTM Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	xxxx040F	<p><b>Equate Symbol:</b> IXLRSNCODEBUFFERFULL</p> <p><b>Meaning:</b> Program error. The request has completed prematurely because the buffer specified by BUFFER or the buffers specified by BUFLIST are full. For both IXLLSTM READ_LIST and READ_MULT requests, the number of output elements returned in the BUFFER area or BUFLIST buffers has been returned in the answer area. For READ_LIST, the list entry controls for the first unprocessed entry in the list sequence has been returned in the answer area.</p> <p><b>Action:</b> Reissue the request, or increase the size of the buffer(s), and rerun your program. You can reissue the request using the restart token (RESTOKEN) or extended restart token (EXTRESTOKEN).</p> <p>Be sure to process the information returned from this request before reissuing the request. The data returned from this request will be overwritten if you specify the same buffer address. Continue to reissue the request until the return code indicates that all processing has completed.</p> <p>For more information about premature request completion, see <a href="#">z/OS MVS Programming: Sysplex Services Guide</a>.</p>
4	xxxx0410	<p><b>Equate Symbol:</b> IXLRSNCODELOCKCOND</p> <p><b>Meaning:</b> For a LOCKOPER=HELDDBY request, or a request that specified LOCKCOMP, the request could not be completed successfully because the specified lock is not currently held as required. The connection identifier of the lock owner is returned in the answer area (LAACONID field).</p> <p><b>Action:</b> Retry the request, or obtain the lock as required, and retry the request. If you are unable to get the lock, check the ID in the LAACONID field and determine if some recovery is necessary.</p>
4	xxxx0412	<p><b>Equate Symbol:</b> IXLRSNCODELOCKHELDDBYSYS</p> <p><b>Meaning:</b> The lock is not held by any connection, but instead is held by the system. For an LOCKOPER=HELDDBY request, the request could not be completed successfully because the specified lock is not currently held as required:</p> <p><b>Action:</b> Retry the request, or obtain the lock as required, and retry the request.</p>

Table 84. Return and Reason Codes for IXLLSTM Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0801	<p><b>Equate Symbol:</b> IXLRSNCODEBADPARMLIST</p> <p><b>Meaning:</b> Program error. The parameter list for this request is not addressable. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Verify that:</p> <ul style="list-style-type: none"> <li>• The parameter list address is uncorrupted.</li> <li>• The parameter list is addressable in the caller's primary address space.</li> <li>• If you are issuing this macro while disabled, the parameter list resides in either page-fixed or DREF storage.</li> <li>• If you are issuing this macro in AR-mode and you specified the parameter list address using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If you are issuing this macro in AR-mode, SYSSTATE ASCENV=AR must be issued before issuing this macro.</li> </ul>
8	xxxx0804	<p><b>Equate Symbol:</b> IXLRSNCODEBADVERSIONNUM (or IXLRSNCODEBADVERSION#)</p> <p><b>Meaning:</b> The version number in the macro parameter list is not compatible with the level of XES currently being used. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that your program did not overlay the parameter list storage.</li> <li>• Verify that your program was assembled with the correct macro library for the release of MVS on which your program is running.</li> </ul>

Table 84. Return and Reason Codes for IXLLSTM Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx080A	<p><b>Equate Symbol:</b> IXLRSNCODEBADCONTOKEN</p> <p><b>Meaning:</b> Program error. The specified connect token (CONTOKEN) was not valid for one of the following reasons:</p> <ol style="list-style-type: none"> <li>1. The user with the connection identifier represented by the token has disconnected from the structure.</li> <li>2. The connector's task (the task that issued IXLCONN) ended.</li> <li>3. The specified token is not the token that was returned from IXLCONN.</li> <li>4. The request was issued from an address space other than the address space in which IXLCONN was issued.</li> <li>5. The connect token was invalidated during rebuild.</li> <li>6. The connect token was invalidated by XES.</li> </ol> <p><b>Note:</b> The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Take the action with the corresponding meaning.</p> <ol style="list-style-type: none"> <li>1. Discontinue use of the structure. Perform recovery and cleanup for the structure and check the protocol for the use of the structure to determine why you did not know of the disconnection.</li> <li>2. Discontinue use of the structure. Perform recovery and cleanup for the structure and check the protocol for the use of the structure to determine why you did not know that the task ended.</li> <li>3. The contoken is returned in the CONACONTOKEN field of the answer area specified on the IXLCONN request.</li> <li>4. Issue your request from the same address space the IXLCONN was issued in.</li> <li>5. Wait for the rebuild to complete, and try again.</li> <li>6. Discontinue use of the structure. Perform recovery and cleanup for the structure.</li> </ol>
8	xxxx0822	<p><b>Equate Symbol:</b> IXLRSNCODEBADREADTYPE</p> <p><b>Meaning:</b> Program error. You specified that either entry data or adjunct data should be returned, but the list structure does not contain the type of data you requested. No data is returned.</p> <p><b>Action:</b> Ensure that you are requesting the type of data you need from the structure and that the structure supports that type of data. Issue IXLMG to get more information about the structure.</p>

Table 84. Return and Reason Codes for IXLLSTM Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0824	<p><b>Equate Symbol:</b> IXLRSNCODEWRONGSTRTYPE</p> <p><b>Meaning:</b> Program error. The connection specified by CONTOKEN is not to a list structure.</p> <p><b>Action:</b> Specify the appropriate CONTOKEN value, and rerun your program. The CONTOKEN is returned in the answer area provided on the IXLCONN invocation. The type of structure built is passed on the TYPE parameter of the IXLCONN macro.</p>
8	xxxx0825	<p><b>Equate Symbol:</b> IXLRSNCODENOENTRY</p> <p><b>Meaning:</b> Program error. The designated list entry does not exist; therefore, no entries were processed.</p> <p><b>Action:</b> None necessary. However, if this return code and reason code are unexpected, you should check the way the list entry was created. Examine the parameters specified for locating the list entry on the invocation of this macro.</p>
8	xxxx082B	<p><b>Equate Symbol:</b> IXLRSNCODEBADIDINDEX</p> <p><b>Meaning:</b> Program error. The value specified for either FIRSTLEM or LASTLEM was not valid. Some entries may have been processed. When FIRSTLEM or LASTLEM is not valid, the index of the first entry identifier or name that was not processed is returned in the answer area.</p> <p><b>Action:</b> Ensure that the FIRSTLEM and LASTLEM values correspond to entries in the list of entries to be processed.</p>
8	xxxx0833	<p><b>Equate Symbol:</b> IXLRSNCODEBADPGBLATTR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list is specified as being pageable (PAGEABLE=YES), but is not.</p> <p><b>Action:</b> Change the buffer area(s) to pageable storage, or specify PAGEABLE=NO. See the PAGEABLE parameter description for specification instructions.</p>

Table 84. Return and Reason Codes for IXLLSTM Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0834	<p><b>Equate Symbol:</b> IXLRSNCODEBADNONPGBLATTR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list, is specified as being nonpageable (PAGEABLE=NO), but is either pageable or not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The type of storage in the buffer area corresponds to the value you specified on the PAGEABLE parameter. See the PAGEABLE parameter description for specification instructions.</li> <li>• The correct buffer address was used.</li> <li>• The buffer area was not previously freed.</li> <li>• If you are calling IXLLSTM while disabled, the buffers must reside in either page-fixed or DREF storage.</li> <li>• The buffer areas were allocated in a storage key that matches the key specified by the BUFSTGKEY parameter, or, if BUFSTGKEY is omitted, the caller's PSW key.</li> <li>• If BUFLIST was specified and your program is running in AR-mode: <ul style="list-style-type: none"> <li>– If the BUFFER or BUFLIST parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>– The BUFALET specification is correct.</li> <li>– You specified SYSSTATE ASCENV=AR before issuing the IXLLSTM macro.</li> </ul> </li> </ul>
8	xxxx0835	<p><b>Equate Symbol:</b> IXLRSNCODEBADDATAADDR</p> <p><b>Meaning:</b> Program error. The storage area specified by BUFFER, or one of the buffers in the BUFLIST list, is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The correct buffer address was used.</li> <li>• The buffer area was not previously freed.</li> <li>• The buffer area was allocated in a storage key that matches the key specified using the BUFSTGKEY parameter, or, if BUFSTGKEY is omitted, the caller's PSW key.</li> <li>• If BUFLIST was specified and your program is running in AR mode: <ul style="list-style-type: none"> <li>– The BUFALET specification is correct.</li> <li>– You specified SYSSTATE ASCENV=AR before issuing the macro.</li> </ul> </li> </ul>

Table 84. Return and Reason Codes for IXLLSTM Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0836	<p><b>Equate Symbol:</b> IXLRNCODEBADREALADDR</p> <p><b>Meaning:</b> Program error. Real storage addresses were provided in the BUFLIST list, but one of the buffers is not addressable in central storage.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that BUFADDRTYPE was specified as you intended.</li> <li>• Ensure that the buffer addresses specified by BUFLIST are valid.</li> </ul>
8	xxxx0838	<p><b>Equate Symbol:</b> IXLRNCODEBADANSAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by ANSAREA is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The answer area address specified by ANSAREA is valid.</li> <li>• If the caller is running in AR-mode and the ANSAREA parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• The address specified for ANSAREA is addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are calling IXLLSTM while disabled, ANSAREA must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLLSTM in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLSTM macro.</li> </ul>
8	xxxx0839	<p><b>Equate Symbol:</b> IXLRNCODEBADREQTOKENAREA</p> <p><b>Meaning:</b> Program error. The storage area specified by REQTOKEN is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The request token area specified by REQTOKEN is valid.</li> <li>• If the caller is running in AR-mode and the REQTOKEN parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If you are calling IXLLSTM while disabled, REQTOKEN must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLLSTM in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLSTM macro.</li> </ul>

Table 84. Return and Reason Codes for IXLLSTM Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx083D	<p><b>Equate Symbol:</b> IXLRSNCODEBADANSLEN</p> <p><b>Meaning:</b> Program error. The length of the answer area, as specified by ANSLEN, is not sufficient to contain answer area information. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b> Increase the size of the answer area, and rerun your program. The minimum size of the ANSAREA is in the LAA_LEN field of the IXLYLAA mapping macro.</p>
8	xxxx083E	<p><b>Equate Symbol:</b> IXLRSNCODEMAXLISTKEY</p> <p><b>Meaning:</b> Program error. The list key to be assigned to an entry that was being created or moved was not valid. Either the list key or the list key plus the list key increment value was greater than the maximum list key.</p> <p><b>Action:</b> Ensure that you specified a correct list key increment. Depending on the protocol you are using for assigning list key values, you might want to issue a WRITE_LCONTROLS request to update either the list key value to a lower value or the maximum list key value to a higher value.</p>
8	xxxx083F	<p><b>Equate Symbol:</b> IXLRSNCODEBADENTRYVERSION</p> <p><b>Meaning:</b> The entry designated by the specified list entry controls has a version number that does not meet the criteria specified by VERSCOMP and VERSCOMPTYPE. The list entry controls for the entry are returned in the answer area (field LAALCTL).</p> <p><b>Action:</b> None necessary. However, if this return code and reason code are unexpected, you might want to verify the values specified in VERSCOMP and VERSCOMPTYPE and look at the version returned in the answer area.</p>
8	xxxx0840	<p><b>Equate Symbol:</b> IXLRSNCODEBADENTRYLIST</p> <p><b>Meaning:</b> The entry designated by the specified list entry controls does not reside on the list specified by LISTNUM. The list entry controls for the entry are returned in the answer area (field LAALCTL).</p> <p><b>Action:</b> None necessary. However, if you did not expect this return and reason code, you might want to verify what list this entry is on. Maybe the entry was moved, or you designated the wrong list in LISTNUM. Check the protocol for using this list.</p> <p>The information returned in the LAALCTL field of the answer area contains the number of the list that this list entry is on as well as other control information.</p>



Table 84. Return and Reason Codes for IXLLSTM Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0841	<p><b>Equate Symbol:</b> IXLRSNCODEBADENTRYNAME</p> <p><b>Meaning:</b> Program error. The name specified by ENTRYNAME is not a unique name within the structure, and therefore entry creation is suppressed.</p> <p><b>Action:</b> Be sure to provide a unique entry name when creating entries to be written to structures that support entry names.</p>
8	xxxx0842	<p><b>Equate Symbol:</b> IXLRSNCODEPERSISTENTLOCK</p> <p><b>Meaning:</b> Program error. The request specifying a NOTHELD lock operation failed because the lock was held by a connection that is in the failed-persistent state. The connection identifier of the lock owner is returned in the answer area (field LAACONID).</p> <p><b>Action:</b> Either perform recovery for the connection, or wait until recovery for the connection is performed.</p>
8	xxxx0843	<p><b>Equate Symbol:</b> IXLRSNCODEBADENTRYID</p> <p><b>Meaning:</b> Program error. The entry corresponding to either an entry identifier or entry name in either IDLIST or NAMELIST, respectively, does not exist. The index to the failing entry identifier or name was returned in the answer area (field LAAFAILINDEX). The count of entries deleted is also returned in the answer area (field LAADELCNT).</p> <p><b>Action:</b> Remove the failing entry from the list, or reissue the request. Update FIRSTELEM to point after the failing entry (<math>\text{FIRSTELEM} = \text{LAAFAILINDEX} + 1</math>) to the next entry on the list to be processed.</p>
8	xxxx0845	<p><b>Equate Symbol:</b> IXLRSNCODENONAMES</p> <p><b>Meaning:</b> Program error. A list entry was designated by entry name, but the structure does not support entry names.</p> <p><b>Action:</b> Ensure that you are connected to the intended structure. Ensure that the correct specification was made for REFOPTION on the IXLCONN macro. You may want to issue IXLMG to get more information about the structure.</p>
8	xxxx0846	<p><b>Equate Symbol:</b> IXLRSNCODEBADLOCKINDEX</p> <p><b>Meaning:</b> Program error. The specified LOCKINDEX exceeds the size of the lock table for the structure.</p> <p><b>Action:</b> Correct LOCKINDEX to specify an index that is contained within the lock table. The maximum value for the LOCKINDEX is one less than the value specified by the LOCKENTRIES parameter on the IXLCONN macro.</p>

Table 84. Return and Reason Codes for IXLLSTM Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0847	<p><b>Equate Symbol:</b> IXLRSNCODEBADLISTNUMBER</p> <p><b>Meaning:</b> Program error. The specified LISTNUM value exceeds the number of lists for the structure.</p> <p><b>Action:</b> Correct LISTNUM to specify a list number that exists in the structure. The number of lists allocated for a structure is determined on the LISTHEADERS parameter of the IXLCONN macro. The list numbers go from 0 to n-1, where n is the number of lists specified.</p>
8	xxxx0849	<p><b>Equate Symbol:</b> IXLRSNCODEBADRESTOKEN</p> <p><b>Meaning:</b> Program error. The restart token specified by RESTOKEN is not valid.</p> <p><b>Action:</b> Determine why the restart token cannot be used to restart the request. Possible causes are:</p> <ul style="list-style-type: none"> <li>• The specified token does not correspond to the restart token returned in the answer area of the previous request (field LAARESTOKEN).</li> <li>• The user specified RESTOKEN when EXTRESTOKEN was required.</li> <li>• The user specified EXTRESTOKEN when RESTOKEN was required.</li> </ul> <p>For more information about premature request completion, see <a href="#">z/OS MVS Programming: Sysplex Services Guide</a>.</p>
8	xxxx084A	<p><b>Equate Symbol:</b> IXLRSNCODENOKEYS</p> <p><b>Meaning:</b> Program error. The structure does not support the use of entry keys. The IXLLSTM request type either required the structure to support entry keys or designated a sublist, list entry, or list position by list number and entry key.</p> <p><b>Action:</b> Ensure that you are connected to the intended structure. The use of keys for a structure is determined by the REFOPTION keyword on the IXLCONN macro.</p>
8	xxxx084B	<p><b>Equate Symbol:</b> IXLRSNCODENOLOCKS</p> <p><b>Meaning:</b> Program error. A locking operation was requested, but the structure does not contain a lock table.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• You are connected to the intended structure.</li> <li>• You intended to perform a locking operation.</li> </ul> <p>The use of locks is determined by the LOCKENTRIES keyword on the IXLCONN macro.</p>

Table 84. Return and Reason Codes for IXLLSTM Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx084E	<p><b>Equate Symbol:</b> IXLRSNCODEBADMOVETOLIST</p> <p><b>Meaning:</b> Program error. The list number specified for MOVETOLIST exceeds the number of lists defined for the structure.</p> <p><b>Action:</b> Correct MOVETOLIST to specify a list that exists in the structure. The maximum number of lists in a structure is determined by the LISTHEADERS parameter on the IXLCONN macro.</p>
8	xxxx0851	<p><b>Equate Symbol:</b> IXLRSNCODENOSUSPENDISABLE</p> <p><b>Meaning:</b> Program error. The request failed because MODE=SYNCSUSPEND was specified, but the caller is disabled and cannot be suspended.</p> <p><b>Action:</b> Either specify another MODE value, or become enabled (release the CPU lock), then reissue the request.</p>
8	xxxx0854	<p><b>Equate Symbol:</b> IXLRSNCODEBADLOCKCOMP</p> <p><b>Meaning:</b> Program error. The connection identifier specified for LOCKCOMP is not valid.</p> <p><b>Action:</b> The connection identifier is returned in the answer area (mapped by IXLYCONA) when the IXLCONN macro was issued.</p>
8	xxxx0859	<p><b>Equate Symbol:</b> IXLRSNCODEBADLISTAUTH</p> <p><b>Meaning:</b> The list authority value for the specified list does not meet the criteria specified by AUTHCOMP and AUTHCOMPTYPE. The current list authority (field LAALISTAUTH) and description (field LAALISTDESC) are returned in the answer area.</p> <p><b>Action:</b> None required; however you might want to take some action depending on your application. The list authority is set to binary zeros when the structure is allocated. When IXLLSTM is issued with the NEWAUTH keyword, the list authority is changed if the correct comparison list authority value is specified. Check the answer area to determine the list authority value for the list specified. Verify that the correct list number was specified.</p>
8	xxxx0864	<p><b>Equate Symbol:</b> IXLRSNCODEBADBUFSIZE</p> <p><b>Meaning:</b> Program error. The size of the BUFFER area or the buffer areas specified by BUFLIST is not large enough to contain the data being read. No data is returned.</p> <p><b>Action:</b> If more space is available, specify a larger buffer size, and reissue the request. For READ_LIST and READ_MULT requests, the answer area contains the list entry controls for the first list entry selected for processing.</p>

Table 84. Return and Reason Codes for IXLLSTM Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0865	<p><b>Equate Symbol:</b> IXLRSNCODEBADBUFSPEC</p> <p><b>Meaning:</b> Program error. There is an error in the buffer specification.</p> <p><b>Action:</b> Check the following:</p> <ul style="list-style-type: none"> <li>• If BUFLIST was specified, check the requirements for BUFLIST, BUFNUM, and BUFINCRNUM.</li> <li>• If BUFFER was specified, check the requirements for BUFFER and BUFSIZE.</li> <li>• Buffer pointer(s) in BUFLIST</li> <li>• Buffer boundaries.</li> </ul>
8	xxxx0866	<p><b>Equate Symbol:</b> IXLRSNCODEBADBUFKEY</p> <p><b>Meaning:</b> Program error. The buffer storage key specified by BUFSTGKEY is incorrect. For requests that write coupling facility data, the data cannot be fetched from the specified buffer area. For requests that read coupling facility data, the data cannot be stored into the specified buffer area.</p> <p><b>Action:</b> Check the following:</p> <ul style="list-style-type: none"> <li>• Determine if the key of the storage being used for the buffers is different from the PSW key.</li> <li>• If BUFSTGKEY was specified, verify that the key was put in the correct bits (see the explanation of this parameter for more information).</li> <li>• If you are calling IXLLSTM in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLSTM macro.</li> </ul>
8	xxxx0867	<p><b>Equate Symbol:</b> IXLRSNCODEBADBUFLIST</p> <p><b>Meaning:</b> Program error. The 128-byte storage area specified by BUFLIST is not addressable.</p> <p><b>Action:</b> Ensure that the address specified by BUFLIST is valid.</p>
8	xxxx086A	<p><b>Equate Symbol:</b> IXLRSNCODEBADELEMNUM</p> <p><b>Meaning:</b> Program error. The value specified for ELEMNUM is not valid.</p> <p><b>Action:</b> Correct the ELEMNUM specification to be within the allowable range: from zero to whatever MAXELEMNUM value was returned to the connector in the connect answer area.</p>

Table 84. Return and Reason Codes for IXLLSTM Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0887	<p><b>Equate Symbol:</b> IXLRSNCODEBADEXTRESTOKEN</p> <p><b>Meaning:</b> Program error. The extended restart token specified by EXTRESTOKEN is not valid. The specified token refers to an older instance of the target structure.</p> <p><b>Action:</b> Reinitiate the request with an EXTRESTOKEN value of zero. A system-managed process occurred between the time a request returned the extended restart token and the time the connector tried to continue the request using that token. Discard the results of the initial request and reissue the request. For more information about restarting requests, see <a href="#">z/OS MVS Programming: Sysplex Services Guide</a>.</p>
8	xxxx0894	<p><b>Equate Symbol:</b> IXLRSNCODEBADKEYCOMPARE</p> <p><b>Meaning:</b> Program error. The value specified for ENTRYKEY used with the KEYCOMPARE=YES with KEYREQTYPE did not match. The value specified for SECONDARYKEY used with SKEYCOMPARE=YES with SKEYREQTYPE did not match.</p> <p><b>Action:</b> Ensure that the values specified for either ENTRYKEY or SECONDARYKEY are valid.</p>
8	xxxx0895	<p><b>Equate Symbol:</b> IXLRSNCODEBADKEYLISTAREA</p> <p><b>Meaning:</b> Program error. The storage area specified for LISTKEYAREA is not addressable.</p> <p><b>Action:</b> Ensure that:</p> <ul style="list-style-type: none"> <li>• The list key area address specified by LISTKEYAREA is valid.</li> <li>• If the caller is running in AR-mode and the LISTKEYAREA parameter was specified using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• The address specified for LISTKEYAREA is addressable in the caller's primary address space or from the caller's PASN access list.</li> <li>• If you are calling IXLLSTM while disabled, LISTKEYAREA must reside in either page-fixed or DREF storage.</li> <li>• If you are calling IXLLSTM in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLLSTM macro.</li> </ul>
8	xxxx0897	<p><b>Equate Symbol:</b> IXLRSNCODEBADKEYTYPE</p> <p><b>Meaning:</b> Program error. The specified KEYTYPE value is not valid for the specified structure.</p> <p><b>Action:</b> Ensure that the value of KEYTYPE is valid.</p>
8	xxxx0898	<p><b>Equate Symbol:</b> IXLRSNCODEBADKEYSCANTYPE</p> <p><b>Meaning:</b> Program error. The specified KEYSKANTYPE value is not valid for the specified structure.</p> <p><b>Action:</b> Correct the value of KEYSKANTYPE to reflect the use of either entry keys or secondary keys.</p>

Table 84. Return and Reason Codes for IXLLSTM Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0899	<b>Equate Symbol:</b> IXLRSNCODEBADSKKEYCOMPARE <b>Meaning:</b> Program error. The specified SKEYCOMPARE value is not valid for the specified structure. <b>Action:</b> Correct the value of SKEYCOMPARE.
8	xxxx089A	<b>Equate Symbol:</b> IXLRSNCODEBADSKKEYREQTYPE <b>Meaning:</b> Program error. The specified SKEYREQTYPE value is not valid for the specified structure. <b>Action:</b> Correct the value of SKEYREQTYPE.
8	xxxx089B	<b>Equate Symbol:</b> IXLRSNCODEBADKEYCOMPARETYPE <b>Meaning:</b> Program error. The specified KEYCOMPARE value is not valid for the specified structure. <b>Action:</b> Correct the value of KEYCOMPARE.
8	xxxx089C	<b>Equate Symbol:</b> IXLRSNCODEBADMOVETOKEY <b>Meaning:</b> Program error. The specified MOVETOKEY value is not valid for the specified structure. <b>Action:</b> Ensure that the value of MOVETOKEY is correct.
8	xxxx089D	<b>Equate Symbol:</b> IXLRSNCODEBADMOVETOSKEY <b>Meaning:</b> Program error. The specified MOVETOSKEY value is not valid for the specified structure. <b>Action:</b> Ensure that the value of MOVETOSKEY is correct.
C	xxxx0C06	<b>Equate Symbol:</b> IXLRSNCODENOCNN <b>Meaning:</b> Environmental error. This system does not have connectivity to the coupling facility that contains the list structure. Possible reasons for this are: <ul style="list-style-type: none"> <li>• The operator issued VARY PATH,OFFLINE.</li> <li>• The operator issued CONFIG CHP,OFFLINE.</li> <li>• Hardware errors to the coupling facility.</li> <li>• Facility or path failure to the coupling facility.</li> </ul> <b>Action:</b> Begin rebuilding the structure on a different coupling facility, or disconnect from the structure.

Table 84. Return and Reason Codes for IXLLSTM Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C13	<p><b>Equate Symbol:</b> IXLRNICODEREQPURGED</p> <p><b>Meaning:</b> Environmental error. The request was purged prior to completion of the request. Possible reasons include:</p> <ul style="list-style-type: none"> <li>• The connector failed.</li> <li>• The connector disconnected.</li> <li>• The requestor failed.</li> <li>• The request was purged by IXLPURGE.</li> <li>• Requests were purged when the connector provided an IXLEERSP response for the Rebuild Stop or Rebuild Cleanup event.</li> <li>• The secondary address space was no longer valid.</li> </ul> <p><b>Action:</b> None if this is expected. Otherwise, determine why the connector failed.</p>
C	xxxx0C14	<p><b>Equate Symbol:</b> IXLRNICODESTATUSUNKNOWN</p> <p><b>Meaning:</b> Environmental error. The request has completed, but the final disposition of the request cannot be determined. The answer area (ANSAREA) fields do not contain valid information.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify the validity of your data by comparing the expected results with what is in the coupling facility.</li> <li>• Consider running rebuild processing to move the structure to attempt to protect it from further errors of this kind.</li> </ul>
C	xxxx0C18	<p><b>Equate Symbol:</b> IXLRNICODELISTFULL</p> <p><b>Meaning:</b> Environmental error. The entry could not be placed on the designated target list because the list is full.</p> <p><b>Action:</b> Either change the maximum number of entries allowed on the list by specifying a new LISTLIMIT on a WRITE_LCONTROLS request. You should be monitoring the usage of the list structure every time a read or write is done. (LAATOTALCNT is the total count of allocated entries in the list structure, and LAATOTALELECNT is the total count of allocated elements in the list structure.) This data should be evaluated periodically to ensure structure resources are being used efficiently.</p>
C	xxxx0C25	<p><b>Equate Symbol:</b> IXLRNICODESTRFAILURE</p> <p><b>Meaning:</b> Environmental error. The list structure failed prior to completion of the request.</p> <p><b>Action:</b> Either rebuild or disconnect from the structure.</p>

Table 84. Return and Reason Codes for IXLLSTM Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C68	<b>Equate Symbol:</b> IXLRSNCODEBADREQCFLEVEL <b>Meaning:</b> Environmental error. The request type is not permitted for the level of coupling facility in which the target structure is allocated. <b>Action:</b> Either disconnect from the structure (using IXLDISC) or initiate a rebuild of the structure, if allowed (using IXLREBLD) in a coupling facility of the correct CFLEVEL.
C	xxxx0CA0	<b>Equate Symbol:</b> IXLRSNCODEQUIESCEDSUSPENDFAIL <b>Meaning:</b> Environmental error. The request is failed because the structure is quiesced for a system-managed process and SUSPEND=FAIL is specified on the IXLCONN. <b>Action:</b> None, if this is expected.
C	xxxxFFFF	<b>Equate Symbol:</b> IXLRSNCODENOTAVAILABLE <b>Meaning:</b> Environmental error. XES functions are not available. This could occur because the coupling facility hardware necessary to provide XES functions is not present. <b>Action:</b> Re-IPL the system, or follow your particular failure management protocol.
10	xxxx10xx	<b>Meaning:</b> System error. XES processing failure. The state of the involved structure and the disposition of the request are unpredictable. The answer area (ANSAREA) fields do not contain valid information. <b>Action:</b> Contact the IBM support center.



## Chapter 79. IXLMG – Coupling facility measurement

### Description

The IXLMG macro allows an authorized caller to request measurement data related to the use of a coupling facility. The information is available by coupling facility or by structure. Information on subchannels and connectivity is returned when any coupling facility information is requested.

Structures in the IXLYAMDA mapping macro provide the format for the data:

- IXLYAMDHD maps common area for header entries
- IXLYAMDAREA maps the header record
- IXLYAMDCF and IXLYAMDCF1 map coupling facility information
- IXLYAMDSTRL, IXLYAMDSTRL1, IXLYAMDSTRL2, IXLYAMDSTRL3, and IXLYAMDSTRL4 map coupling facility list structure information
- IXLYAMDSTRC, IXLYAMDSTRC1, IXLYAMDSTRC2, IXLYAMDSTRC3, and IXLYAMDSTRC4 map coupling facility cache structure information
- IXLYAMDSC and IXLYAMDSC1 map coupling facility subchannel information
- IXLYAMDSLL and IXLYANDSLL1 map structure limits for a list structure
- IXLYAMDSLC and IXLYAMDSLC1 map structure limits for a cache structure
- IXLYAMDSSCM maps storage-class memory information for a structure
- IXLYAMDCFMI maps coupling facility measurement header information
- IXLYAMDCFMINFO maps coupling facility measurement information
- IXLYAMDCFRF and IXLYAMDCFRF1 map coupling facility remote facility information
- IXLYAMDCFCP maps coupling facility channel path header information
- IXLYAMDCFCPINFO maps coupling facility channel path information
- IXLYAMDSCSC and IXLYANDSCSC1 map structure information for cache storage classes
- IXLYAMDSCOC maps structure information for cache cast-out classes
- IXLYAMDSCOCSTATS maps structure information for cache cast-out classes
- IXLYAMDSSCC maps the structure copy controls record.
- IXLYAMDADUP maps the structure asynchronous duplexing record

The data returned may be system-oriented data (system attachment information or statistics on the system's use of a structure) or sysplex-wide data (data that is retrieved from the coupling facility itself).

### Environment

The following are the environment requirements for the caller.

Environment	Environment requirement
Minimum authorization:	Supervisor state or PKM allowing key 0 - 7
Dispatchable unit mode:	Task or SRB for CFNAME invocations Task for STRNAME invocations
Cross memory mode:	Any PASN, any HASN, any SASN
AMODE:	31-bit
ASC mode:	Primary or access register (AR)
Interrupt status:	Enabled or disabled for I/O and external interrupts

Environment	Environment requirement
Locks:	No locks held or only the CPU lock held
Control parameters:	Must be in the primary address space

## Programming Requirements

---

If the program is in AR mode, issue SYSSTATE ASCENV=AR before invoking IXLMG. SYSSTATE ASCENV=AR tells the system to generate code appropriate for AR mode.

The IXLYAMDA macro provides the format of the area that DATAAREA points to. Include that macro in your program.

## Restrictions

---

The input parameter list must be addressable in the caller's primary address space.

## Input Register Information

---

Before issuing the IXLMG macro, the caller does not have to place any information into any register unless using it in register notation for a particular parameter, or using it as a base register.

## Output Register Information

---

When control returns to the caller of the IXLMG macro, the general purpose registers (GPRs) contain:

### Register

#### Contents

**0**

Reason code, if applicable, if GPR 15 return code is non-zero

**1**

Used as work register by the system

**2-13**

Unchanged

**14**

Used as work register by the system

**15**

Return code

When control returns to the caller of the IXLMG macro, the access registers (ARs) contain:

### Register

#### Contents

**0-1**

Used as work registers by the system

**2-13**

Unchanged

**14-15**

Used as work registers by the system

**For registers that the system changes**, a caller who depends on these registers containing the same value before and after issuing the macro must save these registers before issuing the macro, and restore them after the system returns control.

## Performance Implications

IXLMG processing might involve referencing or updating the CFRM active policy to reflect the operation that has been requested. When invoking this macro, be aware of the I/O to the CFRM couple data set that might be generated.

## Understanding IXLMG Version Support

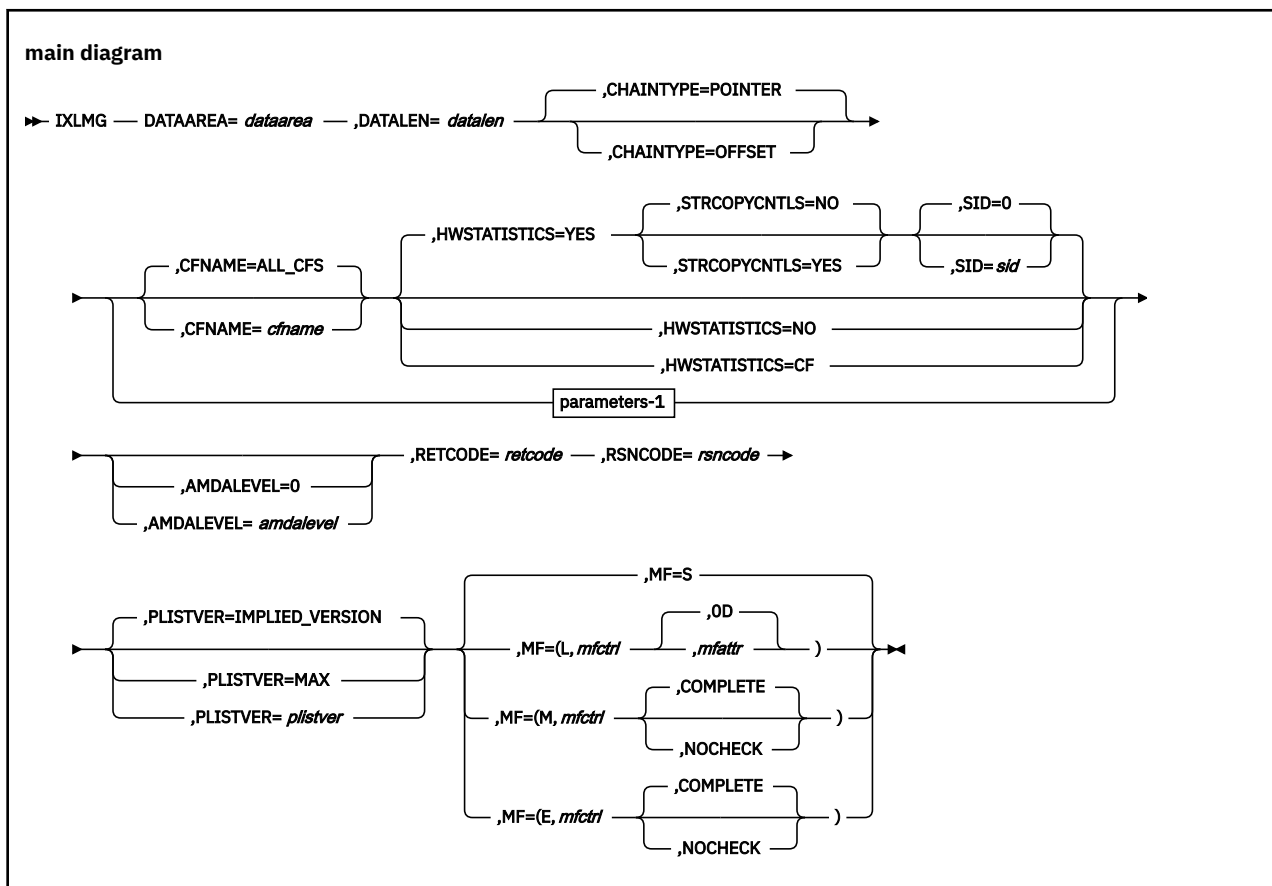
The IXLMG macro supports versions in the range of 0 - 2.

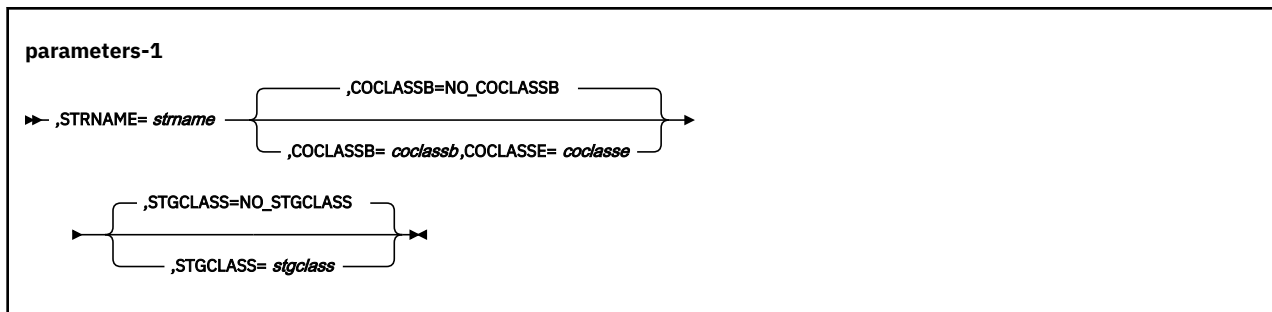
- Keywords not specifically noted here are supported by all versions starting with version 0 and higher of the IXLMG macro.
- The following keyword is supported by all versions starting with version 1 and higher of the IXLMG macro.
  - AMDALEVEL
- The following keyword is supported by all versions starting with version 2 and higher of the IXLMG macro.
  - STRCOPYCNTLS

Specify the version of the parameter list that you want generated with the PLISTVER keyword. See Chapter 2, “Specifying a Macro Version Number,” on page 5 for considerations when specifying the version of the parameter list with PLISTVER.

## Syntax diagram

The syntax of the IXLMG macro is as follows:





## Parameter descriptions

The parameter descriptions are listed in alphabetical order. Default values are underlined.

### **,AMDALEVEL=0**

#### **,AMDALEVEL=*amdalevel***

Use this input parameter to specify the level of the IXLYAMDA record mappings that the system returns in the answer area. Valid values are 0-4.

- A value of 0 indicates that base level IXLYAMDA records will be returned.
- A value of N indicates that level-N IXLYAMDA records will be returned. To request a level-N IXLYAMDA record, use version 1 of the IXLING macro by specifying AMDALEVEL=N.

IXLMG AMDALEVEL=2 requires system-managed asynchronous duplexing support. If IXLING AMDALEVEL=2 is requested on a system that does not support it, the request is rejected for the unsupported AMDALEVEL (IxIRsnCodeBadAmdaLevel). Macro IXCYQUAA defines the QuReqRfAsyncDuplex bit in the QuReqFeatures string that can be used to test for system-managed asynchronous duplexing support.

The QuReqRfWriteReadMetrics bit in the QuReqFeatures string can be used to test for support of AMDALEVEL=3 on the system.

The QuReqRfAmdaLevel4 bit in the QuReqFeatures string can be used to test for support of AMDALEVEL=4 on the system.

Use IXCQUERY REQINFO=FEATURES to get the QuReqFeatures string.

### **,CFNAME=ALL\_CFS**

#### **,CFNAME=*cfname***

Use this input parameter to specify the name of the coupling facility for which data is to be gathered or to indicate that data is to be collected for all coupling facilities that are attached to the system on which the IXLING macro is issued. If not specified, information about all coupling facilities that have a configured attachment to the system on which the IXLING macro is invoked will be gathered. If CFNAME specifies a coupling facility name, the coupling facility and all allocated structures are included in the reported data.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the eight-character field that contains the name of the coupling facility for which data is to be gathered.

### **,CHATYPE=POINTER**

#### **,CHATYPE=OFFSET**

Use this input parameter to specify how the entries in the IXLYAMDA data area should be chained. This selection applies to all chain fields in the data area, including:

- IXLYAMDAREA\_CFEnt@
- Chain fields within entry types that link to the next entry of the same type. These fields are mapped generically by IXLYAMDHD\_Next.
- Chain fields within entry types that link to entries of another type, such as IXLYAMDCF\_SL@ or IXLYAMDCF1\_RFAddr.

Test the QuReqRfIxlmChainType bit in the QuReqFeatures string returned by an IXCQUERY REQINFO=FEATURES invocation to determine whether the current system supports the CHAINTYPE keyword. Macro IXCYQUAA defines the QuReqRfIxlmChainType flag and QuReqFeatures string.

#### **POINTER**

Data area entries are to be chained by pointers. Each chain field contains an address within the area identified by the DataArea key.

#### **OFFSET**

Data area entries are to be chained by offsets. Each chain field contains an offset from the start of the area identified by the DataArea key. The address of the chained entry can be obtained by adding the offset to the address of the data area.

#### **,COCLASSB=NO\_COCLASSB**

#### **,COCLASSB=coclassb**

Use this input parameter to specify the first cast-out class to be reported on for a cache structure. The specified value must fall within the range of 1 to the maximum coupling facility class value defined for the structure, inclusive. (The number of cast-out classes is determined on the IXLCONN macro.)

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the 2-byte field that contains the first cast-out class in the range of cast-out classes to be reported on.

#### **,COCLASSE=coclasse**

Use this input parameter to specify the last cast-out class to be reported on for a cache structure. The specified value must fall within the range 1 to the maximum cast-out class value defined for the structure, inclusive, and be greater than or equal to the value for COCLASSB.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of a 2-byte field that contains the last cast-out class in the range of cast-out classes to be reported on.

#### **DATAAREA=dataarea**

Use this output parameter to identify the name of the output storage area that is to contain the measurement data returned by the measurement gathering routine. The storage area is defined by the macro IXLYAMDA. The storage area that is provided must be in fixed or disabled reference storage.

This macro does not clear the caller's storage area. It is the caller's responsibility to clear the area, if this is required.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the name of the storage area to contain the measurement data.

#### **,DATALEN=datalen**

Use the input parameter to specify the length of the storage area to contain the measurement data returned by the measurement gathering routine.

Use the IXLYAMDA macro to determine the length of the DATAAREA by determining the length of each section that you request and how many times each section will be repeated.

The minimum length is the length of the IXLYAMDA header. If the DATAAREA is not large enough to hold all the data requested, a return code X'4' with a reason code of IXLRNCCODEMOREDATA will be returned. The length of the output data will be returned in the IXLYAMDA, in the field IXLYAMDAREA\_TLEN.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the fullword field that contains the length of the storage area.

#### **,HWSTATISTICS=YES**

#### **,HWSTATISTICS=NO**

#### **,HWSTATISTICS=CF**

Use this input parameter to indicate whether the statistics that are gathered from the coupling facility should be included in the measurement data. The statistics include coupling facility measurement information and structure control information.

#### **YES**

Include statistics that are gathered from the coupling facility in the returned measurement data.

**NO**

Do not include statistics that are gathered from the coupling facility in the returned measurement data.

**CF**

Include statistics that are gathered from the coupling facility only for coupling facility measurement information, and not for structure control information.

**Note:**

1. This option does not apply to the STRNAME keyword.
2. If HWSTATISTICS=YES is specified, the data includes information that is retrieved from the coupling facility, specifically structure control information for each structure and measurement data for each coupling facility. The data returned will contain information on all allocated structures in the coupling facility whether the structure has an active connector from the system on which IXLMG was invoked or not.  
  
A structure ID can be specified to limit structure processing to that specific structure ID, otherwise all allocated structures are processed. The structure information returned will only include structures that are currently allocated in the coupling facility, and might contain structures that are allocated in the coupling facility, and might contain structures that are not currently known currently in the active policy.  
  
This information might contain structures that are not currently known in the CFRM policy.
  - The data returned for structures with active connectors from the system on which IXLMG was invoked contains both structure usage and control information.
  - The data returned for structures with no active connectors from the system on which IXLMG was invoked contains only structure control information.
  - The data that is returned for each coupling facility contains coupling facility usage, control, and measurement information.
3. If HWSTATISTICS=NO is specified, the data will not include structure control information or facility measurement information that is retrieved from the coupling facility.
  - The data returned for structures with active connectors from the system on which IXLMG was invoked contains structure usage information.
  - No data is returned for structures with no active connectors from the system on which IXLMG was invoked.
  - The data that is returned for each coupling facility contains coupling facility usage and control information.
4. If HWSTATISTICS=CF is specified, the data includes measurement data for each coupling facility that is retrieved from the coupling facility, but will not include the structure control information for each structure.
  - The data returned for structures with active connectors from the system on which IXLMG was invoked contains structure usage information.
  - No data is returned for structures with no active connectors from the system on which IXLMG was invoked.
  - The data that is returned for each coupling facility contains coupling facility usage, control, and measurement information.

**,MF=S**  
**,MF=(L,mfctrl)**  
**,MF=(L,mfctrl,mfattr)**  
**,MF=(L,mfctrl,0D)**  
**,MF=(M,mfctrl)**  
**,MF=(M,mfctrl,COMPLETE)**  
**,MF=(M,mfctrl,NOCHECK)**  
**,MF=(E,mfctrl)**  
**,MF=(E,mfctrl,COMPLETE)**  
**,MF=(E,mfctrl,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

**,mfctrl**

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

**,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

**,COMPLETE**

**,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

**COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if SMILE=*var* were an optional parameter and the default is SMILE=NO\_SMILE then it would not be documented. However, if the default was SMILE=:), then it would be documented because a value would be the default.

**NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

**,PLISTVER=IMPLIED\_VERSION**

**,PLISTVER=MAX**

**,PLISTVER=plistver**

Use this input parameter to specify the version of the macro. See [“Understanding IXLMG Version Support”](#) on page 1523 for a description of the options available with PLISTVER.

**,RETCODE=retcode**

Use this output parameter to specify the location in which the system is to copy the return code from GPR 15.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a fullword to contain the return code.

**,RSNCODE=rsncode**

Use this output parameter to specify the location in which the system is to copy the reason code from GPR 0.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of a fullword to contain the reason code.

**,STGCLASS= NO STGCLASS****,STGCLASS=stgclass**

Use this input parameter to specify for a cache structure the storage class for which you would like storage class statistics.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12to) of a 1-byte field that contains the storage class for which storage class statistics should be reported.

**,SID=0****,SID=sid**

Use this input parameter to specify the structure ID for which data is to be gathered and reported. If not specified (or a value of zero is specified), information about all allocated structures will be gathered and reported. When the structure ID is known, specifying it can be more efficient than using the IXLMG STRNAME key. Note that one or more structure IDs associated with a structure name can change through deallocation/allocation and rebuild processing.

Specifying a structure ID requires IXLMG SID system support. If a nonzero structure ID is specified on a system that does not support it, the request is rejected for the nonzero value (IxIRsnCodeReservedNot0). Macro IXCYQUAA defines the QuReqRfIxlmgSid bit in the QuReqFeatures string that can be used to test for IXLMG SID support. Use IXCQUERY REQINFO=FEATURES to get the QuReqFeatures string.

**,STRCOPYCNTLS=NO****,STRCOPYCNTLS=YES**

Use this input parameter to specify whether the structure copy controls record should be included in the measurement data that is associated with the structure records. Note that this information is intended for system use only, and therefore no mapping is provided for the structure copy controls data.

**NO**

Structure copy controls should not be included in the measurement data.

**YES**

Structure copy controls should be included in the measurement data.

Specifying STRCOPYCNTLS=YES requires an AMDALEVEL of 1 or higher.

**,STRNAME=strname**

Use this input parameter to specify the name of the coupling facility structure for which data is to be gathered. When the STRNAME keyword is specified and the structure has active XES connectors from the system on which IXLMG was invoked, data will be gathered which includes both software usage information kept on the z/OS system and hardware structure control information kept in the coupling facility.

If the structure has no active XES connectors from the system on which IXLMG was invoked, only hardware structure control information kept in the coupling facility will be gathered. If the structure specified is in the process of rebuild, IXLMG returns information for both structures.

**To Code:** Specify the RS-type name or address (by using a register from 2 to 12) of the 16-character field that contains the name of the coupling facility structure for which data is to be gathered.



## ABEND Codes

Abend X'026' (See [z/OS MVS System Codes](#) for more information on this abend.)

## Return and Reason Codes

When the system returns control to your program:

- GPR 15 (and *retcode*, if you coded RETCODE) contains a return code.
- GPR 0 (and *rsncode*, if you coded RSNCODE) contains a reason code.

Table 85. Return and Reason Codes for the IXLMG Macro

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0	None	<b>Meaning:</b> IXLMG completed successfully and returned the requested information in the data area provided. <b>Action:</b> None.
4	xxxx0404	<b>Equate Symbol:</b> IXLRNCODEMOREDATA <b>Meaning:</b> Not all requested data could be returned in the data area provided because the data area was not large enough. <b>Action:</b> Retry the request with a larger data area. The required size is provided in the IXLYAMDA header in the field IXLYAMDAREA_TLEN.
4	xxxx0421	<b>Equate Symbol:</b> IXLRNCODESTGCLASSERR <b>Meaning:</b> The requested data could not be returned for the storage class information. The STGCLASS specified is a storage class that exceeds the maximum defined storage class for the structure. The maximum defined storage class is defined on the IXLCONN for initial structure allocation. <b>Action:</b> Retry the request with the correct value.
4	xxxx0422	<b>Equate Symbol:</b> IXLRNCODECOCLASSERR <b>Meaning:</b> Not all requested data could be returned for the cast-out class information. Either the COCLASSE was larger than the maximum cast-out class, or the COCLASSB was larger than the maximum cast-out class for the structure. The start and end of range values returned indicates the amount of data reported. The cast-out class values are set on the IXLCONN invocation. <b>Action:</b> Retry the request with correct start and end values.
4	xxxx0423	<b>Equate Symbol:</b> IXLRNCODESTRUCTUREERR <b>Meaning:</b> No data could be returned for the structure specified. The name is not for a known structure. <b>Action:</b> Retry the request with the correct structure name.
4	xxxx0426	<b>Equate Symbol:</b> IXLRNCODESTRUCTUREFAIL <b>Meaning:</b> No data could be returned for the structure specified. The structure is in the failed state. <b>Action:</b> None necessary. IXLMG should only be called for valid structures. If this is unexpected, check your protocol to determine why you did not know about the problem with this structure.

Table 85. Return and Reason Codes for the IXLMG Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0801	<p><b>Equate Symbol:</b> IXLRNCODEBADPARMLIST</p> <p><b>Meaning:</b> Program error. The parameter list for this request is not addressable. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b> Verify that:</p> <ul style="list-style-type: none"> <li>• The parameter list address is uncorrupted.</li> <li>• The parameter list is addressable in the caller's primary address space.</li> <li>• If you are issuing this macro while disabled, the parameter list resides in either page-fixed or DREF storage.</li> <li>• If you are issuing this macro in AR-mode and you specified the parameter list address using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If you are issuing this macro in AR-mode, SYSSTATE ASCENV=AR must be issued before issuing this macro.</li> <li>• The parameter list resides in storage with a key that is consistent with the PSW key of the IXLMG invoker.</li> </ul>
8	xxxx0802	<p><b>Equate Symbol:</b> IXLRNCODEBADPARMLISTALET</p> <p><b>Meaning:</b> The ALET for the parameter list is not valid.</p> <p><b>Action:</b> All parameters must be in the primary address space so all ALETs should be set up accordingly.</p>
8	xxxx0803	<p><b>Equate Symbol:</b> IXLRNCODERESERVEDNOT0</p> <p><b>Meaning:</b> Reserved field in parameter list is not zero.</p> <p><b>Action:</b> Check to see if your program overlaid the parameter list. Check the parameter list address to make sure it is uncorrupted. Make sure you are running with the same version of MVS that the macro was compiled with.</p>
8	xxxx080D	<p><b>Equate Symbol:</b> IXLRNCODEAREATOOSMALL</p> <p><b>Meaning:</b> DATAAREA is too small to contain the minimum response information. (The minimum response information is the IXLYAMDA header information mapped by IXLYAMDAREA.)</p> <p><b>Action:</b> Run the request again and specify a DATAAREA with a minimum DATALEN of IXLYAMDAREA_LENGTH.</p>
8	xxxx080E	<p><b>Equate Symbol:</b> IXLRNCODEBADAREA</p> <p><b>Meaning:</b> Error accessing DATAAREA.</p> <p><b>Action:</b> Verify the following:</p> <ul style="list-style-type: none"> <li>• The DATAAREA specified is addressable in the primary address space.</li> <li>• The DATAAREA is in fixed or disabled reference storage.</li> <li>• The DATAAREA storage key is consistent with the PSW key of the IXLMG invoker.</li> </ul>
8	xxxx080F	<p><b>Equate Symbol:</b> IXLRNCODEBADAREAALET</p> <p><b>Meaning:</b> DATAAREA ALET is not valid.</p> <p><b>Action:</b> All parameters must be in the primary address space so all ALETs should be set up accordingly.</p>

Table 85. Return and Reason Codes for the IXLMG Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0885	<b>Equate Symbol:</b> IXLRNSCODEBADAMDALEVEL <b>Meaning:</b> The value specified for AMDALEVEL is not valid. <b>Action:</b> Retry the request with the correct AMDALEVEL value.
C	FFFFFFFF	<b>Equate Symbol:</b> IXLRNSCODENOTAVAILABLE <b>Meaning:</b> Environment error. There are no coupling facility services available. The hardware support necessary to provide coupling facility services might not be present. <b>Action:</b> XES may not be used when XCF is running in local mode. XES requires that the coupling facility be installed.
10	xxxx10xx	<b>Meaning:</b> Software failure in XES software processing. <b>Action:</b> Contact the IBM support center.



## Chapter 80. IXLPURGE — Purge Operations to a Coupling Facility

### Description

The IXLPURGE macro allows an authorized caller to complete operations in progress to a coupling facility and purge operations not yet processed. The operations include requests initiated by the IXLCACHE, IXLLIST, and IXLRT services.

### Environment

The following are the environment requirements for the caller.

Environment	Environment requirement
Minimum authorization:	Supervisor state and PKM allowing key 0 -
Dispatchable unit mode:	Task or SRB
Cross memory mode:	Any PASN, any HASN, any SASN
AMODE:	31-bit
ASC mode:	Primary or Access Register (AR)
Interrupt status:	Enabled for I/O and external interrupts
Locks:	No locks held
Control parameters:	Must be in the primary address space

### Programming Requirements

If the program is in AR mode, issue SYSSTATE ASCENV=AR before invoking IXLPURGE. SYSSTATE ASCENV=AR tells the system to generate code appropriate for AR mode.

### Restrictions

None.

### Input Register Information

Before issuing the IXLPURGE macro, the caller does not have to place any information into any register unless using it in register notation for a particular parameter, or using it as a base register.

### Output Register Information

When control returns to the caller of the IXLPURGE macro, the general purpose registers (GPRs) contain:

#### Register Contents

**0**

Reason code, if applicable, if GPR15 return code is non-zero

**1**

Used as work register by the system

**2-13**

Unchanged

**14**

Used as work register by the system

**15**

Return code

When control returns to the caller of the IXLPURGE macro, the access registers (ARs) contain:

**Register  
Contents**

**0-1**

Used as work registers by the system

**2-13**

Unchanged

**14-15**

Used as work registers by the system

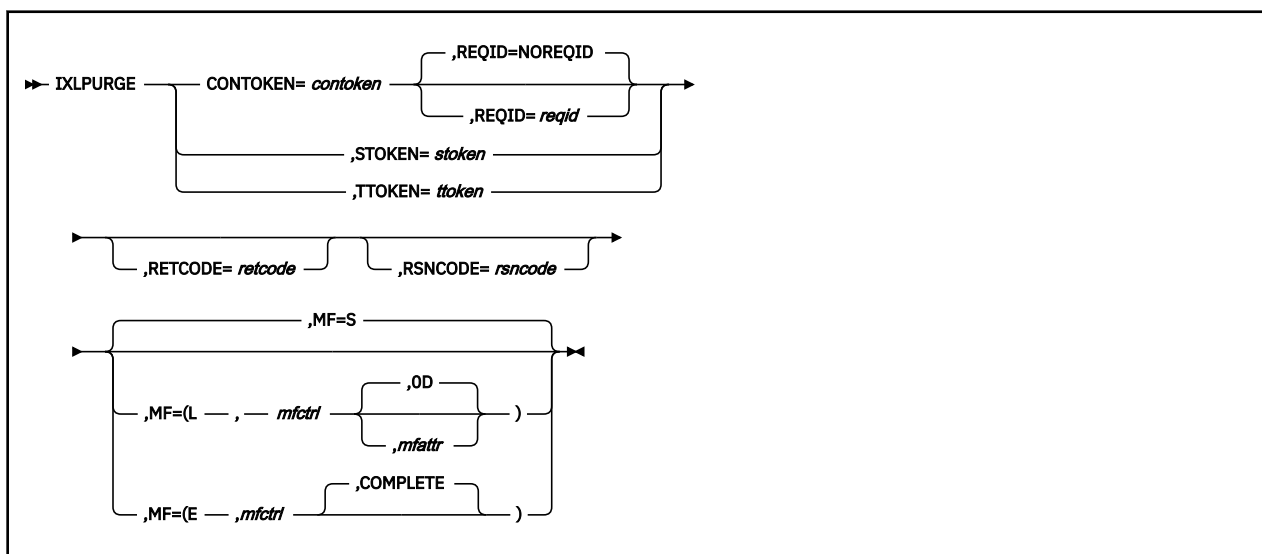
**For registers that the system changes**, a caller who depends on these registers containing the same value before and after issuing the macro must save these registers before issuing the macro and restore them after the system returns control. For asynchronous processing of requests, see [z/OS MVS Programming: Sysplex Services Guide](#).

## Performance Implications

None.

## Syntax Diagram

The syntax of the IXLPURGE macro is as follows:



## Parameter Descriptions

The parameter descriptions are listed in alphabetical order. Default values are underlined:

**CONTOKEN=contoken**

Use this input parameter to specify the connect token that identifies the connector this request is to be processed for. All requests for that connection will be purged. (The connect token is in the IXLCONA answer area, after the IXLCONN is issued, in the CONACONTOKEN field.)

**To Code:** Specify the name or address (using a register from 2 - 12) of the 16-character storage area that contains the connector token.

**,MF=S**  
**,MF=(L,mfctrl)**  
**,MF=(L,mfctrl,mfattr)**  
**,MF=(L,mfctrl,0D)**  
**,MF=(M,mfctrl)**  
**,MF=(M,mfctrl,COMPLETE)**  
**,MF=(M,mfctrl,NOCHECK)**  
**,MF=(E,mfctrl)**  
**,MF=(E,mfctrl,COMPLETE)**  
**,MF=(E,mfctrl,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

**,mfctrl**

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

**,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

**,COMPLETE**

**,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

**COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if SMILE=*var* were an optional parameter and the default is SMILE=NO\_SMILE then it would not be documented. However, if the default was SMILE=-), then it would be documented because a value would be the default.

**NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

**,REQID=NOREQID**

**,REQID=reqid**

Use this input parameter with the CONTOKEN keyword to identify the request or requests that are associated with the connection to be processed. This parameter corresponds to the REQID parameter

specified on an IXLLIST or IXLCACHE request. Only those requests that were initiated with this REQID and the indicated CONTOKEN will be purged.

**To Code:** Specify the name or address (using a register from 2 - 12) of the 8-character storage area that contains the request identifier.

**STOKEN=stoken**

Use this input parameter to specify the name of the token that identifies the address space for which the request is to process. All requests associated with the address space will be purged.

**To Code:** Specify the name or address (using a register from 2 - 12) of the 8-character storage area that contains the address space token.

**TOKEN=token**

Use this input parameter to specify the name of the token that identifies the task for which the request is to process. All requests associated with the task will be purged.

**To Code:** Specify the name or address (using a register from 2 - 12) of the 16-character storage area that contains the task token.

## ABEND Codes

Abend X'026' (See [z/OS MVS System Codes](#) for more information on this abend.)

## Return and Reason Codes

The following table identifies the hexadecimal return and reason codes and the equate symbol associated with each reason code.

Table 86. Return and Reason Codes for the IXLPURGE Macro		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0	None	<b>Meaning:</b> Purge is successful. <b>Action:</b> None.
8	xxxx0801	<b>Equate Symbol:</b> IXLRNCODEBADPARMLIST <b>Meaning:</b> Program error. The parameter list for this request is not addressable. The answer area (ANSAREA) fields are not valid. <b>Action:</b> Verify that: <ul style="list-style-type: none"> <li>• The parameter list address is uncorrupted.</li> <li>• The parameter list is addressable in the caller's primary address space.</li> <li>• If you are issuing this macro while disabled, the parameter list resides in either page-fixed or DREF storage.</li> <li>• If you are issuing this macro in AR-mode and you specified the parameter list address using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If you are issuing this macro in AR-mode, SYSSTATE ASCENV=AR must be issued before issuing this macro.</li> </ul>



Table 86. Return and Reason Codes for the IXLPURGE Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0804	<p><b>Equate Symbol:</b> IXLRSNCODEBADVERSIONNUM (or IXLRSNCODEBADVERSION#)</p> <p><b>Meaning:</b> The version number in the macro parameter list is not compatible with the level of XES currently being used. The answer area (ANSAREA) fields are not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that your program did not overlay the parameter list storage.</li> <li>• Verify that your program was assembled with the correct macro library for the release of MVS on which your program is running.</li> </ul>



## Chapter 81. IXLREBLD – Rebuilding or duplexing a structure

### Description

Use the IXLREBLD macro to control the rebuild process for a coupling facility structure. There are two types of rebuild processing, rebuild and duplexing rebuild. The method by which the rebuild processing is accomplished can be either user-managed or system-managed.

- User-managed rebuild allows you to allocate another structure with the same name in the same or a different coupling facility for the purpose of planned reconfiguration and recovery. In a structure rebuild, the user is responsible for reconstructing data in the newly allocated structure. Rebuild allows you to change the location or attributes of a structure.
- User-managed duplexing rebuild allows you to allocate another cache structure in a different coupling facility for the purpose of duplexing the data in the structure to achieve better availability and usability. In a user-managed duplexing rebuild, the user is responsible for managing both instances of the structure and when necessary, can revert to using only one of the structures. User-managed duplexing rebuild is valid only for cache structures.
- System-managed rebuild allows the system to provide the support necessary for rebuild processing. Connectors recognize that the structure is temporarily unavailable for requests and are not active participants in the rebuild process. System-managed rebuild requires that active connectors have specified IXLCONN ALLOWAUTO=YES. It is also recommended that ALLOWALTER=YES be specified when connecting to the structure.

System-managed rebuild provides the capability for persistent structures with no connectors or with only failed-persistent connectors to be rebuilt.

System-managed rebuild is valid for all types of structures; however, if a user-managed rebuild implementation is already in place, the system will give precedence to that method of rebuild.

- System-managed duplexing rebuild allows the system to allocate another structure in a different coupling facility for the purpose of duplexing the data in the structure. System-managed duplexing rebuild provides a recovery mechanism by which rapid failover to the unaffected structure instance of a duplexed pair can occur in the event of a failure. System-managed duplexing rebuild does not require connectors to be participants in the duplexing process, although the connectors recognize that the structure is temporarily unavailable for requests, but does require that connectors be aware of its progress. System-managed duplexing rebuild is valid for all types of structures; however, if a user-managed duplexing rebuild implementation is already in place, the system will give precedence to that method of duplexing.

See *z/OS MVS Programming: Sysplex Services Guide* for a complete description of rebuild processing and its requirements.

Phases of the user-managed rebuild process are reported to the event exits of the users coordinating the structure rebuild. IXLREBLD requestors are not required to be connected to a structure to start or stop a structure rebuild. However, only an active connector to the structure can issue the IXLREBLD request to indicate that the rebuild process is complete.

Phases of the system-managed rebuild process are managed by the system on behalf of the connector. System-managed rebuild does not require that there be any active connectors to the structure.

You can use IXLREBLD to do the following processes:

- Start rebuilding.
- Stop rebuilding.
- Indicate that you have completed rebuild processing. (User-managed rebuild only)

- Start duplexing rebuild
- Stop duplexing rebuild
- Indicate that you have completed duplexing rebuild processing. (User-managed duplexing rebuild only)
- Report progress during structure repopulation (user-managed only).

For starting and stopping a rebuild process, you must provide a reason for the operation. The reason is passed to the event exits of the users participating in the rebuild process.

You initiate the rebuild process by issuing an IXLREBLD REQUEST=START . . . request for either an individual structure or for all structures in a particular coupling facility.

You initiate the duplexing rebuild process by issuing an IXLREBLD REQUEST=STARTDUPLEX . . . request for either an individual structure or for all structures in a particular coupling facility. Depending on the method of duplexing, if any, supported by each structure, the system will start either user-managed or system-managed duplexing rebuild appropriate for each structure. User-managed duplexing rebuild supports only cache structures; system-managed duplexing rebuild supports all structure types.

Another use for the IXLREBLD macro is to populate a coupling facility with a set of appropriate structures after the coupling facility has been removed from the configuration for service and then returned to the sysplex configuration. The POPULATECF function, available with OS/390 Release 6 and higher, provides a means of redistributing multiple structures in a coupling facility with a single IXLREBLD macro invocation or SETXCF command. The advantage of the POPULATECF function is that it rebuilds structures that are currently allocated in a coupling facility into the coupling facility that the structure has specified (in the CFRM policy) is preferable. If the structure already resides in a coupling facility that is higher in its preference list than the coupling facility being populated, the system will not rebuild the structure. See *z/OS MVS Setting Up a Sysplex* for a description of how to use the POPULATE function.

The security administrator may have controlled the use of installation structures through the use of RACF or another security product. If so, ensure that you are authorized to issue the IXLREBLD macro for the structure.

For more information about the structure rebuild process, see *z/OS MVS Programming: Sysplex Services Guide*.

## Environment

The following are the environment requirements for the caller.

Environment	Environment requirement
Authorization:	Supervisor state or PKM keys 0 - 7
Dispatchable unit mode:	Task,
Cross memory mode:	PASN=HASN, any SASN. For IXLREBLD REQUEST=COMPLETE, DUPLEXCOMPLETE, POPULATING, or WAITING, the primary address space must be equal to the requestor's primary address space at the time of the connection.
AMODE:	31 bit.
ASC mode:	Primary or access register (AR)
Interrupt status:	Enabled for I/O and external interrupts
Locks:	No locks held, with no enabled, unlocked task (EUT) FRRs established
Control parameters:	Must be in the primary address space or be in an address/data space that is addressable through a public entry on the caller's dispatchable unit access list (DU-AL)

## Programming Requirements

---

If the program is in AR mode, issue the SYSSTATE ASCENV=AR macro before IXLREBLD. SYSSTATE ASCENV=AR tells the system to generate code appropriate for AR mode.

## Restrictions

---

- If the caller is running in AR ASC mode and specifies a parameter using explicit register notation, the access register corresponding to the general register must appropriately qualify the general register.
- The virtual storage areas specified by parameters can be addressable in the caller's primary, secondary, or home address spaces, from the PASN access list, or from the dispatchable unit access list (DU-AL).

## Input Register Information

---

Before issuing the IXLREBLD macro, the caller does not have to place any information into any register unless using it in register notation for a particular parameter, or using it as a base register.

## Output Register Information

---

When control returns to the caller, the general purpose registers (GPRs) contain:

### Register Contents

- 0**  
Reason code, if applicable, if GPR 15 return code is nonzero
- 1**  
Used as a work register by the system
- 2-13**  
Unchanged
- 14**  
Used as work register by the system
- 15**  
Return code

When control returns to the caller of the IXLREBLD macro, the access registers (ARs) contain:

### Register Contents

- 0-1**  
Used as work registers by the system
- 2-13**  
Unchanged
- 14-15**  
Used as work registers by the system

**For registers that the system changes**, a caller who depends on these registers containing the same value before and after issuing the macro must save these registers before issuing the macro and restore them after the system returns control.

## Performance Implications

---

The rebuild process entails coordination among active connections to achieve the reconstruction of the structure. The performance of IXLREBLD will depend on the size of the structure, the number of connections to the structure, and the amount of data to be reconstructed. Also, depending on the protocols of connections during IXLREBLD, the services provided by those connections may be temporarily unavailable.

IXLREBLD processing might involve referencing or updating the CFRM active policy to reflect the operation that has been requested. When invoking this macro, be aware of the I/O to the CFRM couple data set that might be generated.

## Understanding IXLREBLD version support

---

The IXLREBLD macro supports versions 0, 1, 2, and 3.

- Keywords not specifically noted here are supported by all versions starting with version 0 and higher of the IXLREBLD macro.
- The following keyword is supported by all versions starting with version 1 and higher of the IXLREBLD macro.
  - LESSCONNECTION
- The following keyword is supported by all versions starting with version 2 and higher of the IXLREBLD macro.
  - POPULATECF
- The following keywords are supported by version 3 and subsequent versions of the IXLREBLD macro.
  - USERTEXT
  - WAITFORCONNAME
  - WAITFORCONTOKEN
  - WAITFORSYSNAME

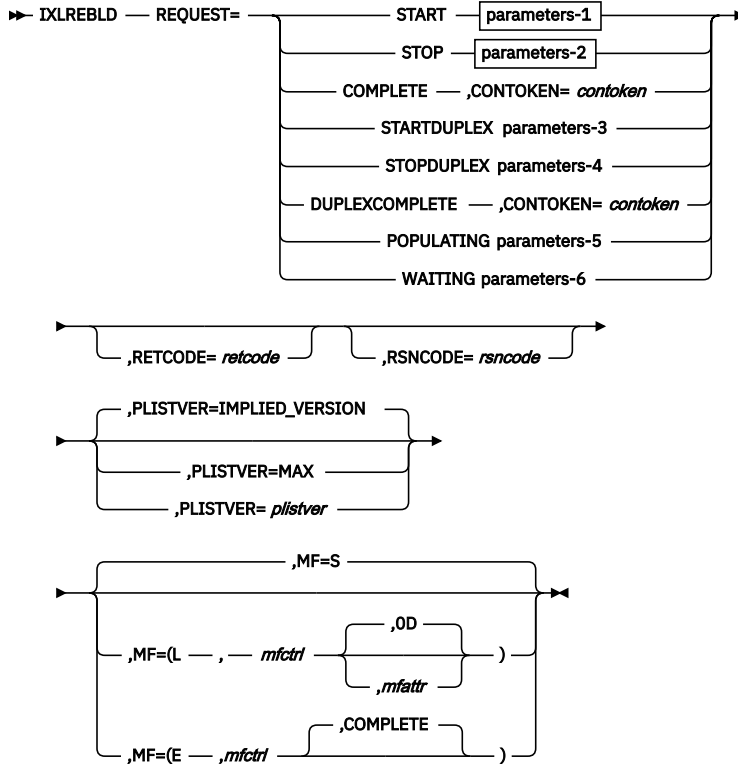
Specify the version of the parameter list that you want generated with the PLISTVER keyword. See Chapter 2, “[Specifying a Macro Version Number](#),” on page 5 for considerations when specifying the version of the parameter list with PLISTVER.

## Syntax diagram

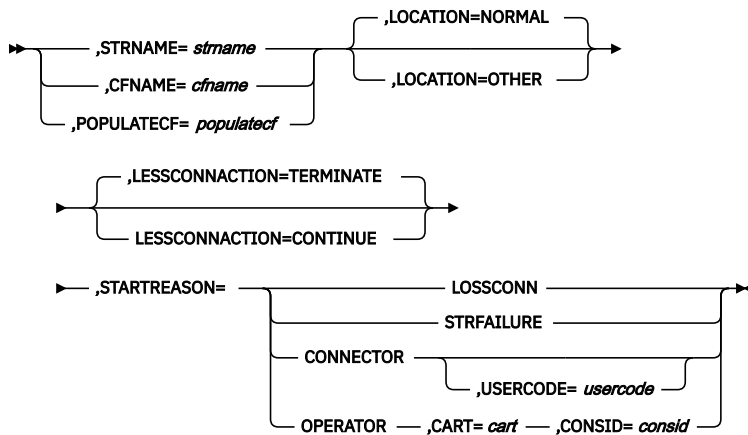
---

The syntax of the IXLREBLD macro is as follows:

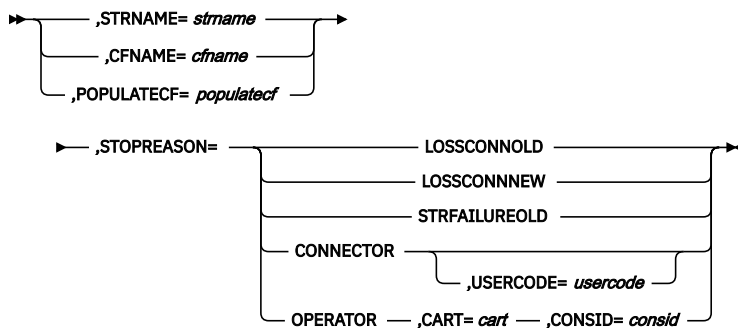
## main diagram

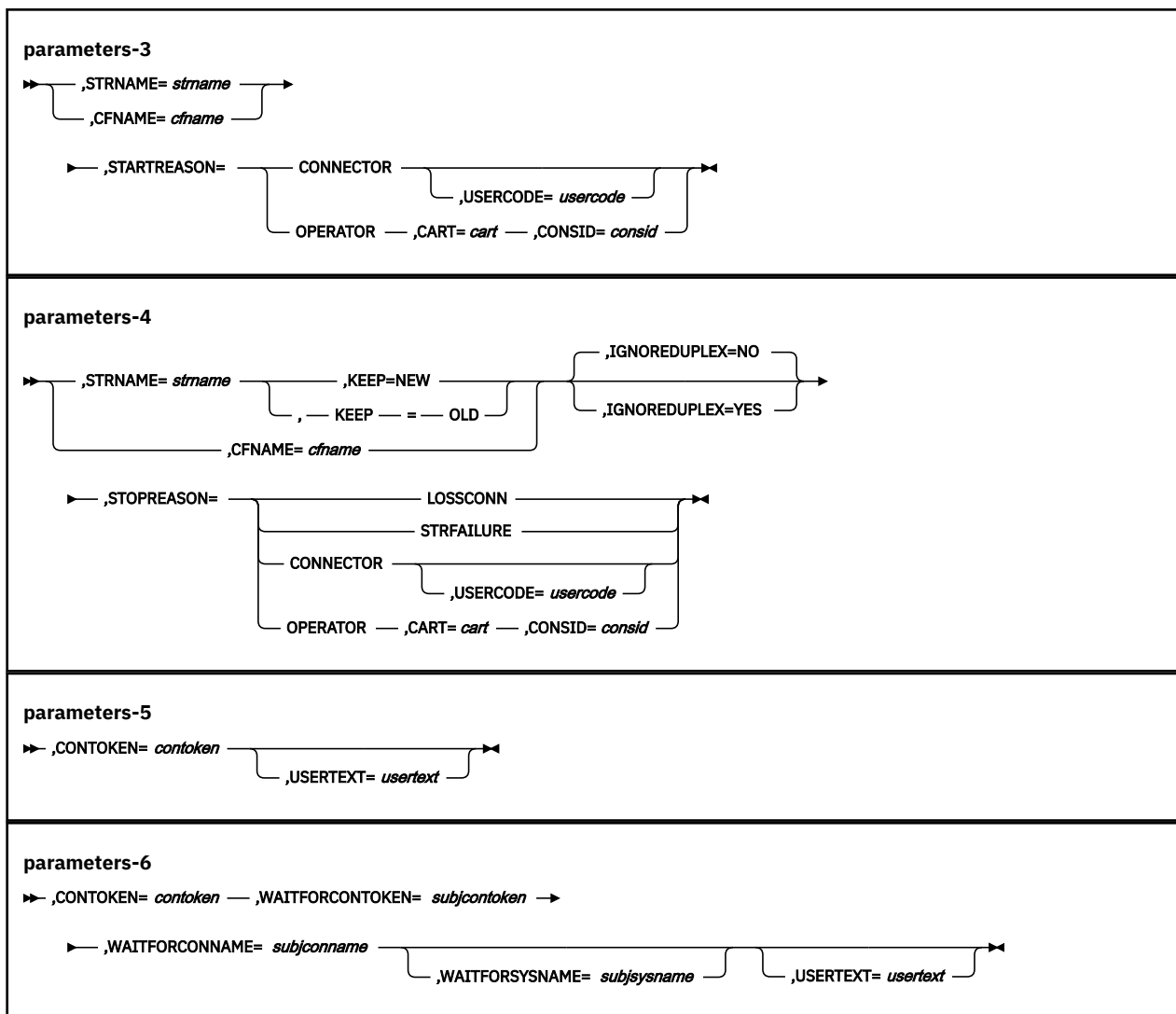


## parameters-1



## parameters-2





## Parameter descriptions

The parameter descriptions are listed in alphabetical order. Default values are underlined:

### **,CART=cart**

Use this input parameter to specify command-and-response token (CART) associated with the console to be notified when any rebuild- or duplexing rebuild-related start or completion occurs. The system reports a Rebuild Quiesce event with OPERATOR as the reason and places the CART value in the event exit parameter list IXLVEEPL (field EEPLCART) for each connected user.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the eight-character input field that contains the command-and-response token (CART).

### **,CFNAME=cfname**

Use this input parameter to specify the name of a coupling facility in which structure rebuild is requested for every structure allocated other than XCF signalling structures. Note that this function is intended only for use by a cleanup utility. The system will not start rebuild for XCF signalling structures in the named coupling facility. The user should rebuild XCF signalling structures in a coupling facility one at a time.

The coupling facility name must be 8 characters long, padded on the right with blanks if necessary. The name may contain numeric characters, uppercase alphabetic characters, national characters (\$, @, #), or an underscore (\_), and must begin with an uppercase alphabetic character.



**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the 8-character input field that contains the coupling facility name.

**,CONSID=consid**

Use this input parameter to specify the system-generated console ID for the console or extended MCS console to be notified when any rebuild- or duplexing rebuild-related start or completion occurs.

The system reports a Rebuild Quiesce event with OPERATOR as the reason, and places the console id of the issuing console in the event exit parameter list IXLYEEPL (field EEPLCONSID) for each connected user.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the fullword input field that contains the console ID.

**,CONTOKEN=contoken**

Use this input parameter to specify the original connect token returned to the requestor by the IXLCONN macro when connecting to the structure.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the 16-character input field that contains the connect token.

**,IGNOREDUPLEX=NO**

**,IGNOREDUPLEX=YES**

Use this input parameter to specify whether to use the completion of the processing initiated by this request as a trigger to reduplex the structure. A specification of DUPLEX(ENABLED) in the CFRM active policy means that the system might automatically duplex the structure in response to various triggers.

**IGNOREDUPLEX=NO**

Use the completion of the processing that is initiated by this request as a trigger to reduplex the structure.

**IGNOREDUPLEX=YES**

Do not use the completion of the processing that is initiated by this request as a trigger to reduplex the structure. This request does not prevent a duplexing rebuild from being started at a later time after the processing completes.

To prevent the system from reduplexing the structure at a later time, change the DUPLEX specification for the structure to DUPLEX(ALLOWED) before stopping duplexing, or change the DUPLEX setting for the structure to DUPLEX(DISABLED), which will cause XCF to initiate the stop processing. Change the DUPLEX setting back to DUPLEX(ENABLED) when you no longer need to prevent the system from reduplexing.

**,KEEP=NEW**

**,KEEP=OLD**

Use this input parameter to specify which structure should remain after the duplexing rebuild has been stopped.

**KEEP=NEW**

Duplexing rebuild should stop and switch to using the new structure.

**KEEP=OLD**

Duplexing rebuild should stop to fall back to using the old structure.

**,LESSCONNACTION=TERMINATE**

**,LESSCONNACTION=CONTINUE**

Use this input parameter to indicate whether XES is to allow a rebuild to continue, in spite of a degradation in connectivity to the new structure.

When a structure rebuild is requested, XES evaluates whether the new structure will have the same or better connectivity than the current structure, as determined by the Sysplex Failure Management (SFM) weights of the connector's systems attached to the coupling facility. If the evaluation shows that the new structure will have poorer connectivity than the current structure, XES uses the LESSCONNACTION specification to determine what action to take in regard to the structure rebuild.

**Note:**

1. If the IXLREBLD invocation specifies REASON=LOSSCONN, XES ignores the LESSCONNACTION keyword and processes the request as if LESSCONNACTION=TERMINATE were specified.
2. For duplexing rebuild requests, LESSCONNACTION=TERMINATE is assumed; the system will not initiate a duplexing rebuild if the new structure has poorer connectivity than the old structure, as determined by the SFM weights of the systems connected to the structure in the coupling facility.
3. XES ignores the LESSCONNACTION keyword when a rebuild start request results in system-managed processing. If any active connector loses connectivity to the structure being rebuilt, the system-managed process always stops.

**LESSCONNACTION=TERMINATE**

The system is to stop the rebuild if the new structure has poorer connectivity than the old structure, as determined by the SFM weights of the systems connected to the structure in the coupling facility.

**LESSCONNACTION=CONTINUE**

The system is not to stop the structure rebuild. Users that cannot connect to the new structure must disconnect from the structure to allow the rebuild to continue, or must stop the rebuild.

**Note:**

1. If LESSCONNACTION is not specified when the rebuild is started, the default action is to not to continue any rebuild, despite its effect on the connectors to the structure.
2. If LESSCONNACTION=CONTINUE is specified with the CFNAME= keyword, all structures in the specified coupling facility are rebuilt, despite any degradation in connectivity for the structures being rebuilt.

**,LOCATION=NORMAL****,LOCATION=OTHER**

Use this input parameter to indicate where the new structure that is allocated for rebuild can be located.

**NORMAL**

The new structure can be allocated in any coupling facility in the preference list using the normal allocation rules.

**OTHER**

The new structure cannot be allocated in the same coupling facility as the original structure. The new structure can be allocated in any coupling facility in the preference list, other than the coupling facility containing the original structure, using normal allocation rules. LOCATION=OTHER is assumed when the rebuild is a duplexing rebuild.

This option is intended for planned reconfiguration cases where a system administrator has verified that the policy does contain a coupling facility where the structure can be rebuilt. If there is only one coupling facility in the preference list for the structure being rebuilt and the coupling facility is the location of the original structure, each connector will receive reason code IXLSNOCODENOFAC when the rebuild connect is attempted. In IXLYCONA, CONAFACILITYARRAY will indicate that the current coupling facility was not chosen because the LOCATION=OTHER option was specified on IXLREBLD.

**Note:** A structure rebuild initiated with STARTREASON=LOSSCONN will always be stopped if the connectivity will not be improved for the current set of active connectors by the rebuild process. LOCATION=OTHER is NOT implied by a LOSSCONN rebuild, as there may not be another viable coupling facility available in which to rebuild the structure.

**,MF=S**  
**,MF=(L,mfctrl)**  
**,MF=(L,mfctrl,mfattr)**  
**,MF=(L,mfctrl,0D)**  
**,MF=(M,mfctrl)**  
**,MF=(M,mfctrl,COMPLETE)**  
**,MF=(M,mfctrl,NOCHECK)**  
**,MF=(E,mfctrl)**  
**,MF=(E,mfctrl,COMPLETE)**  
**,MF=(E,mfctrl,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

#### **,mfctrl**

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

#### **,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

#### **,COMPLETE**

#### **,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

#### **COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if SMILE=*var* were an optional parameter and the default is SMILE=NO\_SMILE then it would not be documented. However, if the default was SMILE=:), then it would be documented because a value would be the default.

#### **NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

#### **,PLISTVER=IMPLIED\_VERSION**

#### **,PLISTVER=MAX**

#### **,PLISTVER=plistver**

Use this input parameter to specify the version of the macro. See [“Understanding IXLREBLD version support”](#) on page 1542 for a description of the options available with PLISTVER.

**,POPULATECF=populatecf**

Use this input parameter to specify the name of the coupling facility which is to be populated with structures selected from those currently allocated in other less preferred coupling facilities.

A structure rebuild will be attempted for each allocated structure in the policy that contains the specified coupling facility in its preference list, if the specified coupling facility is at a higher position in the preference list than the coupling facility in which the structure currently is allocated. If the structure already is allocated in a more preferable coupling facility, the rebuild is not started.

Each selected structure will be placed in a 'pending rebuild' state; these 'pending rebuild' structures are then processed serially to completion (either a stop complete event or a rebuild complete event is received) before the system selects the next structure for rebuild.

This option is intended for use by an operations product or subsystem only. It is not intended for use by an application initiating rebuilds for structures owned by other applications.

The coupling facility name must be 8 characters long, padded on the right with blanks if necessary. The name may contain numeric characters, uppercase alphabetic characters, national characters (\$, @, #), or an underscore (\_), and must begin with an uppercase alphabetic character.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the 8-character field that contains the name of the coupling facility.

**REQUEST=START**

**REQUEST=STOP**

**REQUEST=COMPLETE**

**REQUEST=STARTDUPLEX**

**REQUEST=STOPDUPLEX**

**REQUEST=DUPLEXCOMPLETE**

**REQUEST=POPULATING**

**REQUEST=WAITING**

Use this input parameter to specify the type of rebuild request.

- Use REQUEST=START to request start rebuild processing for the specified structure or structures within the specified coupling facility. The request will start non-duplexing rebuild processing, which might be either user-managed or system-managed.
- Use REQUEST=STOP to request stop rebuild processing for the specified structure or structures within the specified coupling facility. The request will stop non-duplexing rebuild processing only, either for a user-managed or system-managed rebuild.
- Use REQUEST=COMPLETE to indicate that user-managed rebuild processing is complete for the specified connection.
- Use REQUEST=STARTDUPLEX to request start duplexing rebuild processing for the specified structure or structures within the specified coupling facility.
- Use REQUEST=STOPDUPLEX to request stop duplexing rebuild processing for the specified structure or structures within the specified coupling facility. The request implies that the connector is reverting to simplex mode by either switching to the new structure or falling back to the old structure.
- Use REQUEST=DUPLEXCOMPLETE to indicate that duplexing rebuild processing is complete for the specified connection.
- Use REQUEST=POPULATING to indicate that the connector is making satisfactory progress toward structure repopulation during user-managed rebuild or duplexing rebuild. Repopulation refers to the establishment of exploiter data in the rebuild new structure instance in the interval after IXLCONN REQTYPE=REBUILDCONNECT and before IXLREBLD REQUEST=COMPLETE. It is also acceptable to use REQUEST=POPULATING when the connector is waiting for another (master) connector to repopulate the rebuild new structure instance, even if the connector has no work of its own to do. REQUEST=POPULATING is valid when the connector previously specified IXLCONN REQTYPE=REBUILDCONNECT MONITOR=IXLREBLD.

Each connector that is actively repopulating the structure or is using REQUEST=POPULATING (rather than REQUEST=WAITING) while waiting for some other connector to do so should invoke

IXLREBLD REQUEST=POPULATING at intervals not to exceed 2 minutes as long as repopulation is making satisfactory progress, as defined by the application.

**Notes:**

- Connectors responsible for repopulating the structure must not simply initiate periodic POPULATING requests to avoid declaration of a hang by z/OS hang detection processing. Bypassing hang detection in this manner can expose the entire sysplex to delays and loss of function.
- To ensure that the IXLREBLD POPULATING support is installed on the system on which you are running, issue IXCQUERY REQINFO=FEATURES. QuReqRfRepopulateProgress, returned from the IXCQUERY request, indicates whether the support is installed.
- Use REQUEST=WAITING to indicate that the connector is waiting for another (master) connector to repopulate the rebuild new structure instance. REQUEST=WAITING is valid when the connector previously specified IXLCONN REQTYPE=REBUILDCONNECT MONITOR=IXLREBLD. A connector that is waiting need only invoke IXLREBLD REQUEST=WAITING once within 2 minutes of the start of the repopulation phase, or if a new master connector is assigned.

**Note:** To ensure that the IXLREBLD WAITING support is installed on the system on which you are running, issue IXCQUERY REQINFO=FEATURES. QuReqRfRepopulateProgress, returned from the IXCQUERY request, indicates whether the support is installed.

**,RETCODE=retcode**

Use this output parameter to specify the location in which the system is to copy the return code from GPR 15.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the fullword output field to contain the return code.

**,RSNCODE=rsncode**

Use this output parameter to specify the location in which the system is to copy the reason code from GPR 0.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the fullword output field to contain the reason code.

**,STARTREASON=LOSSCONN**

**,STARTREASON=STRFAILURE**

**,STARTREASON=CONNECTOR**

**,STARTREASON=OPERATOR**

Use this input keyword to specify the reason for starting the rebuild or duplexing rebuild process for the structure.

**STARTREASON=LOSSCONN**

Loss of connectivity to the structure (for rebuild processing only)

**STARTREASON=STRFAILURE**

Structure failure (for rebuild processing only)

**STARTREASON=CONNECTOR**

Connected user-specified reason. The requestor can specify the USERCODE parameter to indicate the reason for initiating the rebuild.

**STARTREASON=OPERATOR**

Operator-initiated rebuild processing. This option requires that you also provide a command-and-response token (CART) and console ID (CONSID) associated with the console to be notified of the rebuild progress.

**,STOPREASON=LOSSCONNOLD**  
**,STOPREASON=LOSSCONNNEW**  
**,STOPREASON=LOSSCONN**  
**,STOPREASON=STRFAILUREOLD**  
**,STOPREASON=STRFAILURE**  
**,STOPREASON=CONNECTOR**  
**,STOPREASON=OPERATOR,CART=*cart*,CONSID=*consid***

Use this input parameter to specify the reason for stopping the rebuild or the duplexing rebuild process for the structure. You can specify one of the following reasons:

**STOPREASON=LOSSCONNOLD**

Loss of connectivity to the original structure (for rebuild processing only)

**STOPREASON=LOSSCONNNEW**

Loss of connectivity to the new structure (for rebuild processing only)

**STOPREASON=LOSSCONN**

Loss of connectivity to the structure (for duplexing rebuild processing only)

**STOPREASON=STRFAILUREOLD**

Failure of the original structure (for rebuild processing only)

**STOPREASON=STRFAILURE**

Failure of the structure (for duplexing rebuild processing only)

**STOPREASON=CONNECTOR**

Connected user-specified reason. The requestor can specify the USERCODE parameter to indicate the reason for stopping the rebuild.

**STOPREASON=OPERATOR**

Operator-initiated request to stop rebuild processing. This option requires that you also provide a command-and-response token (CART) and console ID (CONSID) associated with the console to be notified of the rebuild processing.

**,STRNAME=*strname***

Use this input parameter to specify the name of the structure. The structure name must be 16 characters long, padded on the right with blanks if necessary. The name may contain numeric characters, uppercase alphabetic characters, national characters (\$, @, #), or an underscore (\_).

Users can use the IXCQUERY service macro to determine what structures have been defined in the active CFRM policy.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the 16-character input field that contains the name of the target structure.

**,USERCODE=0**

**,USERCODE=*usercode***

Use this input parameter to specify the user code that represents the connector's reason for initiating rebuild processing.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the fullword input field that contains the user code representing the connector's reason for rebuilding.

**,USERTEXT=*usertext***

Use this input parameter to supply optional explanatory text to be included in message IXL057I in the event of a delay in completing the repopulation phase of the rebuild or duplexing rebuild. For example, you can supply information about progress toward completion, or factors affecting the duration of the repopulation phase.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the 70-character field that contains the text to be included in the message.

**,WAITFORCONNAME=*subjconname***

Use this input parameter to specify the name of the master connector responsible for repopulating the rebuild new structure instance.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the 16-character input field that contains the connector name.

**,WAITFORCONTOKEN=subjcontoken**

Use this input parameter to specify the original connect token identifying the master connector responsible for repopulating the rebuild new structure instance.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the 16-character input field that contains the connect token.

**,WAITFORSYSNAME=subjsysname**

Use this input parameter to specify the name of the system on which the master connector responsible for repopulating the rebuild new structure instance resides. If supplied, it will be included in message IXL057I in the event of a delay in completing the repopulation phase of the rebuild or duplexing rebuild.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the 8-character input field that contains the system name.

## Return and reason codes

When the IXLREBLD macro returns control to your program:

- GPR 15 (and *retcode*, if you coded RETCODE) contains a return code.
- GPR 0 (and *rsncode*, if you coded RSNCODE) contains a reason code, if applicable.

Macro IXLYCON provides equate symbols for the return and reason codes. The equate symbols that are associated with each hexadecimal return code are as follows:

**0**

IXLRETCODEOK

**4**

IXLRETCODEWARNING

**8**

IXLRETCODEPARMERROR

**C**

IXLRETCODEENVERROR

**10**

IXLRETCODECOMPERROR

The following table identifies the hexadecimal return and reason codes and the equate symbol that is associated with each reason code.

Table 87. Return and reason codes for the IXLREBLD macro		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0	None.	<b>Meaning:</b> IXLREBLD request was successful. <b>Action:</b> None.
4	xxxx0415	<b>Equate Symbol:</b> IXLRSNCODEALREADYREBUILDING <b>Meaning:</b> Structure rebuild or duplexing rebuild for the structure has already been initiated. The REBUILD REQUEST=START REQUEST=STARTDUPLEX request is not processed. <b>Action:</b> Examine your protocol for initiating the rebuild process.

Table 87. Return and reason codes for the IXLREBLD macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	xxxx0416	<b>Equate Symbol:</b> IXLRSNCODEALREADYSTOPPING <b>Meaning:</b> Structure rebuild or duplexing rebuild stop has already been initiated. The REBUILD REQUEST=STOP or REQUEST=STOPDUPLEX request is not processed. <b>Action:</b> Examine your protocol for stopping the rebuild process.
4	xxxx041D	<b>Equate Symbol:</b> IXLRSNCODEIGNOREFORREBUILDSTOP <b>Meaning:</b> The system ignores IXLREBLD REQUEST=COMPLETE, DUPLEXCOMPLETE, POPULATING, or WAITING because a rebuild stop or stopduplex request for the structure is in progress. The request fails. <b>Action:</b> Examine your rebuild protocol.
4	xxxx0425	<b>Equate Symbol:</b> IXLRSNCODENOSTRFOUND <b>Meaning:</b> The system did not find any structures eligible for rebuild in the specified coupling facility. The request fails. <b>Action:</b> Verify that the names of the structure and the coupling facility have been specified correctly.
8	xxxx0801	<b>Equate Symbol:</b> IXLRSNCODEBADPARMLIST <b>Meaning:</b> The IXLREBLD parameter list is not accessible. <b>Action:</b> Verify that: <ul style="list-style-type: none"> <li>• The parameter list address is uncorrupted.</li> <li>• The parameter list is addressable in the caller's primary address space.</li> <li>• If you are calling IXLREBLD in AR-mode and you specified the parameter list address using explicit register notation, the corresponding access register was updated appropriately.</li> <li>• If you are calling IXLREBLD in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLREBLD.</li> </ul>
8	xxxx0802	<b>Equate Symbol:</b> IXLRSNCODEBADPARMLISTALET <b>Meaning:</b> The IXLREBLD parameter list ALET is not accessible. <b>Action:</b> Ensure that the ALET is zero or that the ALET represents a valid entry on the DU-AL.
8	xxxx0804	<b>Equate Symbol:</b> IXLRSNCODEBADVERSION# <b>Meaning:</b> There is an invalid version number in the IXLREBLD parameter list. <b>Action:</b> <ul style="list-style-type: none"> <li>• Verify that your program did not overlay the parameter list storage.</li> <li>• Verify that your program was assembled with the correct macro library for the release of MVS that your program is running on.</li> </ul>



Table 87. Return and reason codes for the IXLREBLD macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0806	<b>Equate Symbol:</b> IXLRSNCODESRBMODE <b>Meaning:</b> The requester is in SRB mode. <b>Action:</b> Do not issue the IXLREBLD macro when running in SRB mode.
8	xxxx0807	<b>Equate Symbol:</b> IXLRSNCODENOTENABLED <b>Meaning:</b> The requester is not in an enabled state. <b>Action:</b> Issue the IXLREBLD macro while running enabled for I/O and external interrupts.
8	xxxx0809	<b>Equate Symbol:</b> IXLRSNCODEPRIMARYNOTHOME <b>Meaning:</b> The requester's primary address space is not the same as the home address space. <b>Action:</b> Make sure that the primary address space and the home address space are the same at the time of the IXLREBLD invocation.
8	xxxx080A	<b>Equate Symbol:</b> IXLRSNCODEBADCONTOKEN <b>Meaning:</b> The requester specified a CONTOKEN that is not valid. The contoken is invalid for one of the following reasons: disconnect has occurred, EOT of the connector's task, input contoken is not the contoken returned from IXLCONN, or the request was issued outside the connector's address space. <b>Action:</b> Verify that the CONTOKEN value specified was valid and for the correct structure.
8	xxxx084C	<b>Equate Symbol:</b> IXLRSNCODENOSAFAUTH <b>Meaning:</b> The requester does not have the required SAF authorization. <b>Action:</b> Determine why the installation has not allowed the proper SAF authorization for access to the structure.
8	xxxx0863	<b>Equate Symbol:</b> IXLRSNCODETASKTERM <b>Meaning:</b> The request is rejected because the requesting task is going through termination. IXLREBLD cannot be issued from a resource manager. <b>Action:</b> Examine your protocol to ensure that IXLREBLD is not issued from a resource manager.
8	xxxx0886	<b>Equate Symbol:</b> IXLRSNCODEBADREQUEST <b>Meaning:</b> The IXLREBLD REQUEST type is not supported. <b>Action:</b> Ensure that the request type has been specified correctly and that you are running on an appropriate level of z/OS.

Table 87. Return and reason codes for the IXLREBLD macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx08B4	<b>Equate Symbol:</b> IXLRSNCODEBADUSERTEXT <b>Meaning:</b> Program error. The USERTEXT area that is specified on an IXLREBLD POPULATING or WAITING request is not addressable. <b>Action:</b> Ensure that the user defined text resides in an accessible address or data space.
C	xxxx0C05	<b>Equate Symbol:</b> IXLRSNCODESTRNOTINPOLICY <b>Meaning:</b> Environmental error. The structure specified is not defined in the active CFRM policy. <b>Action:</b> Ensure that the structure name has been specified correctly.
C	xxxx0C07	<b>Equate Symbol:</b> IXLRSNCODECFNOTINPOLICY <b>Meaning:</b> Environmental error. The coupling facility specified is not defined in the active CFRM policy. <b>Action:</b> Ensure that the coupling facility name has been specified correctly.
C	xxxx0C0A	<b>Equate Symbol:</b> IXLRSNCODESTRNOTALLOCATED <b>Meaning:</b> Environmental error. The structure specified is not allocated. <b>Action:</b> Ensure that the structure name has been specified correctly.
C	xxxx0C13	<b>Equate Symbol:</b> IXLRSNCODEREQPURGED <b>Meaning:</b> Environmental error. The request was purged prior to completion of the request. Possible reasons include: <ul style="list-style-type: none"> <li>• The connector failed.</li> <li>• The connector disconnected.</li> <li>• The requestor failed.</li> <li>• The request was purged by IXLPURGE.</li> <li>• Requests were purged when the connector provided an IXLEERSP response for the Rebuild Stop or Rebuild Cleanup event.</li> <li>• The secondary address space was no longer valid.</li> </ul> <b>Action:</b> None if this is expected. Otherwise, determine why the connector failed.
C	xxxx0C25	<b>Equate Symbol:</b> IXLRSNCODESTRFAILURE <b>Meaning:</b> Environmental error. The rebuild request has been rejected because the structure has failed. A rebuild request that would result in a system-managed rebuild will be rejected if the structure has failed. <b>Action:</b> Examine the protocol in use by the application for recovery from structure failure.

Table 87. Return and reason codes for the IXLREBLD macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C29	<b>Equate Symbol:</b> IXLRSNCODEXESNOTACTIVE <b>Meaning:</b> Environmental error. CFRM services are not active or are not available. <b>Action:</b> Bring the CFRM couple data set in use in the sysplex.
C	xxxx0C35	<b>Equate Symbol:</b> IXLRSNCODENOACTIVECONNS <b>Meaning:</b> Environmental error. There are no active connections to the structure. The request to rebuild the structure is rejected. <b>Action:</b> Ensure that there is at least one active connection to the structure that is to be rebuilt.
C	xxxx0C36	<b>Equate Symbol:</b> IXLRSNCODESTOPINPROGRESS <b>Meaning:</b> Environmental error. The rebuild start or startduplex request for the structure fails because processing to stop rebuilding the same structure name is in progress. <b>Action:</b> Examine your rebuild protocol.
C	xxxx0C3D	<b>Equate Symbol:</b> IXLRSNCODENOTREBUILDING <b>Meaning:</b> Environmental error. The request to rebuild the structure is rejected because the structure is not in the rebuild process. <b>Action:</b> Ensure that the structure name was specified correctly.
C	xxxx0C3E	<b>Equate Symbol:</b> IXLRSNCODEINCLEANUP <b>Meaning:</b> Environmental error. The rebuild stop or stopduplexing request cannot be processed once the rebuild has entered the Cleanup phase. The rebuild process cannot be stopped. <b>Action:</b> Do not attempt to stop the rebuild or duplex rebuild process once the rebuild has entered the Cleanup phase.
C	xxxx0C3F	<b>Equate Symbol:</b> IXLRSNCODECONNNOTDEFINED <b>Meaning:</b> Environmental error. One of the following: <ul style="list-style-type: none"> <li>• The connection making the rebuild complete or duplex complete request is not defined.</li> <li>• The connection specified by the WAITING request WAITFORCONTOKEN keyword is the same as the invoker.</li> </ul> <b>Action:</b> <ul style="list-style-type: none"> <li>• For IXLREBLD REQUEST=COMPLETE or REQUEST=DUPLEXCOMPLETE, ensure that the request is initiated by an active connector to the structure.</li> <li>• For an IXLREBLD REQUEST=WAITING request, ensure that the WAITFORCONTOKEN keyword designates a connector with an active rebuild connection to the structure and is not the invoking connector.</li> </ul>

Table 87. Return and reason codes for the IXLREBLD macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C40	<p><b>Equate Symbol:</b> IXLRSNCODECONNNOTACTIVE</p> <p><b>Meaning:</b></p> <ul style="list-style-type: none"> <li>The connection indicating rebuild complete or duplex complete is not active.</li> <li>The connection specified by the WAITING request WAITFORCONTOKEN keyword has not connected to the rebuild new structure.</li> </ul> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>For IXLREBLD REQUEST=COMPLETE or REQUEST=DUPLEXCOMPLETE, ensure that the request is invoked by an active connector to the structure.</li> <li>For an IXLREBLD REQUEST=WAITING request, ensure that the WAITFORCONTOKEN keyword designates a connector with an active rebuild connection to the structure.</li> </ul>
C	xxxx0C41	<p><b>Equate Symbol:</b> IXLRSNCODEUNEXPECTEDRESPONSE</p> <p><b>Meaning:</b> Environmental error. The system did not expect a rebuild complete or duplex complete request from the connection.</p> <p><b>Action:</b> Examine the rebuild protocol in use by the application to ensure that the sequence of rebuild events/actions is correct.</p>
C	xxxx0C46	<p><b>Equate Symbol:</b> IXLRSNCODEREQUESTNOTEXPECTED or IXLRSNCODEREBUILDCOMPLETE (deprecated)</p> <p><b>Meaning:</b> Environmental error. The request is not expected during the current phase of processing. Applicable requests include:</p> <ul style="list-style-type: none"> <li>IXLREBLD REQUEST=COMPLETE</li> <li>IXLREBLD REQUEST=POPULATING</li> <li>IXLREBLD REQUEST=WAITING.</li> </ul> <p>For IXLREBLD REQUEST=POPULATING or WAITING, this reason code might also indicate that the connector did not specify IXLCONN MONITOR=IXLREBLD on the rebuild connect request</p> <p><b>Action:</b> Examine the rebuild protocol in use by the application to ensure that the sequence of rebuild events/actions is correct.</p>
C	xxxx0C4A	<p><b>Equate Symbol:</b> IXLRSNCODEREBUILDNOTPERMITTED</p> <p><b>Meaning:</b> Environmental error. At least one active connection specified the ALLOWREBLD=NO option on IXLCONN for the structure. This prevents structure rebuild or duplexing rebuild from being initiated for the structure.</p> <p><b>Action:</b> Examine the rebuild protocol to determine why structure rebuild or duplexing rebuild is not allowed for this structure.</p>

Table 87. Return and reason codes for the IXLREBLD macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C67	<p><b>Equate Symbol:</b> IXLRSNCODEREBLDNOOTHER</p> <p><b>Meaning:</b> Environmental error. LOCATION=OTHER either was specified on the rebuild request or was defaulted to for STARTREASON=LOSSCONN or REQUEST=STARTDUPLEX rebuild requests. When duplexing is stopped by the operator and DUPLEX(ENABLED) is specified in the active policy for the structure, the subsequent duplexing rebuild request initiated due to DUPLEX(ENABLED) will both avoid the coupling facility in which the current structure is allocated and the coupling facility in which the previous instance of the structure was allocated when the duplexing rebuild was stopped. No other coupling facility exists in the active (or pending) policy preference list for the structure.</p> <p><b>Action:</b> Ensure that the CFRM policy is set up with more than one coupling facility in the preference list for the structure.</p>
C	xxxx0C6A	<p><b>Equate Symbol:</b> IXLRSNCODEREBLDNOOTHERCONN</p> <p><b>Meaning:</b> Environmental error. No coupling facility in the preference list provided better connectivity than the current facility for this LOSSCONN rebuild. The rebuild was not started to avoid a further degradation in connectivity for the application.</p> <p>When STARTREASON=LOSSCONN is specified, the system ignores the LESSCONNACTION specification and processes the request as if LESSCONNACTION=TERMINATE were specified.</p> <p><b>Action:</b> Either disconnect from the structure or reattempt to rebuild the structure by reissuing the IXLREBLD macro with a different STARTREASON.</p>
C	xxxx0C6B	<p><b>Equate Symbol:</b> IXLRSNCODEREBLDINSUFFCONN</p> <p><b>Meaning:</b> Environmental error. No coupling facility in the preference list provided better or equivalent connectivity than the current facility. The rebuild was not started to avoid a further degradation in connectivity for the application.</p> <p><b>Action:</b> If you want the rebuild to occur despite a degradation in connectivity, use the LESSCONNACTION=CONTINUE option.</p>

Table 87. Return and reason codes for the IXLREBLD macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C6F	<p><b>Equate Symbol:</b> IXLRNCODEDUPLEXNOTPERMITTED</p> <p><b>Meaning:</b> Environmental error. The structure does not support duplexing rebuild for one of the following reasons:</p> <ul style="list-style-type: none"> <li>• The ALLOWDUPREBLD=NO option on IXLCONN was specified or defaulted to by at least one active or failed-persistent connection.</li> <li>• DUPLEX(DISABLED) was specified or defaulted to in the CFRM active policy for this structure.</li> <li>• A user-managed duplexing rebuild was required, and user-managed duplexing rebuilds are not supported for the type of structure.</li> <li>• There are connections pending reconciliation into the CFRM policy.</li> <li>• A system-managed duplexing rebuild is not supported when a CFRM policy change is pending for the structure.</li> </ul> <p><b>Action:</b> Rebuild the structure first to cause the pending policy change to take effect.</p>
C	xxxx0C70	<p><b>Equate Symbol:</b> IXLRNCODEWRONGBUILDTYPE</p> <p><b>Meaning:</b> Environmental error. Either of the following conditions exists:</p> <ul style="list-style-type: none"> <li>• IXLREBLD REQUEST=STOP was requested and a duplexing rebuild is in progress</li> <li>• IXLREBLD REQUEST=STOPDUPLEX was requested and a structure rebuild is in progress.</li> </ul> <p><b>Action:</b> Examine the rebuild protocol in use by the application to ensure that the sequence of rebuild or duplexing events and actions is correct.</p>
C	xxxx0C71	<p><b>Equate Symbol:</b> IXLRNCODENOTDUPLEXESTAB</p> <p><b>Meaning:</b> Environmental error. An IXLREBLD REQUEST=STOP to switch to the new structure was requested and the rebuild process is not yet in the Duplex Established phase. A stop to switch to the new structure cannot be accepted until the rebuild reaches the Duplex Established phase.</p> <p><b>Action:</b> Examine the duplexing protocol in use by the application to ensure that the sequence of duplexing events and actions is correct.</p>
C	xxxx0C72	<p><b>Equate Symbol:</b> IXLRNCODEDUPLEXCOMPLETE</p> <p><b>Meaning:</b> Environmental error. An IXLREBLD REQUEST=DUPLEXCOMPLETE request was not expected at this time. Either switch is not in progress or the connector has not established duplexing yet. If the latter, the connector must either establish duplexing or disconnect, allowing switch processing to proceed.</p> <p><b>Action:</b> Examine the duplexing protocol in use by the application to ensure that the sequence of duplexing events and actions is correct.</p>

Table 87. Return and reason codes for the IXLREBLD macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C73	<p><b>Equate Symbol:</b> IXLRNCODESTRFAILED</p> <p><b>Meaning:</b> Environmental error. The rebuild request was rejected because the structure failed. A duplexing rebuild request will be rejected if the structure has failed. A structure rebuild request will be permitted.</p> <p><b>Action:</b> Start a structure rebuild for the structure.</p>
C	xxxx0C74	<p><b>Equate Symbol:</b> IXLRNCODESTOPPINGDIRECTION</p> <p><b>Meaning:</b> Environmental error. The duplexing rebuild is stopping in a direction that will not keep the structure specified on the IXLREBLD STOPDUPLEX request. The request could not be processed as requested for one of the following reasons:</p> <ul style="list-style-type: none"> <li>• A request to stop structure duplexing has already been initiated in the other direction. This request is rejected.</li> <li>• A recovery manager is active and this request with REASON=LOSSCONN specified would not have kept the structure in the coupling facility at the recovery site. Duplexing is instead stopped in the other direction to keep the structure in the coupling facility at the recovery site.</li> </ul> <p><b>Action:</b> Examine the duplexing protocol in use by the application to ensure that the sequence of duplexing events and actions is correct.</p>
C	xxxx0C75	<p><b>Equate Symbol:</b> IXLRNCODEDUPLEXNOTFEASIBLE</p> <p><b>Meaning:</b> The IXLREBLD START DUPLEX request was not processed because XES determined that allocation of the rebuild new structure would not be feasible. Message IXC574I contains additional diagnostic information.</p> <p>Reasons why a duplexing rebuild might not be feasible include the following:</p> <ul style="list-style-type: none"> <li>• There is no coupling facility in the preference list that has CF-to-CF connectivity to the coupling facility in which the primary structure is allocated.</li> <li>• The CFRM policy requires that the active instance of the structure be duplexed using asynchronous duplexing, but no coupling facility in the preference list is eligible for asynchronous duplexing with the active structure instance.</li> <li>• A loss of coupling facility connectivity would occur for active connectors to the structure.</li> </ul> <p><b>Action:</b> Depending on the reason that the allocation of the rebuild new structure was not considered feasible, put actions in place to correct the situation.</p> <p>For additional information about the failure, search the hardcopy log for message IXC574I. Additional diagnostic details can also be found in a system LOGREC symptom record.</p>

Table 87. Return and reason codes for the IXLREBLD macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C80	<p><b>Equate Symbol:</b> IXLRNCDEREBUILDPOPCFINPROGRESS</p> <p><b>Meaning:</b> Environmental error. An IXLREBLD REQUEST=START,POPULATECF=<i>cfname</i> was attempted when either a previous POPULATECF request or REALLOCATE process was already in progress. The request is not processed.</p> <p><b>Action:</b> Examine the protocol in use by the application to ensure that the sequence of events and actions when populating a coupling facility is correct. The POPULATECF function and REALLOCATE process are mutually exclusive. The REALLOCATE process can only be started or stopped by using the SETXCF operator command.</p>
C	xxxx0C81	<p><b>Equate Symbol:</b> IXLRNCDEREBUILDPOPCFNOTINPROGRESS</p> <p><b>Meaning:</b> Environmental error. An IXLREBLD STOP,POPULATECF=<i>cfname</i> was attempted. However, there is no currently active POPULATECF request in progress for the specified coupling facility. The request is not processed.</p> <p><b>Action:</b> Examine the protocol in use by the application to ensure that the sequence of events and actions when populating a coupling facility is correct.</p>
C	xxxx0C83	<p><b>Equate Symbol:</b> IXLRNCDEREBUILDPOPCFNOSTRUCTS</p> <p><b>Meaning:</b> Environmental error. An IXLREBLD REQUEST=START,POPULATECF=<i>cfname</i> was attempted. No structures were selected for the request. The request is not processed.</p> <p><b>Action:</b> Examine the protocol in use by the application to ensure that the sequence of events and actions when populating a coupling facility is correct.</p>
C	xxxx0C84	<p><b>Equate Symbol:</b> IXLRNCDEREBUILDPOPCFFAILED</p> <p><b>Meaning:</b> Environmental error. An IXLREBLD REQUEST=START,POPULATECF=<i>cfname</i> was attempted. The specified coupling facility has failed. The request is not processed.</p> <p><b>Action:</b> Resubmit the request specifying a coupling facility that is available in the CFRM active policy or change the policy to reinstate the coupling facility.</p>
C	xxxx0C85	<p><b>Equate Symbol:</b> IXLRNCDEREBUILDPOPCFINCLEANUP</p> <p><b>Meaning:</b> Environmental error. An IXLREBLD REQUEST=START,POPULATECF=<i>cfname</i> was attempted. The specified coupling facility is in cleanup processing. The request is not processed.</p> <p><b>Action:</b> Resubmit the request after the coupling facility has completed cleanup.</p>



Table 87. Return and reason codes for the IXLREBLD macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C86	<p><b>Equate Symbol:</b> IXLRNCDEREBUILDPOPCFDELETEPENDING</p> <p><b>Meaning:</b> Environmental error. An IXLREBLD REQUEST=START,POPULATECF=<i>cfname</i> was attempted. The specified coupling facility is being deleted from the CFRM active policy. The request is not processed.</p> <p><b>Action:</b> Resubmit the request specifying a coupling facility that is available in the CFRM active policy or change the policy to reinstate the coupling facility.</p>
C	xxxx0C88	<p><b>Equate Symbol:</b> IXLRNCDEREBUILDPOPCFALLOCNOTPERMITTED</p> <p><b>Meaning:</b> A request to start a POPULATECF rebuild was attempted. Structure allocation is not permitted in the specified coupling facility. The request is not processed.</p> <p><b>Action:</b> The specified coupling facility is not eligible for structure allocation. Resubmit the request and specify a coupling facility that is available for structure. Otherwise, if the specified coupling facility is the intended target, issue the DISPLAY XCF,CF command to determine the possible reasons why the coupling facility is not eligible for structure allocation, and take any necessary actions to allow structure allocation.</p>
C	xxxx0C92	<p><b>Equate Symbol:</b> IXLRNCDENCODESYSGDNOTSUPPORTEDSTR</p> <p><b>Meaning:</b> An IXLREBLD REQUEST=START was attempted that needed system-managed processing (for example, rebuild). The system-managed process cannot be initiated for one of the following reasons:</p> <ul style="list-style-type: none"> <li>• The structure was not allocated in a coupling facility at or above the minimum CFLEVEL required for the current process.</li> <li>• The structure was not allocated by a system that supports system-managed processing.</li> <li>• The structure has connections that have not been reconciled into the CFRM active policy.</li> <li>• Structure cleanup is in progress for the structure (applicable to lock and serialized list structures only).</li> </ul> <p>The request is not processed.</p> <p><b>Action:</b> Determine the CFLEVEL of the coupling facility in which the structure is allocated through one of the following methods: using the IXLMG macro, issuing the DISPLAY XCF,STR and DISPLAY CF commands, or referencing the data returned in the CONA on the original connect to the structure.</p> <ul style="list-style-type: none"> <li>• If the CFLEVEL is too low, follow application protocols to shut down and allocate the structure in a coupling facility of suitable CFLEVEL.</li> <li>• If the CFLEVEL is sufficient, try the IXLREBLD request later to allow CFRM policy reconciliation or lock structure cleanup to complete.</li> </ul>

Table 87. Return and reason codes for the IXLREBLD macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C93	<p><b>Equate Symbol:</b> IXLRNICODESYSMGDSTRPREFLIST</p> <p><b>Meaning:</b> An IXLREBLD REQUEST=START was attempted that needed system-managed processing (for example, rebuild). The system-managed process cannot be initiated because the preference list for the structure was unsuitable for one of the following reasons:</p> <ul style="list-style-type: none"> <li>• The preference list is empty.</li> <li>• The preference list contains no other coupling facility at the required CFLEVEL or higher.</li> <li>• The structure already exists in the only suitable coupling facility and this coupling facility could not be selected as the target for the system-managed process because no CFRM policy change is pending for this structure.</li> </ul> <p>The request is not processed.</p> <p><b>Action:</b> The system programmer must update the structure's preference list to include at least two coupling facilities at the required CFLEVEL.</p>
C	xxxx0C94	<p><b>Equate Symbol:</b> IXLRNICODESYSMGDNOTSUPPORTEDCONN</p> <p><b>Meaning:</b> An IXLREBLD REQUEST=START request was attempted and would have resulted in system-managed processing (for example, rebuild). The system-managed process could not be initiated because there is at least one active connection and all connections did not specify ALLOWAUTO=YES on IXLCONN. The request is not processed.</p> <p><b>Action:</b> The application must support either user-managed or system-managed rebuild for the rebuild start request to succeed. If the application intends to support system-managed processing, all active connectors that specified or defaulted to IXLCONN ALLOWAUTO=NO must disconnect and, if desired, reconnect with ALLOWAUTO=YES, before the request can be processed successfully.</p>
C	xxxx0C95	<p><b>Equate Symbol:</b> IXLRNICODESYSMGDBADSTARTREASON</p> <p><b>Meaning:</b> An IXLREBLD REQUEST=START invocation would have resulted in a system-managed rebuild. The rebuild could not be initiated because the request specified a STARTREASON of LOSSCONN or STRFAILURE, which are not valid reasons for starting a system-managed rebuild. The request is not processed.</p> <p><b>Action:</b> Specify a STARTREASON other than LOSSCONN or STRFAILURE.</p>

Table 87. Return and reason codes for the IXLREBLD macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C96	<p><b>Equate Symbol:</b> IXLRNICODESYSMGDLOSSCONN</p> <p><b>Meaning:</b> An IXLREBLD REQUEST=START invocation would have resulted in a system-managed rebuild. The rebuild could not be initiated because an active or failing connector does not have connectivity to the target structure. The request is not processed.</p> <p><b>Action:</b> Try the start request again when all connectors who have lost connectivity to the structure have disconnected and become either failed-persistent or undefined.</p>
C	xxxx0C97	<p><b>Equate Symbol:</b> IXLRNICODESYSMGDREQUESTNOTPERMITTED or IXLRNICODESYSMGDCOMPLETENOTPERMITTED (deprecated)</p> <p><b>Meaning:</b> Environmental error. The request was issued for a structure that is undergoing a system-managed process (for example, rebuild). The request is not processed. Applicable requests include:</p> <ul style="list-style-type: none"> <li>• IXLREBLD REQUEST=COMPLETE</li> <li>• IXLREBLD REQUEST=DUPLEXCOMPLETE</li> <li>• IXLREBLD REQUEST=POPULATING</li> <li>• IXLREBLD REQUEST=WAITING</li> </ul> <p><b>Action:</b> Do not issue the above requests against a structure that is undergoing system managed processing.</p>
C	xxxx0C99	<p><b>Equate Symbol:</b> IXLRNICODESYSMGDNOTSUPPORTEDCDS</p> <p><b>Meaning:</b> An IXLREBLD REQUEST=START request was attempted which needed system-managed processing (for example, rebuild). The system-managed process could not be initiated because the CFRM couple data set was not formatted at the minimum required level. The request is not processed.</p> <p><b>Action:</b> Format and activate a CFRM couple data set that supports the requested system-managed process. For rebuild, specify ITEM NAME(SMREBLD) NUMBER(1) when formatting.</p>
C	xxxx0C9B	<p><b>Equate Symbol:</b> IXLRNICODESYSMGDNOHISTORY</p> <p><b>Meaning:</b> A request to initiate a duplexing rebuild was attempted which needed system-managed processing. The system-managed duplexing rebuild cannot be initiated because there are no connections to the structure and the structure has not previously been duplexed using system-managed processing. The request is not processed.</p> <p><b>Action:</b> Start a connector that supports system-managed duplexing rebuild, and try again.</p>

Table 87. Return and reason codes for the IXLREBLD macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0CA2	<b>Equate Symbol:</b> IXLRSNCODESTORAGECLASSMEMORYINUSE <b>Meaning:</b> The request to start a duplexing rebuild was rejected because the structure contains objects in storage-class memory. <b>Action:</b> Retry the rebuild request when the structure no longer contains objects in storage-class memory.
C	xxxx0CA9	<b>Equate Symbol:</b> IXLRSNCODEASYNCDUPLEXSTR <b>Meaning:</b> An IXLREBLD REQUEST=STARTDUPLEX was attempted. The CFRM policy requires that duplexing with the active structure be done with system-managed asynchronous duplexing, but the active structure does not support system-managed asynchronous duplexing for one of the following reasons: <ul style="list-style-type: none"> <li>• The CFLEVEL of the coupling facility in which the structure is allocated does not support asynchronous duplexing.</li> <li>• Asynchronous duplexing not supported for the structure type.</li> <li>• The structure was not allocated by a system supporting system-managed asynchronous duplexing.</li> </ul> The request is not processed. <b>Action:</b> Notify the system programmer.
	xxxx0CAA	<b>Equate Symbol:</b> IXLRSNCODEASYNCDUPLEXCONN <b>Meaning:</b> An IXLREBLD REQUEST=STARTDUPLEX was attempted. The CFRM policy requires that duplexing with the active structure be done with system-managed asynchronous duplexing, but one or more active connectors is ASYNCDUPLEX=NO. The request is not processed. <b>Action:</b> Ensure that the IXLCONN does not specify (or default to) ASYNCDUPLEX=NO. Notify the system programmer.
C	FFFFFFFF	<b>Equate Symbol:</b> IXLRSNCODENOTAVAILABLE <b>Meaning:</b> XES functions are not available. This can occur because the coupling facility hardware necessary to provide XES functions is not present. <b>Action:</b> Re-IPL the system, or follow your particular failure management protocol.
10	xxxx10xx	<b>Equate Symbol:</b> IXLRSNCODESYSMDXCFERROR <b>Meaning:</b> Failure in XES processing. The state of the resource request is unpredictable. <b>Action:</b> Save the reason code information and contact the IBM support center.

## Chapter 82. IXLRT – Lock structure record data processing

### Description

The IXLRT service enables you to perform recovery procedures if a connection to a lock structure fails. The IXLRT service is the recovery interface to obtain and clean up recording information in a lock structure.

The following services are available:

- Create a record data entry and write data to the entry.
- Read the entire set of record data entries in the lock structure.
- Read the entire set of record data entries associated with a connected user.
- Read a single record data entry by entry identifier.
- Delete all record data entries identified by a list of entry identifiers.
- Delete the entire set of record data entries associated with a connected user.
- Delete a single record data entry by entry identifier.
- Update a record data entry by entry identifier.

### Environment

The following are the environment requirements for the caller.

Environment	Environment requirement
Minimum authorization:	Supervisor state or PKM allowing key 0 - 7
Dispatchable unit mode:	Task or SRB
Cross memory mode:	Any PASN, any HASN, any SASN. The primary address space must be the same as the primary address space at the time the connection service (IXLCONN) was issued.
AMODE:	31-bit
ASC mode:	Primary or access register (AR)
Interrupt status:	Enabled for I/O and external interrupts
Locks:	No locks held
Control parameters:	Must be in the primary address space or be in an address/data space that is addressable through a public entry on the caller's dispatchable unit access list (DU-AL)

### Programming Requirements

Before using the IXLRT service, you must identify your connection to the system through the IXLCONN service. On successful completion of the IXLCONN service, you receive a sysplex-wide unique CONNECT token. This token identifies your connection to the lock structure. The connect token must be specified on every IXLRT request to ensure that you are allowed access to the designated record data structure.

The caller's parameter list must be addressable from the unit of work issuing the request.

The IXLYMRTD macro provides the format of the area that the DATAREA points to. Include that macro in your program.

The IXLYRTAA mapping macro provides the format of the area that the ANSAREA points to. Include that macro in your program.

## Restrictions and Limitations

---

The system does not serialize the record data entry. Therefore, unless the caller provides serialization to prevent entries from being created while IXLRT reads or deletes entries, it is possible that not all record data entries will be returned to the requestor.

## Input Register Information

---

Before issuing the IXLRT macro, the caller does not have to place any information into any register unless using it in register notation for a particular parameter, or using it as a base register.

## Output Register Information

---

When control returns to the caller of the IXLRT macro, the general purpose registers (GPRs) contain:

Register	Contents
----------	----------

<b>0</b>	Reason code if GPR15 return code is nonzero
<b>1</b>	Used as a work register by the system
<b>2-13</b>	Unchanged
<b>14</b>	Used as a work register by the system
<b>15</b>	Return code

When control returns to the caller of the IXLRT macro, the access registers (ARs) contain:

Register	Contents
----------	----------

<b>0-1</b>	Used as a work register by the system
<b>2-13</b>	Unchanged
<b>14-15</b>	Used as a work register by the system

**For registers that the system changes,** a caller who depends on these registers containing the same value before and after issuing the macro must save these registers before issuing the macro and restore them after the system returns control.

## Performance Implications

---

None.

## Understanding IXLRT Version Support

---

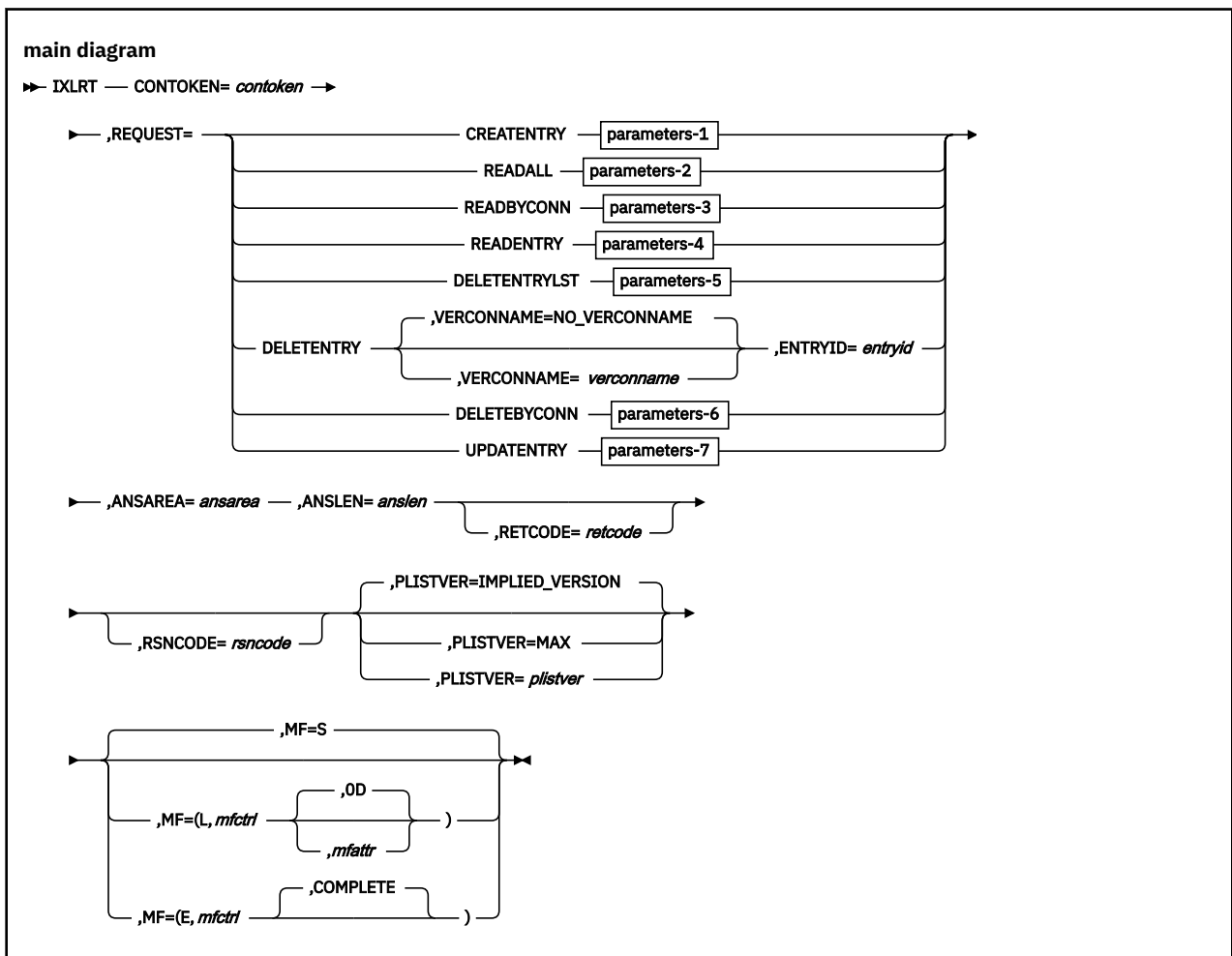
The IXLRT macro supports versions 0 through 4.

- Keywords not specifically noted here are supported by all versions starting with version 0 and higher of the IXLRT macro.
- The following keyword is supported by all versions starting with version 1 and higher of the IXLRT macro.
  - FASTRESTOKEN
- The following keyword is supported by all versions starting with version 2 and higher of the IXLRT macro.
  - RDATA TYPE
- The following keyword is supported by all versions starting with version 3 and higher of the IXLRT macro.
  - EXTRESTOKEN
- The following keywords are supported by all versions starting with version 4 and higher of the IXLRT macro.
  - MRTDLEVEL
  - OUTRDATA TYPE

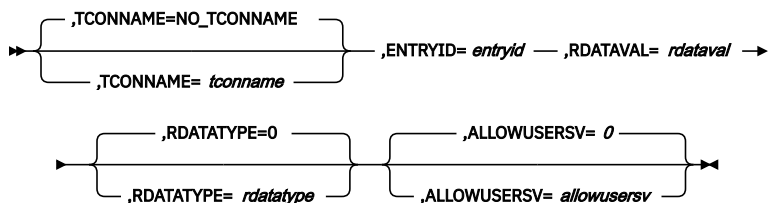
Specify the version of the parameter list that you want generated with the PLISTVER keyword. See Chapter 2, “Specifying a Macro Version Number,” on page 5 for considerations when specifying the version of the parameter list with PLISTVER.

## Syntax Diagram

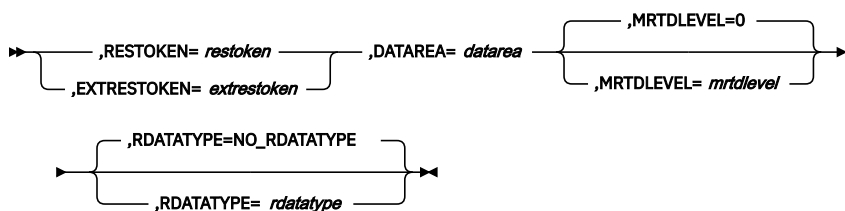
The syntax of the IXLRT macro is as follows:



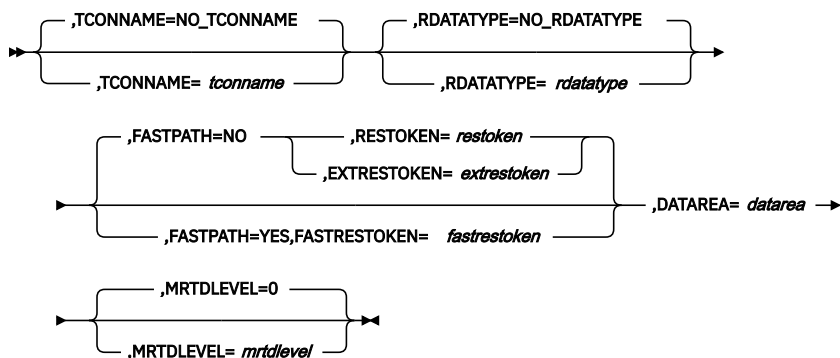
## parameters-1



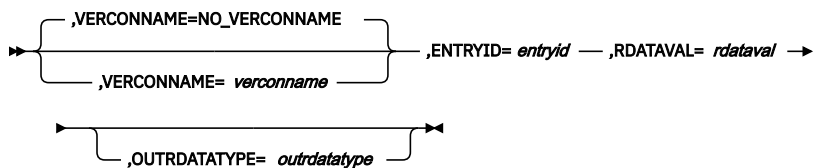
## parameters-2



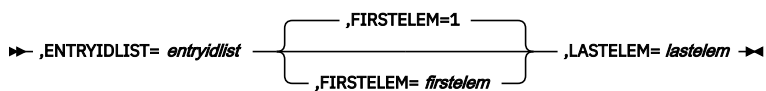
## parameters-3



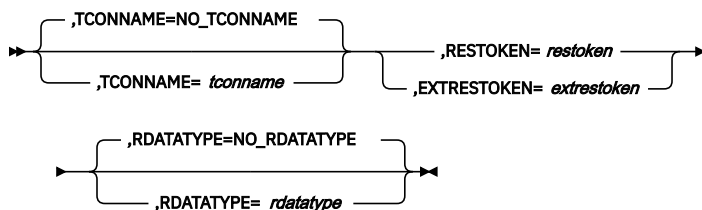
## parameters-4



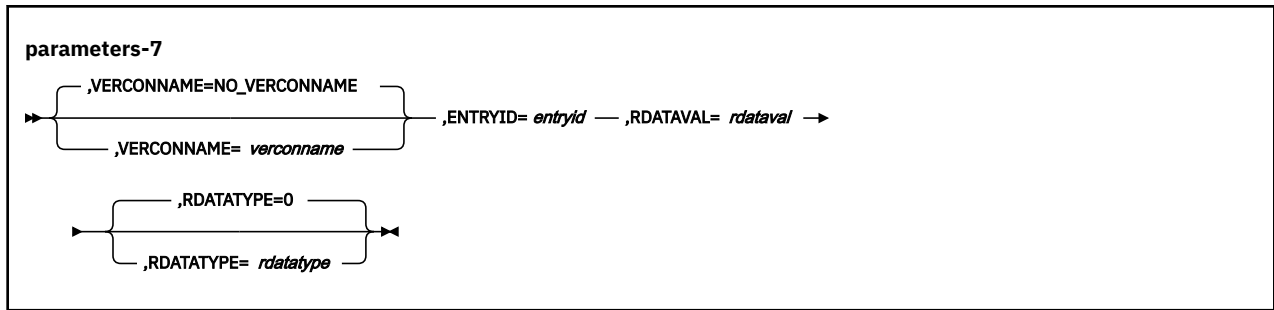
## parameters-5



## parameters-6







**Note:** FASTPATH=NO is the default and does not need to be coded. However, if you select this parameter, you must code either RESTOKEN=*restoken* or EXTRESTOKEN=*extrestoken*.

## Parameter descriptions

The parameter descriptions are listed in alphabetical order. Default values are underlined:

### **,ALLOWUSERSV=0**

#### **,ALLOWUSERSV=*allowusersv***

Use this input parameter to indicate whether to allow a request that attempts to create a record data entry to proceed if the resulting percentage of free entries at the completion of the request would be less than the established percent entry reserved threshold (if any) for the structure.

The ALLOWUSERSV parameter is meaningful only when the PCTENTRYRSV parameter is used on an IXLCONN service invocation to establish a non-zero percent entry reserved threshold for the lock structure and the lock structure is allocated in a CFLEVEL=25 or higher coupling facility.

A value of 0 (IxLLockAllowUseRsvNo) indicates that if request processing creates an entry and the resulting percentage of free entries at the completion of the request would be less than the established percent entry reserved threshold for the structure, the request is not permitted to create the record data entry and the request will fail with a return code of IxlRetCodeEnvError, reason code of IxlRsnCodeRtFull.

A value of 1 (IxLLockAllowUseRsvYes) indicates that the established percent entry reserved threshold for the structure should be ignored for this request and an entry should be created as long as there is a free record data entry to satisfy the request. Use IxLLockAllowUseRsvYes as the value for AllowUseRsv during application and connector recovery scenarios when using reserve entries to create a record data entry is deemed necessary.

Any other specified value will have the same behavior as specifying a value of 0 (IxLLockAllowUseRsvNo).

DEFAULT: 0

### **,ANSAREA=*ansarea***

Use this output area to contain information about the request. The storage area is mapped by the IXLYRTAA macro and must be addressable in the caller's primary address space.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an output area is to contain information about the request.

### **,ANSLEN=*anslen***

Use this input area to specify the length of the ANSAREA storage area.

The answer area length determines the level of data to be returned in the ANSAREA. For example, level 1 data is only returned if the answer area is large enough to hold level 1 data. Otherwise level 0 data is returned. Length constants are defined in the IXLYRTAA macro to indicate the answer area length required for each data level. Consult the IXLYRTAA mapping macro for information on the data to be returned for each level.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a halfword input field that specifies the length of ANSAREA.

**CONTOKEN=contoken**

Use this input parameter to specify the connect token of the requester. The token is the connect token that was returned to the requester by the IXLCONN service. CONTOKEN uniquely identifies the user's connection to a lock structure.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-character input field that contains the connect token returned in the answer area by the IXLCONN service.

**,DATAREA=datarea**

Use this output area to contain the data returned by a READALL or READBYCONN request. This area must be 4096 bytes long and is mapped by the IXLYMRTD macro.

Upon successful completion of a READALL or READBYCONN request, DATAREA contains, starting at offset zero, an array of records containing 64 bytes of record data, the entry identifier (ENTRYID) for the element that was successfully read, and the connection identifier (CONID) of the connected user who is associated with each element that was successfully read. Additional information may also be provided, based on the MRTDLEVEL requested. The number of record data entries returned is indicated in the answer area specified by ANSAREA, mapped by IXLYRTAA.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 4096-byte output area to contain the data returned by a READALL or READBYCONN request.

**,ENTRYID=entryid**

Use this input/output parameter to specify the identifier assigned to the record data entry.

For a CREATENTRY request, this identifier is returned by IXLRT and must be used on all subsequent requests to access the record data entry.

For READENTRY and DELETENTRY requests, this identifier specifies the unique entry identifier assigned to the record data entry.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 12-character input/output field that contains the entry identifier.

**,ENTRYIDLIST=entryidlist**

Use this input parameter to identify an area that contains a list of record data entry identifiers to be processed by the DELETENTRYLIST request. This list must be aligned on a page boundary and be 4096 bytes long. The ENTRYIDLIST must be formatted into 12-byte elements starting at offset zero. Each record data entry to be processed consists of a 12-byte identifier.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 4096-byte input area to contain a list of record data entry identifiers to be processed by a DELETENTRYLIST request.

**,EXTRESTOKEN=extrestoken**

Use this input/output parameter to specify an extended restart token that can be used to resume processing of all READALL, READBYCONN, and DELETEBYCONN requests. Requesters that specify IXLCONN ALLOWAUTO=YES must use the extended restart token. Requesters that specify or default to IXLCONN ALLOWAUTO=NO must use the standard restart token (RESTOKEN).

You must provide an initial value of zero in this field. The system then initializes the field with control information relevant to the READALL, READBYCONN, or DELETEBYCONN request. On each subsequent request, the system uses this information. You must not modify the contents of this field.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-character input/output area to contain an extended restart token.

**,FASTPATH=NO****,FASTPATH=YES**

Use this input parameter to indicate that the system is to process this READBYCONN request using an optimized access method when accessing the record data.

**NO**

The system is not to use the optimized access method when accessing the record data.

**YES**

The system is to use the optimized access method when accessing the record data. Use FASTPATH=YES only when the caller has serialization that ensures that the record data entries belonging to the target connector will remain unchanged throughout the process.

**,FASTRESTOKEN=*fastrestoken***

Use this input/output parameter to specify a 12-character field that must be passed on READBYCONN FASTPATH=YES requests.

You must provide an initial value of zero in the field. The system then initializes the field with control information. On each subsequent READBYCONN FASTPATH=YES request, the system uses this information. You must not modify the contents of this field.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 12-character input/output field to contain control information.

**,FIRSTELEM=1****,FIRSTELEM=*firstelem***

Use this input parameter to specify the index of the first ENTRYIDLIST element to be processed. The value specified must be in the range of 1 to 341 inclusive.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a halfword input field to contain the index of the first ENTRYIDLIST element to be processed.

**,LASTELEM=*lastelem***

Use this input parameter to specify the index of the last ENTRYIDLIST element to be processed. The value must be in the range of 1 to 341 inclusive and must be greater than, or equal to, the value of FIRSTELEM.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a halfword input field to contain the index of the last ENTRYIDLIST element to be processed.

**,MF=S****,MF=(L,*mfctrl*)****,MF=(L,*mfctrl*,*mfattr*)****,MF=(L,*mfctrl*,0D)****,MF=(M,*mfctrl*)****,MF=(M,*mfctrl*,COMPLETE)****,MF=(M,*mfctrl*,NOCHECK)****,MF=(E,*mfctrl*)****,MF=(E,*mfctrl*,COMPLETE)****,MF=(E,*mfctrl*,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

**,*mfctrl***

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

**,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

**,COMPLETE****,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

**COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if *SMILE=var* were an optional parameter and the default is *SMILE=NO\_SMILE* then it would not be documented. However, if the default was *SMILE=-:-*, then it would be documented because a value would be the default.

**NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

**,MRTDLEVEL=0****,MRTDLEVEL=mrtplevel**

Use this input parameter to specify the level of MRTD data that is to be returned in the area specified by DATAREA. Valid values are 0 and 1.

- A value of 0 indicates that the returned entries will be mapped by the MRTD mapping in IXLYMRTD.
- A value of 1 indicates that the returned entries will be mapped by the MRTD1 mapping in IXLYMRTD.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the one-byte area containing the level of the IXLYMRTD record mappings.

**,OUTRDATATYPE=outdatatype**

Use this output parameter to contain the record data type (RDATATYPE) value associated with the record data element that was returned for a READENTRY request.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-character field that will contain the returned record data element's record data type.

**,PLISTVER=IMPLIED\_VERSION****,PLISTVER=MAX****,PLISTVER=plistver**

Use this input parameter to specify the version of the macro. See [“Understanding IXLRT Version Support”](#) on page 1566 for a description of the options available with PLISTVER.

**,RDATATYPE=NO\_RDATATYPE****,RDATATYPE=0****,RDATATYPE=rdatatype**

Use this input parameter to specify the type of record data in the lock structure.

For a CREATENTRY request, RDATATYPE specifies the record data type that is to be assigned to the created record data entry. If a nonzero record data type value is not specified, a record data type of zero will be assigned to the entry. Note that record data that is created by the IXLLOCK service in conjunction with a held lock resource has a record data type of zero. DEFAULT: 0.

For a READALL, READBYCONN, and DELETEBYCONN request, RDATATYPE specifies the record data type to be used as a filter for processing the record data entries. When specified, only those record data entries whose record data type matches the specified value will be selected for processing. Note that record data that is created by the IXLLOCK service in conjunction with a held lock resource has a record data type of zero. DEFAULT: NO\_RDATATYPE.

For an UPDATENTRY request, RDATA TYPE specifies the record data type to be assigned to the updated record data entry. The record data type can be the same as or different from that current record data type for the entry. If a nonzero record data type value is not specified, a record data type of zero will be assigned to the entry. Note that record data that is created by the IXLLOCK service in conjunction with a held lock resource has a record data type of zero. DEFAULT: 0

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an 8-character field that contains the record data type.

**,REQUEST=CREATENTRY**  
**,REQUEST=READALL**  
**,REQUEST=READBYCONN**  
**,REQUEST=READENTRY**  
**,REQUEST=DELETENTRYLST**  
**,REQUEST=DELETENTRY**  
**,REQUEST=DELETEBYCONN**  
**,REQUEST=UPDATENTRY**

Use this input parameter to specify the function requested.

### **CREATENTRY**

Create a record data entry in the lock structure to which the caller is connected and write data to that entry.

When the request completes successfully, the unique entry identifier assigned by XES, the number of entries associated with the target connector, and the total number of in-use entries in the record data entry are returned in the answer area specified by ANSAREA.

If the answer area specified by ANSAREA is large enough to hold level 1 data (mapped by RTAA1), and an asynchronous duplexing request sequence number is generated for the request, then the asynchronous duplexing request sequence number is also returned in the ANSAREA. An asynchronous duplexing request sequence number is only generated for the request if:

- The request returns with a return code of 0, and
- The structure is duplexed by system-managed asynchronous duplexing when the request completes, and
- The request might not have been committed in the secondary instance of the structure.

The asynchronous duplexing request sequence number can be used on a subsequent invocation of IXLADUPX to ensure the request is committed in the secondary instance of the asynchronously duplexed structure.

### **READALL**

Read the next group of record data entries associated with the lock structure to which the caller is connected.

When the request completes successfully, the number of entries for which record data was read is returned in the answer area specified by ANSAREA.

A READALL request may complete prematurely. If so, indicative return and reason codes are provided, the number of record data entries read is provided in the answer area, and a token is returned in the location specified by the RESTOKEN or EXTRESTOKEN keyword. This token may be specified as input on a subsequent READALL request to resume processing with the appropriate record data entry.

### **READBYCONN**

Read the next group of record data entries that are associated with the specified connection name.

When the request completes successfully, the number of entries for which record data was read is returned in the answer area specified by ANSAREA.

A READBYCONN request may complete prematurely. If so, indicative return and reason codes are provided, the number of record data entries read is provided in the answer area, and a token is

returned in the location that is specified by the RESTOKEN or EXTRESTOKEN keyword. This token may be specified as input on a subsequent READBYCONN request to resume processing with the appropriate record data entry.

### **READENTRY**

Read a particular record data entry by entry identifier.

When the request completes successfully, the number of entries associated with the target connector and the total number of in-use record data entries are returned in the answer area specified by ANSAREA.

### **DELETENTRYLIST**

Delete all record data entries specified by a list of entry identifiers. This list of entry identifiers is specified by the ENTRYIDLIST keyword.

When the request completes successfully, the number of entries deleted for this request is returned in the answer area specified by ANSAREA.

If the answer area specified by ANSAREA is large enough to hold level 1 data (mapped by RTAA1), and an asynchronous duplexing request sequence number is generated for the request, then the asynchronous duplexing request sequence number is also returned in the ANSAREA. An asynchronous duplexing request sequence number is only generated for the request if:

- The request returns with a return code of 0 or 4, and
- The structure is duplexed by system-managed asynchronous duplexing when the request completes, and
- At least one record data entry was deleted, and
- The request might not have been committed in the secondary instance of the structure.

The asynchronous duplexing request sequence number can be used on a subsequent invocation of IXLADUPX to ensure the request is committed in the secondary instance of the asynchronously duplexed structure.

A DELETENTRYLIST request may complete prematurely. If so, indicative return and reason codes are provided, the number of record data entries deleted, and the index of the next entry to be deleted are provided in the ANSAREA. To continue deleting the remaining elements in the ENTRYIDLIST, the DELETENTRYLIST request can be reissued with the FIRSTLEM keyword updated to indicate the new starting point in the element list.

If any entry specified in the ENTRYIDLIST does not exist, the processing is halted and the index of the offending element in the ENTRYIDLIST is also returned in the answer area. When this occurs, all specified elements preceding the offending element have been processed. All succeeding elements have not been processed. To continue deleting the remaining elements in the ENTRYIDLIST, the DELETENTRYLIST request can be reissued with the FIRSTLEM keyword updated to indicate the new starting point in the element list.

### **DELETENTRY**

Delete an existing record data entry by entry identifier.

When the request completes successfully, the remaining number of entries associated with the target connector and the remaining total number of in-use record data entries are returned in the answer area specified by ANSAREA.

If the answer area specified by ANSAREA is large enough to hold level 1 data (mapped by RTAA1), and an asynchronous duplexing request sequence number is generated for the request, then the asynchronous duplexing request sequence number is also returned in the ANSAREA. An asynchronous duplexing request sequence number is only generated for the request if:

- The request returns with a return code of 0, and
- The structure is duplexed by system-managed asynchronous duplexing when the request completes, and
- The request might not have been committed in the secondary instance of the structure.

The asynchronous duplexing request sequence number can be used on a subsequent invocation of IXLADUPX to ensure the request is committed in the secondary instance of the asynchronously duplexed structure.

### **DELETEBYCONN**

Delete the next group of record data entries associated with the specified connection name.

When the request completes successfully, the number of entries for which record data was deleted is returned in the answer area specified by ANSAREA.

If the answer area specified by ANSAREA is large enough to hold level 1 data (mapped by RTAA1), and an asynchronous duplexing request sequence number is generated for the request, then the asynchronous duplexing request sequence number is also returned in the ANSAREA. An asynchronous duplexing request sequence number is only generated for the request if:

- The request returns with a return code of 0 or 4, and
- The structure is duplexed by system-managed asynchronous duplexing when the request completes, and
- At least one record data entry was deleted, and
- The request might not have been committed in the secondary instance of the structure.

The asynchronous duplexing request sequence number can be used on a subsequent invocation of IXLADUPX to ensure the request is committed in the secondary instance of the asynchronously duplexed structure.

A DELETEBYCONN request may complete prematurely. If so, indicative return and reason codes are provided, the number of record data entries deleted is provided in the answer area, and a token is returned in the location specified by the RESTOKEN or EXTRESTOKEN keyword. This token may be specified as input on a subsequent DELETEBYCONN request to resume processing with the appropriate record data entry.

### **UPDATENTRY**

Update an existing record data entry by entry identifier.

When the request completes successfully, the number of entries associated with the connector with whom the updated entry was associated and the total number of in-use record data entries are returned in the answer area specified by ANSAREA.

If the answer area specified by ANSAREA is large enough to hold level 1 data (mapped by RTAA1), and an asynchronous duplexing request sequence number is generated for the request, then the asynchronous duplexing request sequence number is also returned in the ANSAREA. An asynchronous duplexing request sequence number is only generated for the request if:

- The request returns with a return code of 0, and
- The structure is duplexed by system-managed asynchronous duplexing when the request completes, and
- The request might not have been committed in the secondary instance of the structure.

The asynchronous duplexing request sequence number can be used on a subsequent invocation of IXLADUPX to ensure the request is committed in the secondary instance of the asynchronously duplexed structure.

### **,RDATAVAL=rdataval**

Use this input/output parameter to specify 64 bytes of user-defined data.

For a CREATENTRY request, the field contains user-defined data to be written to the record data entry.

For a READENTRY request, the field will contain the data returned from the record data entry.

For an UPDATENTRY request, the field contains the data with which to update the record data entry.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 64-character input/output field to contain the user-defined data for a record data entry.

**,RESTOKEN=restoken**

Use this input/output parameter to specify a containing a standard restart token that must be passed on READALL, READBYCONN FASTPATH=NO, and DELETEBYCONN requests. Requesters that specify or default to IXLCONN ALLOWAUTO=NO must use the standard restart token. Requesters that specify IXLCONN ALLOWAUTO=YES must use the extended restart token (EXTRESTOKEN).

You must provide an initial value of zero in the field. The system then initializes the field with control information relevant to the READ or DELETE request. On each subsequent READ or DELETE request, the system uses this information. You must not modify the contents of this field.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an eight-character input/output field to contain control information.

**,RETCODE=retcode**

Use this output parameter to specify the location in which the system is to copy the return code from GPR 15.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword to contain the return code.

**,RSNCODE=rsncode**

Use this output parameter to specify the location in which the system is to copy the reason code from GPR 0.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword to contain the reason code.

**,TCONNAME=NO\_TCONNAME****,TCONNAME=tconname**

Use this input parameter to specify the target connection name (CONNAME) of the connected user. If a target connection name is not specified, the record data is associated with the connected user identified by the CONTOKEN keyword.

For a CREATENTRY request, the record data entry that is to be created will be associated with the connected user identified by the TCONNAME keyword.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-character input field that contains the target connection name.

**,VERCONNAME=NO\_VERCONNAME****,VERCONNAME=verconname**

Use this input parameter to specify the name of a connected user to be verified as the connector associated with the record data entry that is to be read. If the connector indicated by the VERCONNAME is not associated with the entry specified by ENTRYID, the IXLRT request fails.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 12-character input field to contain the connection name to be verified.

## ABEND Codes

---

Abend X'026' (See [z/OS MVS System Codes](#) for more information on this abend.)

## Return and Reason Codes

---

When the IXLRT macro returns control to your program:

- GPR 15 (and *retcode*, you coded RETCODE) contains a return code.
- GPR 0 (and *rsncode*, if you coded RSNCODE) contains a reason code if applicable.

Macro IXLRT provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

**0**

IXLRETCODEOK



- 4** IXLRETCODEWARNING
- 8** IXLRETCODEPARMERROR
- C** IXLRETCODEENVERROR
- 10** IXLRETCODECOMPERROR

The following table identifies the hexadecimal return and reason codes and the equate symbol associated with each reason code.

<i>Table 88. Return and Reason Codes for the IXLRT Macro</i>		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0	None.	<b>Meaning:</b> IXLRT request successful. <b>Action:</b> None.
4	xxxx0409	<b>Equate Symbol:</b> IXLRSNCODETIMEOUT <b>Meaning:</b> <ul style="list-style-type: none"> <li>A READALL, READBYCONN, DELETEBYCONN, or DELETENTRYLST request has completed prematurely due to a model-dependent time-out condition. The number of reliable record data entries that has been read or deleted has been returned in the answer area.</li> <li>All other request types: Not applicable.</li> </ul> <b>Action:</b> Respecify the request by issuing it with the RESTOKEN, EXTRESTOKEN, or FASTRESTOKEN returned for READALL, READBYCONN, and DELETEBYCONN, or by adjusting FIRSTELEM on a DELETENTRYLST request.  Be sure to process the information returned from this request before reissuing the request. The data returned from this request will be overwritten if you specify the same buffer address. Continue to reissue the request until the return code indicates that all processing has completed.  For more information about premature request completion, see <a href="#">z/OS MVS Programming: Sysplex Services Guide</a> .

Table 88. Return and Reason Codes for the IXLRT Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	xxxx040F	<p><b>Equate Symbol:</b> IXLRNICODEBUFFERFULL</p> <p><b>Meaning:</b></p> <ul style="list-style-type: none"> <li>A READALL or READBYCONN request has completed prematurely due to a buffer full condition. The number of reliable record data entries that have been read has been returned in the answer area.</li> <li>All other request types: Not applicable.</li> </ul> <p><b>Action:</b> Reissue the request specifying RESTOKEN, EXTRESTOKEN, or FASTRESTOKEN and an empty DATAREA.</p> <p>Be sure to process the information returned from this request before reissuing the request. The data returned from this request will be overwritten if you specify the same buffer address. Continue to reissue the request until the return code indicates that all processing has completed.</p> <p>For more information about premature request completion, see <a href="#">z/OS MVS Programming: Sysplex Services Guide</a>.</p>
8	xxxx0801	<p><b>Equate Symbol:</b> IXLRNICODEBADPARMLIST</p> <p><b>Meaning:</b> Program error. The parameter list is either not addressable or not accessible.</p> <p><b>Action:</b> Verify that the parameter list address is valid.</p>
8	xxxx0804	<p><b>Equate Symbol:</b> IXLRNICODEBADVERSION#</p> <p><b>Meaning:</b> The version number in the parameter list is not valid.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>Verify that your program did not overlay the parameter list storage.</li> <li>Verify that your program was assembled with the correct macro library for the release of MVS on which your program is running.</li> </ul>
8	xxxx0807	<p><b>Equate Symbol:</b> IXLRNICODENOTENABLED</p> <p><b>Meaning:</b> Program error. Caller is not enabled.</p> <p><b>Action:</b> Verify that the program is enabled for I/O and external interrupts.</p>
8	xxxx080A	<p><b>Equate Symbol:</b> IXLRNICODEBADCONTOKEN</p> <p><b>Meaning:</b> Invalid CONTOKEN specified. The contoken is invalid for one of the following reasons: disconnect has occurred, EOT of the connector's task, input contoken is not the contoken returned from IXLCONN, the request was issued outside the connector's address space, or the contoken has been invalidated for rebuild.</p> <p><b>Action:</b> Verify that the CONTOKEN value specified is valid and for the correct structure.</p>

Table 88. Return and Reason Codes for the IXLRT Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx080B	<b>Equate Symbol:</b> IXLRSNCODEBADCONNAME <b>Meaning:</b> Program error. The TCONNAME or VERCONNAME specified was not valid. <b>Action:</b> Ensure that the TCONNAME or VERCONNAME was specified correctly.
8	xxxx080E	<b>Equate Symbol:</b> IXLRSNCODEBADAREA <b>Meaning:</b> Program error. The answer area specified by ANSAREA is either not addressable or not accessible. <b>Action:</b> Ensure that the address specified for ANSAREA is valid.
8	xxxx0816	<b>Equate Symbol:</b> IXLRSNCODENORTE EXISTS <b>Meaning:</b> Program error. There are no record data entries allocated. <b>Action:</b> Ensure that RECORD=YES was specified on the IXLCONN macro to provide recording.
8	xxxx0818	<b>Equate Symbol:</b> IXLRSNCODENOTLOCKSTR <b>Meaning:</b> Program error. The connection specified by CONTOKEN does not represent a lock structure. <b>Action:</b> Verify that the CONTOKEN is specified correctly and is for the correct structure.
8	xxxx081A	<b>Equate Symbol:</b> IXLRSNCODENORTENTRY <b>Meaning:</b> Program error. A request to read, update, or delete a record data entry found no such entry allocated. <b>Action:</b> Verify that the record data entry specified is correctly identified.
8	xxxx082B	<b>Equate Symbol:</b> IXLRSNCODEBADIDINDEX <b>Meaning:</b> Program error. A DELETEENTRYLIST request had an invalid index specified, either by FIRSTLEM or LASTLEM, for the first or last element in the element list. The RTAADELCNT and RTAAFAILINDEX fields in the RTAA will contain the count of elements deleted and the index of the failing entry, respectively. <b>Action:</b> Update FIRSTLEM to point past the failing entry and re-issue the request.
8	xxxx0835	<b>Equate Symbol:</b> IXLRSNCODEBADDATAADDR <b>Meaning:</b> Program error. The storage area specified by DATAREA or ENTRYIDLIST is not addressable. <b>Action:</b> Ensure that the address specified for DATAREA or ENTRYIDLIST is valid.

Table 88. Return and Reason Codes for the IXLRT Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx083D	<p><b>Equate Symbol:</b> IXLRSNCODEBADANSLEN</p> <p><b>Meaning:</b> Program error. The length of the answer area, as specified by ANSLEN, is not sufficient for providing answer area information.</p> <p><b>Action:</b> Determine the length of the answer area and correct the value specified in ANSLEN.</p>
8	xxxx0849	<p><b>Equate Symbol:</b> IXLRSNCODEBADRESTOKEN</p> <p><b>Meaning:</b> Program error.</p> <ul style="list-style-type: none"> <li>• A READALL, READBYCONN, or DELETEBYCONN request specified a restart token that was not valid. Possible causes are: <ul style="list-style-type: none"> <li>– The specified token does not correspond to a previous prematurely-completed request.</li> <li>– The user specified RESTOKEN when EXTRESTOKEN was required.</li> <li>– The user specified EXTRESTOKEN when RESTOKEN was required.</li> </ul> </li> <li>• All other request types: Not applicable.</li> </ul> <p><b>Action:</b> Ensure that you specified the correct restart token and that you have not modified it.</p>
8	xxxx0855	<p><b>Equate Symbol:</b> IXLRSNCODEENTRIESCHANGED</p> <p><b>Meaning:</b> Program error. The record table entry that was represented by the FASTRESTOKEN was deleted or reacquired between IXLRT REQUEST=READBYCONN FASTPATH=YES requests.</p> <p><b>Action:</b> The request cannot be processed. Reset the value of FASTRESTOKEN to zero and restart the IXLRT process from the beginning.</p>
8	xxxx0887	<p><b>Equate Symbol:</b> IXLRSNCODEBADEXTRESTOKEN</p> <p><b>Meaning:</b> Program error.</p> <ul style="list-style-type: none"> <li>• A READALL, READBYCONN, or DELETEBYCONN request specified an extended restart token that was not valid. The specified token refers to an older instance of the target structure.</li> <li>• All other request types: Not applicable.</li> </ul> <p><b>Action:</b> Reset the value of EXTRESTOKEN to zero and resubmit the IXLRT request.</p>

Table 88. Return and Reason Codes for the IXLRT Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx08A8	<p><b>Equate Symbol:</b> IXLSNCODEBADMRDLEVEL</p> <p><b>Meaning:</b> Program error. The value specified for MRDLEVEL was not valid.</p> <p><b>Action:</b> Correct your program so that it specifies a valid value for MRDLEVEL.</p>
C	xxxx0C06	<p><b>Equate Symbol:</b> IXLSNCODENOCNN</p> <p><b>Meaning:</b> Environmental error. No connectivity to structure containing record data. This may occur due to operator commands such as VARY PATH OFFLINE or CONFIG CHP OFFLINE or hardware errors such as coupling facility or path failures. The contoken will be invalidated.</p> <p><b>Action:</b> Disconnect from the structure or rebuild.</p>
C	xxxx0C0B	<p><b>Equate Symbol:</b> IXLSNCODERTFULL</p> <p><b>Meaning:</b></p> <ul style="list-style-type: none"> <li>• The record portion of the lock structure is full and cannot accommodate the CREATENTRY request. This reason code may also be issued when the creation of a record data entry would result in the percentage of free entries being less than the percent entry reserved value in effect for the structure.</li> <li>• All other request types: Not applicable.</li> </ul> <p><b>Action:</b> Rebuild or alter the structure to allow for more record data entries. If the structure has a percent entry reserved threshold established, specify the ALLOWUSERSV keyword on the IXLRT request.</p>
C	xxxx0C13	<p><b>Equate Symbol:</b> IXLSNCODEREQPURGED</p> <p><b>Meaning:</b> Environmental error. The request was purged prior to completion of the request. Possible reasons include:</p> <ul style="list-style-type: none"> <li>• The connector failed.</li> <li>• The connector disconnected.</li> <li>• The requestor failed.</li> <li>• The request was purged by IXLPURGE.</li> <li>• Requests were purged when the connector provided an IXLEERSP response for the Rebuild Stop or Rebuild Cleanup event.</li> <li>• The secondary address space was no longer valid.</li> </ul> <p><b>Action:</b> None if this is expected. Otherwise, determine why the connector failed.</p>

Table 88. Return and Reason Codes for the IXLRT Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C25	<b>Equate Symbol:</b> IXLRNCODESTRFAILURE <b>Meaning:</b> Environmental error. Prior to the completion of the request, the lock structure failed. <b>Action:</b> Attempt to rebuild the structure using IXLREBLD or disconnect from the structure using IXLDISC.
C	FFFFFFFF	<b>Meaning:</b> XES functions are not available. This can occur because the coupling facility hardware necessary to provide XES function is not present. <b>Action:</b> Re-IPL the system, or follow your particular management protocol.
10	xxxx10xx	<b>Meaning:</b> Failure in XES processing. The state of the resource request is unpredictable. <b>Action:</b> Save the reason code information and contact the IBM support center.

## Chapter 83. IXLSYNCH – Synchronous update to a lock structure

### Description

The IXLSYNCH service enables a connected user to modify information in the notify exit parameter list (NEPL). The IXLSYNCH service is called from the notify exit to provide a synchronous update of the state and/or user data associated with a resource. When the notify exit returns control to the system, the updated data can be used to resolve resource contention.

### Environment

The following are the environment requirements for the caller.

Environment	Environment requirement
Minimum Authorization:	Supervisor state or PKM keys 0 - 7
Dispatchable unit mode:	SRB
Cross memory mode:	Any PASN, any HASN, any SASN. The primary address space must be the same as the primary address space at the time the connection service (IXLCONN) was issued.
AMODE:	31-bit
ASC mode:	Primary or Access Register (AR)
Interrupt status:	Enabled for I/O and external interrupts
Locks:	No locks held
Control parameters:	Must be in the primary address space

### Programming Requirements

The IXLSYNCH service is called from the notify exit to update information in the notify exit parameter list (NEPL). The IXLSYNCH service is the only method of successfully updating this information before the notify exit returns control to the system. If the connected user makes changes to the NEPL and does not invoke the IXLSYNCH service, the changes to the NEPL are ignored.

The IXLYNEPL macro provides the format of notify exit parameter list. Include that macro in your program.

### Restrictions and Limitations

The IXLSYNCH service is supported only when issued out of the notify exit.

### Input Register Information

Before issuing the IXLSYNCH macro, the caller does not have to place any information in any register unless using it in register notation for a particular parameter, or using it as a base register.

### Output Register Information

When control returns to the caller of the IXLSYNCH macro, the general purpose registers (GPRs) contain:

### Register Contents

- 0**  
Reason code if GPR15 return code is nonzero
- 1**  
Used as a work register by the system
- 2-13**  
Unchanged
- 14**  
Used as a work register by the system
- 15**  
Return code

When control returns to the caller of the *IXLSYNCH macro*, the access registers (ARs) contain:

### Register Contents

- 0-1**  
Used as a work register by the system
- 2-13**  
Unchanged
- 14-15**  
Used as a work register by the system

**For registers that the system changes**, a caller who depends on these registers containing the same value before and after issuing the macro must save these registers before issuing the macro, and restore them after the system returns control.

## Performance Implications

---

None.

## Understanding IXLSYNCH Version Support

---

The IXLSYNCH macro supports version 0 keywords and functions.

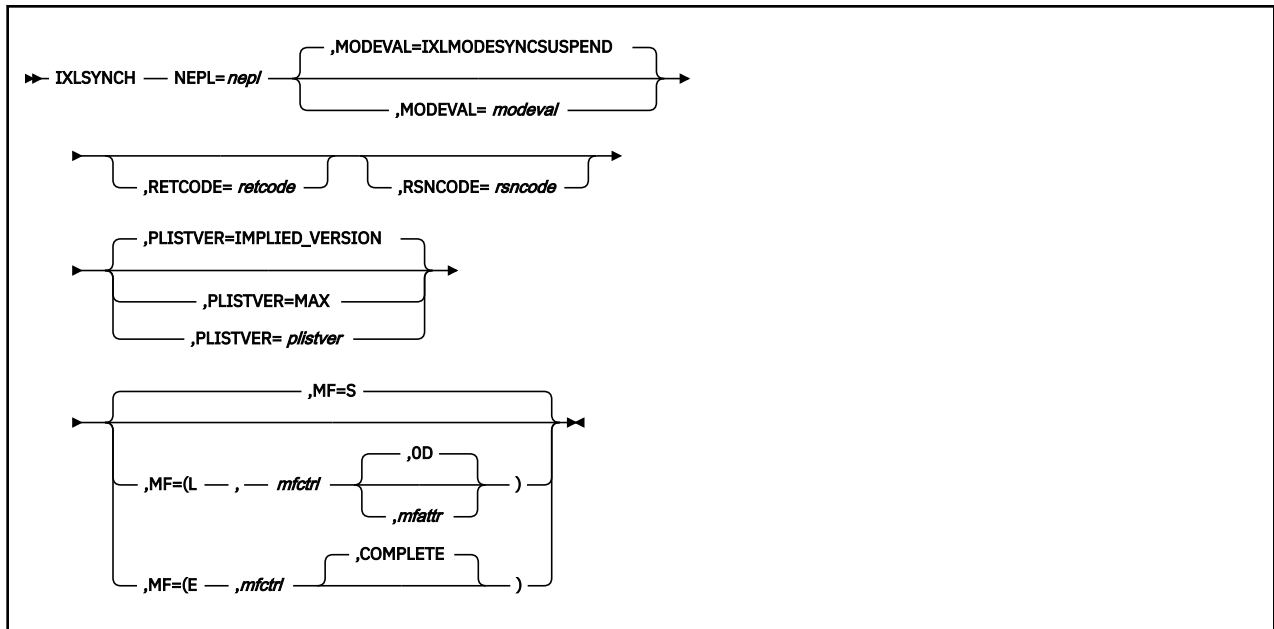
Specify the version of the parameter list that you want generated with the PLISTVER keyword. See [Chapter 2, “Specifying a Macro Version Number,” on page 5](#) for considerations when specifying the version of the parameter list with PLISTVER.

## Syntax Diagram

---

The syntax of the IXLSYNCH macro is as follows:





## Parameter descriptions

The parameter descriptions are listed in alphabetical order. Default values are underlined:

**,MF=S**

**,MF=(L,mfctrl)**

**,MF=(L,mfctrl,mfattr)**

**,MF=(L,mfctrl,OD)**

**,MF=(M,mfctrl)**

**,MF=(M,mfctrl,COMPLETE)**

**,MF=(M,mfctrl,NOCHECK)**

**,MF=(E,mfctrl)**

**,MF=(E,mfctrl,COMPLETE)**

**,MF=(E,mfctrl,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

**,mfctrl**

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

**,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of OD, which forces the parameter list to a doubleword boundary.

**,COMPLETE****,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

**COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if `SMILE=var` were an optional parameter and the default is `SMILE=NO_SMILE` then it would not be documented. However, if the default was `SMILE=:-)`, then it would be documented because a value would be the default.

**NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

**NEPL=nepl**

Use this input parameter to identify the notify exit parameter list (IXLYNEPL). The parameter list must be the actual list passed and not a copy of the list.

Upon successful completion of the IXLSYNCH request, the parameter list might be updated to include additional data. Consult the IXLYNEPL mapping macro for information on the fields set by the IXLSYNCH service.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the parameter list that was passed to the notify exit in register 1.

**,MODEVAL=IXLMODESYNCSUSPEND****,MODEVAL=modeval**

Use this input parameter to specify how the request should be processed if it cannot be serviced immediately. If the request can be processed immediately, the MODEVAL keyword is ignored and control is returned to the invoker with all information about the completed request.

The value provided must be equivalent to the constants provided in the IXLYCON macro indicating the mode. (See the IXLYCON macro for a list of mode value constants for users of IXLSYNCH and optionally include that macro in your program.)

**Note:** Only MODEVAL values `IxlModeSyncSuspend` and `IxlModeSyncFail` are currently supported for IXLSYNCH requests.

If you specify a value other than one of the IXLYCON constants that is valid for an IXLSYNCH request, the request fails with reason code `IXLRSNCODEBADMODEVAL`.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of an optional byte input.

**Default:** `IXLMODESYNCSUSPEND`

**,PLISTVER=IMPLIED\_VERSION****,PLISTVER=MAX****,PLISTVER=plistver**

Use this input parameter to specify the version of the macro. See [“Understanding IXLSYNCH Version Support” on page 1584](#) for a description of the options available with PLISTVER.

**,RETcode=retcode**

Use this output parameter to specify the location in which the system is to copy the return code from GPR 15.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword to contain the return code.

**,RSNCODE=rsncode**

Use this output parameter to specify the location in which the system is to copy the reason code from GPR 0.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword to contain the reason code.

## ABEND Codes

---

Abend X'026' (See [z/OS MVS System Codes](#) for more information on this abend.)

## Return and Reason Codes

---

When the IXLSYNCH macro returns control to your program,

- GPR 15 (and *retcode*, if you coded RETCODE) contains a return code
- GPR 0 (and *rsncode*, if you coded RSNCODE) contains a reason code if applicable.

Macro IXLYCON provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code as follows:

**0**

IXLRETCODEOK

**4**

IXLRETCODEWARNING

**8**

IXLRETCODEPARMERROR

**C**

IXLRETCODEENVEERROR

**10**

IXLRETCODECOMPERROR

The following table identifies the hexadecimal return and reason codes and the equate symbol associated with each reason code.

Table 89. Return and Reason Codes for the IXLSYNCH Macro		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0	None.	<b>Meaning:</b> IXLSYNCH request successful. Requested changes are reflected in the corresponding system's local structure. <b>Action:</b> None
4	xxxx041E	<b>Equate Symbol:</b> IXLRSNCODESYNCHRTNOTDELETED <b>Meaning:</b> The resource was released through IXLSYNCH. However, a record data element could not be deleted. <b>Action:</b> None expected.

Table 89. Return and Reason Codes for the IXLSYNCH Macro (continued)		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0804	<b>Equate Symbol:</b> IXLRSNCODEBADVERSION# <b>Meaning:</b> The version number in the parameter list is not valid. <b>Action:</b> <ul style="list-style-type: none"> <li>Verify that your program did not overlay the parameter list storage.</li> <li>Verify that your program was assembled with the correct macro library for the release of MVS that your program is running on.</li> </ul>
8	xxxx0807	<b>Equate Symbol:</b> IXLRSNCODENOTENABLED <b>Meaning:</b> IXLSYNCH request unsuccessful. The caller is not enabled. <b>Action:</b> Verify that the program is enabled for I/O and external interrupts.
8	xxxx080A	<b>Equate Symbol:</b> IXLRSNCODEBADCONTOKEN <b>Meaning:</b> Invalid CONTOKEN specified. The contoken is invalid for one of the following reasons: disconnect has occurred, EOT of the connector's task, input contoken is not the contoken returned from IXLCONN, the request was issued outside the connector's address space, or the contoken has been invalidated for rebuild. <b>Action:</b> Verify that the CONTOKEN value specified is valid and for the correct structure.
8	xxxx0810	<b>Equate Symbol:</b> IXLRSNCODERESOURCENOTFOUND <b>Meaning:</b> IXLSYNCH request unsuccessful. Resource that caused the notify exit to be given control has been released by a previous invocation of IXLSYNCH. <b>Action:</b> None expected. If necessary, attempt to regain ownership of the resource.
8	xxxx0811	<b>Equate Symbol:</b> IXLRSNCODESYNCHBADSTATE <b>Meaning:</b> IXLSYNCH request unsuccessful. Attempt to change STATE to a value other than shared, exclusive, or free. <b>Action:</b> Valid STATE values are shared, exclusive, and free. See IXLYCON.
8	xxxx0815	<b>Equate Symbol:</b> IXLRSNCODEBADNEPL <b>Meaning:</b> Specified NEPL is not valid. Note that the NEPL is no longer valid upon return from the notify exit to which it was presented. <b>Action:</b> Ensure that the NEPL address is specified correctly. Also, verify that the protocol does not attempt to reacquire the NEPL after returning from the notify exit.

Table 89. Return and Reason Codes for the IXLSYNCH Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0816	<p><b>Equate Symbol:</b> IXLRSNCODENORTEXISTS</p> <p><b>Meaning:</b> IXLSYNCH request unsuccessful. Record structure does not exist.</p> <p><b>Action:</b> Ensure that RECORD=YES was specified on the IXLCONN macro to request recording.</p>
8	xxxx0879	<p><b>Equate Symbol:</b> IXLRSNCODEBADMODEVAL</p> <p><b>Meaning:</b> Program error. The value specified for MODEVAL is not valid.</p> <p><b>Action:</b> Verify that the value specified for MODEVAL is one of the possible mode value constants provided in the IXLYCON macro and that it is a valid value for the IXLSYNCH request being processed.</p>
8	xxxx08A4	<p><b>Equate Symbol:</b> IXLRSNCODEBADNEPLORTACTION</p> <p><b>Meaning:</b> An invalid NEPL record data action (NeplORtAction) was specified. NeplORtAllowUseRsv was specified without indicating to write record data (NeplORtWrite).</p> <p><b>Action:</b> If NeplORtAllowUseRsv is set to '1'B', NeplORtWrite must also be set to '1'B to indicate that a write of record data is requested.</p>
C	xxxx0C06	<p><b>Equate Symbol:</b> IXLRSNCODENOCONN</p> <p><b>Meaning:</b> No connectivity to lock structure. This may occur due to operator commands such as VARY PATH OFFLINE or CONFIG CHP OFFLINE or hardware errors such as coupling facility or path failures. The contoken will be invalidated.</p> <p><b>Action:</b> Either disconnect from the structure or rebuild.</p>
C	xxxx0C0B	<p><b>Equate Symbol:</b> IXLRSNCODERTFULL</p> <p><b>Meaning:</b> Record structure full. This reason code may also be issued when the creation of a record data entry would result in the percentage of free entries being less than the percent entry reserved value in effect for the structure.</p> <p><b>Action:</b> If your protocol allows, attempt to rebuild the lock structure so that additional record data might be available.</p> <p>If the structure has a percent entry reserved threshold established, set NeplORtAllowUseRsv to '1'B in the Notify Exit Parameter List (IXLYNEPL) to indicate that the percent entry reserve threshold value should be ignored and a record data entry should be created as long as there is a free record table entry to satisfy the request.</p>

Table 89. Return and Reason Codes for the IXLSYNCH Macro (continued)		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C10	<p><b>Equate Symbol:</b> IXLRSNCODEOLDGLOBALMANAGERINSTANCE</p> <p><b>Meaning:</b> Environmental error. IXLSYNCH processing determined the global manager instance that initiated the call to the Notify Exit is no longer valid. The request was purged before processing the request.</p> <p><b>Action:</b> Updates are not made during IXLSYNCH processing. IXLSYNCH requestors should not perform any updates and undo any updates previously made in anticipation of IXLSYNCH completing successfully. If global manager responsibilities need to be reassigned, the new global manager instance can reinitiate a call to the contention exit with the CeplRecovery indication that can provide instructions about where again to notify all owners of the resource.</p>
C	xxxx0C13	<p><b>Equate Symbol:</b> IXLRSNCODEREQPURGED</p> <p><b>Meaning:</b> Environmental error. The request was purged prior to completion of the request. Possible reasons include:</p> <ul style="list-style-type: none"> <li>• The connector failed.</li> <li>• The connector disconnected.</li> <li>• The requestor failed.</li> <li>• The request was purged by IXLPURGE.</li> <li>• Requests were purged when the connector provided an IXLEERSP response for the Rebuild Stop or Rebuild Cleanup event.</li> <li>• The secondary address space was no longer valid.</li> </ul> <p><b>Action:</b> None if this is expected. Otherwise, determine why the connector failed.</p>
C	xxxx0C25	<p><b>Equate Symbol:</b> IXLRSNCODESTRFAILURE</p> <p><b>Meaning:</b> Prior to completion of the request, the lock structure failed.</p> <p><b>Action:</b> Attempt to rebuild the structure using IXLREBLD or disconnect from the structure using IXLDISC.</p>
C	xxxx0C69	<p><b>Equate Symbol:</b> IXLRSNCODENODELAY</p> <p><b>Meaning:</b> Environmental error. An IXLSYNCH request in which the user specified MODEVAL=IXLMODESYNCFail encountered a delay. The request is canceled.</p> <p><b>Action:</b> Retry request at a later time.</p>

Table 89. Return and Reason Codes for the IXLSYNCH Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	FFFFFFFF	<b>Equate Symbol:</b> IXLRSNCODENOTAVAILABLE <b>Meaning:</b> Locking functions are not available. This can occur because the coupling facility hardware necessary to provide locking function is not present. <b>Action:</b> Re-IPL the system, or follow your particular failure management protocol.
10	xxxx10xx	<b>Meaning:</b> Failure in XES processing. The state of the resource request is unpredictable. <b>Action:</b> Save the reason code information and contact the IBM support center.





## Chapter 84. IXLUSYNC – Synchronizing Processing for User-Defined Events

### Description

Use the IXLUSYNC macro to synchronize the processing of user-defined events for multiple connectors to a given structure. IXLUSYNC allows users to establish “sync points” to indicate that all active connectors to a structure have confirmed that processing for an event is complete. Connectors are notified when a sync point is reached by means of the User Sync Point event being presented to their event exit.

You can use IXLUSYNC:

- To define a value to be associated with a user-defined event, the processing for which is to be synchronized among all active connectors.
- To confirm that the processing related to an event has been completed. When all active connectors have confirmed the event, the sync point is reached.
- To confirm that the processing associated with an event has completed *and* then to set a new event for which processing is to be synchronized.

You can also use IXLUSYNC to provide a response on behalf of a peer connector that has disconnected or failed prior to responding to a user-defined event.

Only one user-defined event can be set at a time for the active connectors to a structure.

### Environment

The following are the environment requirements for the caller.

Environment	Environment requirement
Authorization:	Supervisor state or PKM keys 0 - 7
Dispatchable unit mode:	Task
Cross memory mode:	PASN=HASN; any SASN; any HASN. The primary address space must be equal to the requestor's primary address space at the time of the connection.
AMODE:	31-bit
ASC mode:	Primary or access register (AR)
Interrupt status:	Enabled for I/O and external interrupts
Locks:	No locks held, with no enabled, unlocked task (EUT) FRRs established
Control parameters:	Must be in the primary address space or be in an address/data space that is addressable through a public entry on the caller's dispatchable unit access list (DU-AL)

### Restrictions

None.

## Input Register Information

---

Before issuing the IXLUSYNC macro, the caller does not have to place any information into any register unless using it in register notation for a particular parameter, or using it as a base register.

## Output Register Information

---

When control returns to the caller of the IXLUSYNC macro, the general purpose registers (GPRs) contain:

### Register Contents

**0**

Reason code, if applicable, if GPR 15 return code is nonzero

**1**

Used as work register by the system

**2-13**

Unchanged

**14**

Used as work register by the system

**15**

Return code

When control returns to the caller of the IXLUSYNC macro, the access registers (ARs) contain:

### Register Contents

**0-1**

Used as work registers by the system

**2-13**

Unchanged

**14-15**

Used as work registers by the system

**For registers that the system changes,** a caller who depends on these registers containing the same value before and after issuing the macro must save these registers before issuing the macro and restore them after the system returns control.

## Programming Requirements

---

If the program is in AR mode, issue the SYSSTATE ASCENV=AR macro before IXLUSYNC. SYSSTATE ASCENV=AR tells the system to generate code appropriate for AR mode.

## Performance Implications

---

IXLUSYNC processing requires that all active connections reach a sync point before proceeding. The performance of IXLUSYNC depends on the environment, the number of connections to the structure, and the processing required to reach the sync point. Also, depending on the protocols of connections during IXLUSYNC, the services provided by those connections may be temporarily unavailable.

IXLUSYNC processing might involve referencing or updating the CFRM active policy to reflect the operation that has been requested. When invoking this macro, be aware of the I/O to the CFRM couple data set that might be generated.

## Understanding IXLUSYNC Version Support

---

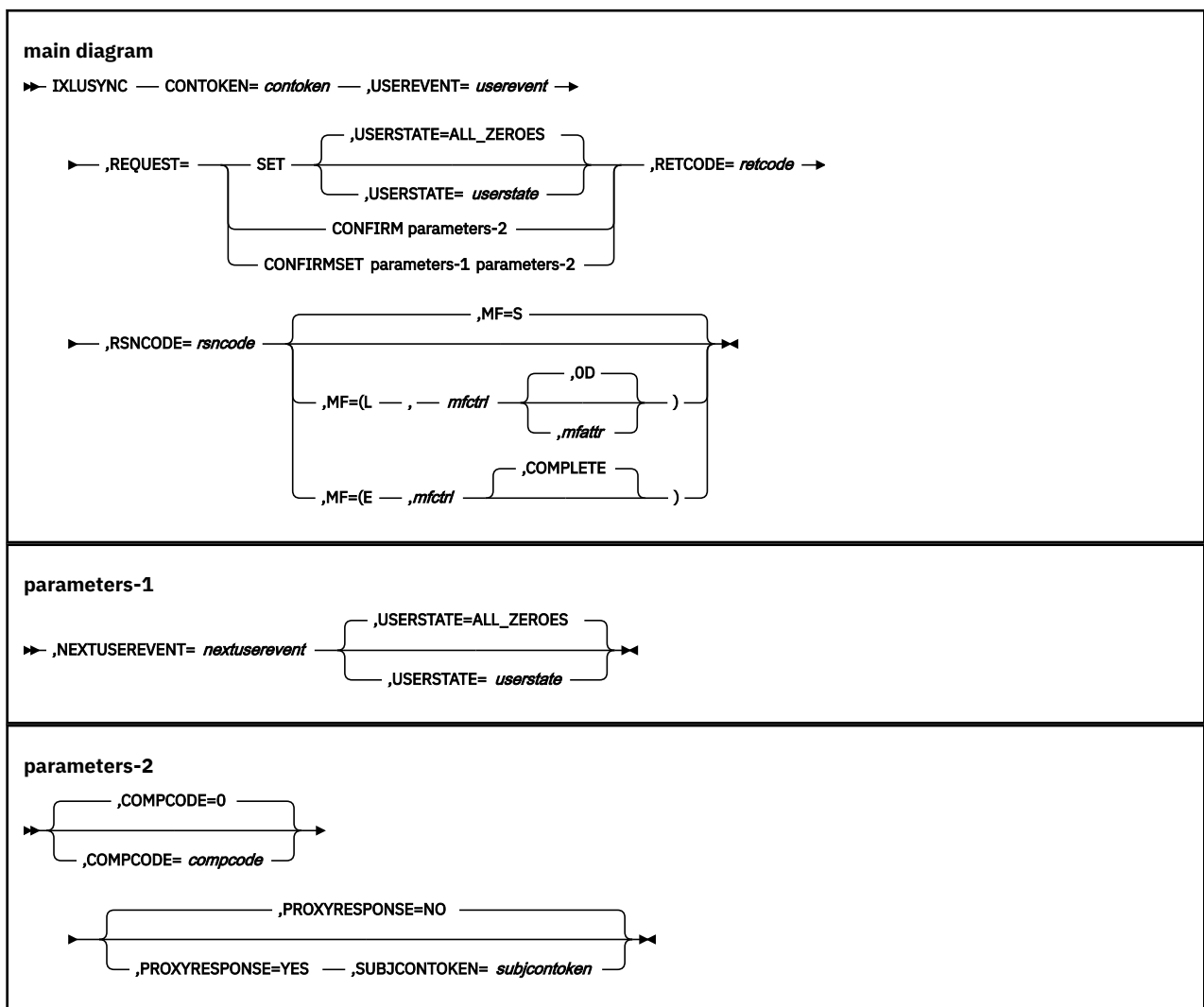
The IXLUSYNC macro supports versions in the range of 0 - 2.

- Keywords not specifically noted here are supported by version 0 and subsequent versions of the IXLUSYNC macro.
- The following keywords are supported by version 1 and subsequent versions of the IXLUSYNC macro.
  - PROXYRESPONSE
  - SUBJCONTOKEN
- The following keyword is supported by all versions starting with version 2 and higher of the IXLUSYNC macro.
  - COMPCODE

Specify the version of the parameter list that you want generated with the PLISTVER keyword. See Chapter 2, “Specifying a Macro Version Number,” on page 5 for considerations when specifying the version of the parameter list with PLISTVER.

## Syntax Diagram

The syntax of the IXLUSYNC macro is as follows:



## Parameter Descriptions

The parameter descriptions are listed in alphabetical order. Default values are underlined>.

**,COMPCODE=0****,COMPCODE=compcode**

Use this input parameter to specify a user-defined completion code value. When the sync point is reached, the system presents the highest completion code that was set by any confirming connector to the event exit. The system provides a default value of zero if you do not specify a completion code value.

For connectors that fail or disconnect at a time that they owe a user sync point confirmation, the system confirms the sync point with a completion code of IXLUSYNCFAILEDUSERCOMPCODE (X'0000FFFF'). Thus, if a given user completion code is to take precedence over the completion code set by the system, the user completion code must be greater than X'0000FFFF'. Similarly, if the completion code set by the system is to take precedence over a user completion code, the user completion code must be less than IXLUSYNCFAILEDUSERCOMPCODE (X'0000FFFF').

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the fullword input field that contains the user-defined completion code value.

**,CONTOKEN=contoken**

Use this input parameter to specify the original connect token returned to the requestor from the IXLCONN macro. The IXLCONN macro allows the requestor to connect to the structure.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the 16-character input field that contains the connect token returned to the requestor when connecting to the structure with IXLCONN.

**,MF=S****,MF=(L,mfctrl)****,MF=(L,mfctrl,mfattr)****,MF=(L,mfctrl,0D)****,MF=(M,mfctrl)****,MF=(M,mfctrl,COMPLETE)****,MF=(M,mfctrl,NOCHECK)****,MF=(E,mfctrl)****,MF=(E,mfctrl,COMPLETE)****,MF=(E,mfctrl,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

**,mfctrl**

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

**,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

**,COMPLETE****,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

**COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if `SMILE=var` were an optional parameter and the default is `SMILE=NO_SMILE` then it would not be documented. However, if the default was `SMILE=-)`, then it would be documented because a value would be the default.

**NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

**,NEXTUSEREVENT=nextuserevent**

Use this input/output parameter to specify a value associated with the next user event to be set. The NEXTUSEREVENT value must be non-zero.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword input/output field to contain the value associated with the next user event.

**,PROXYRESPONSE=NO****,PROXYRESPONSE=YES**

Use this input parameter to indicate whether this response is being provided on behalf of a failing connector.

**NO**

Indicates that this is not a proxy response. The response is on behalf of the connector described by the CONTOKEN keyword.

**YES**

Indicates that this is a proxy response. The response is being provided on behalf of the connector described by the SUBJCONTOKEN keyword.

**Note:** To ensure that the PROXYRESPONSE feature is installed on the system on which you are running, issue `IXCQUERY REQINFO=FEATURES`. `QUREQRFPROXYRESPONSE`, returned from the `IXCQUERY` request, indicates whether the PROXYRESPONSE feature is installed.

**,RETCODE=retcode**

Use this output parameter to specify the location in which the system is to copy the return code from GPR 15.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword to contain the return code.

**,REQUEST=SET****,REQUEST=CONFIRM****,REQUEST=CONFIRMSET**

Use this input parameter to identify the type of user sync point request. You can set a user event, confirm that processing is complete for a user event, or confirm that processing is complete and then set the next user event. Only one user event can be set at a time.

**SET**

Set the user event. When setting an event, a previous event, if there was one, must have been confirmed by all connectors and all connectors must have been told about the completion of the user sync point through their event exit. If the user event is successfully set, the User Sync Point event is presented to the event exit of each connector. The following information is presented to each connector in the event exit parameter list (IXLYEEPL):

- `EEPLCOMPLETEDUSEREVENT` — set to zero.

- EEPLNEXTUSEREVENT — set to the value specified by the USEREVENT keyword.
- EEPLCOMPLETEDUSERSTATE — set to zero.
- EEPLNEXTUSERSTATE — set to the value specified by the USERSTATE keyword or zero if USERSTATE is not specified.
- EEPLCOMPLETEDUSERCOMPCODE — set to zero.

**CONFIRM**

Confirm a user event specified by the USEREVENT keyword. When all connectors have confirmed the event and their confirmations have been received, the system presents the User Sync Point event to the event exit of each connector to indicate that a user sync point has been reached. The following information is presented to each connector in the event exit parameter list (IXLYEEPL):

- EEPLCOMPLETEDUSEREVENT — set to the value specified by the USEREVENT keyword.
- EEPLNEXTUSEREVENT — set to zero.
- EEPLCOMPLETEDUSERSTATE — set to the value specified when the event was set (with the USERSTATE keyword).
- EEPLNEXTUSERSTATE — set to zero.
- EEPLCOMPLETEDUSERCOMPCODE — set to the highest user completion code value that was specified by any confirming connector. Note that if a connector fails or disconnects while XES is expecting a user sync point confirmation from that connector, XES confirms the event for the failing connector with a completion code of IXLUSYNCF FAILEDUSERCOMPCODE (X'0000FFFF').

**CONFIRMSET**

Confirm the user event specified by the USEREVENT keyword, and if this is the last confirmation for the event, set the next user event to the value specified by the NEXTUSEREVENT keyword. If this is not the last confirmation for the event, the “set” is ignored.

When all confirmations have been received, the system presents the User Sync Point event to the event exit of each connector indicating that a sync point has been reached. In the same parameter list, the next user event also is presented to each connector. The following information is presented to each connector in the event exit parameter list (IXLYEEPL):

- EEPLCOMPLETEDUSEREVENT — set to the value specified by the USEREVENT keyword.
- EEPLNEXTUSEREVENT — set to the value specified by the NEXTUSEREVENT keyword.
- EEPLCOMPLETEDUSERSTATE — set to the user state specified by the connector that set the original event.
- EEPLNEXTUSERSTATE — set to the value specified by the USERSTATE keyword or zero if USERSTATE is not specified.
- EEPLCOMPLETEDUSERCOMPCODE — set to the highest user completion code value that was specified by any confirming connector. Note that if a connector fails or disconnects while XES is expecting a user sync point confirmation from that connector, XES confirms the event for the failing connector with a completion code of IXLUSYNCF FAILEDUSERCOMPCODE (X'0000FFFF').

**,RSNCODE=rsncode**

Use this output parameter to specify the location in which the system is to copy the reason code from GPR 0.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword to contain the reason code.

**,SUBJCONTOKEN=subjcontoken**

Use this input parameter to identify the connector for whom the response is being provided. The subject connector must be in the failing state.

Do not provide a proxy response on behalf of a failing connector until AFTER the EEPLDISCFAILCONNECTION event for the subject user has been presented to your event exit. You can use the value of the EEPLSUBJCONTOKEN field from that invocation of the event exit as input for this parameter.

Note that if the subject connector is not in the failing state, the IXLUSYNC request will fail with the IXLRSNCODESUBJCONNNOTFAILING reason code.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the 16-character input field that contains the connect token for which the response is being provided.

**,USEREVENT=*userevent***

Use this input parameter to specify the user event associated with this request. The value of USEREVENT must be non-zero.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the fullword input/output field that contains the value associated with the user event.

**,USERSTATE=ALL\_ZEROES**

**,USERSTATE=*userstate***

Use this input/output parameter to specify a user-defined value to be presented to the event exit. If this value is not provided, the field is set to the default of all zeroes. The user state presented to the event exit is that set by the connector that successfully set the user event.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 32-character input/output field that contains the user-defined value associated with the event that is to be presented to the event exit.

## Return and Reason Codes

When the IXLUSYNC macro returns control to your program:

- GPR 15 (and *retcode*, if you coded RETCODE) contains a return code.
- GPR 0 (and *rsncode*, if you coded RSNCODE) contains a reason code, if applicable.

Macro IXLYCON provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

<b>0</b>	IXLRETCODEOK
<b>4</b>	IXLRETCODEWARNING
<b>8</b>	IXLRETCODEPARMERROR
<b>C</b>	IXLRETCODEENVERROR
<b>10</b>	IXLRETCODECOMPERROR

The following table identifies the hexadecimal return and reason codes and the equate symbol associated with each reason code.

Table 90. Return and Reason Codes for the IXLUSYNC Macro		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
0	None.	<b>Meaning:</b> Processing completed successfully. <b>Action:</b> None.

Table 90. Return and Reason Codes for the IXLUSYNC Macro (continued)		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
4	xxxx0417	<p><b>Equate Symbol:</b> IXLRSNCODENOTLASTCONFIRMATION</p> <p><b>Meaning:</b> Request to confirm the synchronization point has completed, but all connection confirmations have not yet been received. The next synchronization point has not yet been set.</p> <p>Applies only to REQUEST=CONFIRMSET</p> <p><b>Action:</b> Request that the next event be set when all confirmations have been received for this event.</p>
4	xxxx0420	<p><b>Equate Symbol:</b> IXLRSNCODEUSYNCEVENTSET</p> <p><b>Meaning:</b> The user event specified has already been set by a peer connector.</p> <p><b>Action:</b> Ensure that only one user-event is set at a time.</p>
8	xxxx0801	<p><b>Equate Symbol:</b> IXLRSNCODEBADPARMLIST</p> <p><b>Meaning:</b> The IXLUSYNC parameter list is not accessible.</p> <p><b>Action:</b> Verify that:</p> <ul style="list-style-type: none"> <li>• The parameter list is uncorrupted.</li> <li>• The parameter list is addressable in the caller's primary address space.</li> <li>• If you are invoking IXLUSYNC in AR-mode and you specified the parameter list address using explicit register notation, the corresponding access register must be updated appropriately.</li> <li>• If you are invoking IXLUSYNC in AR-mode, SYSSTATE ASCENV=AR must be issued before the IXLUSYNC.</li> </ul>
8	xxxx0802	<p><b>Equate Symbol:</b> IXLRSNCODEBADPARMLISTALET</p> <p><b>Meaning:</b> The IXLUSYNC parameter list ALET is not valid.</p> <p><b>Action:</b> Ensure that the ALET is zero or that the ALET represents a valid entry on the DU-AL.</p>
8	xxxx0804	<p><b>Equate Symbol:</b> IXLRSNCODEBADVERSION#</p> <p><b>Meaning:</b> There is an invalid version number in the IXLUSYNC parameter list.</p> <p><b>Action:</b></p> <ul style="list-style-type: none"> <li>• Verify that your program did not overlay the parameter list storage.</li> <li>• Verify that your program was assembled with the correct macro library for the release of MVS that your program is running on.</li> </ul>
8	xxxx0806	<p><b>Equate Symbol:</b> IXLRSNCODESRBMODE</p> <p><b>Meaning:</b> The requestor is in SRB mode.</p> <p><b>Action:</b> Do not issue the IXLUSYNC macro when running in SRB mode.</p>



Table 90. Return and Reason Codes for the IXLUSYNC Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	xxxx0807	<b>Equate Symbol:</b> IXLRNCCODENOTENABLED <b>Meaning:</b> The requestor is not in an enabled state. <b>Action:</b> Issue the IXLUSYNC macro when running enabled for I/O and external interrupts.
8	xxxx0809	<b>Equate Symbol:</b> IXLRNCCODEPRIMARYNOTHOME <b>Meaning:</b> The primary address space does not equal the home address space. The request fails. <b>Action:</b> Make sure that the primary address space and the home address space are the same at the time of the IXLUSYNC invocation.
8	xxxx080A	<b>Equate Symbol:</b> IXLRNCCODEBADCONTOKEN <b>Meaning:</b> The requestor specified a CONTOKEN that is not valid. The CONTOKEN is invalid for one of the following reasons: disconnect has occurred, EOT of the connector's task, the input contoken is not the original contoken returned from IXLCONN, or the request was issued outside the connector's address space. <b>Action:</b> Verify that the CONTOKEN value specified was valid and for the correct structure.
8	xxxx0858	<b>Equate Symbol:</b> IXLRNCCODEBADUSEREVENT <b>Meaning:</b> The value specified for the USEREVENT or NEXTUSEREVENT keywords must be non-zero. <b>Action:</b> Correct the value specified for the USEREVENT or NEXTUSEREVENT keywords and resubmit the request.
8	xxxx0863	<b>Equate Symbol:</b> IXLRNCCODETASKTERM <b>Meaning:</b> The system rejects the request because the requesting task is going through termination. IXLUSYNC cannot be issued from a resource manager. <b>Action:</b> Examine your protocol to ensure that IXLUSYNC is not issued from a resource manager.
C	xxxx0C37	<b>Equate Symbol:</b> IXLRNCCODEUSEREVENTMISMATCH <b>Meaning:</b> Environment error. When confirming an event with REQUEST=CONFIRM or REQUEST=CONFIRMSET, the user event specified does not match the currently defined user event. <b>Action:</b> Ensure that the value of USEREVENT was specified correctly.

Table 90. Return and Reason Codes for the IXLUSYNC Macro (continued)		
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C38	<p><b>Equate Symbol:</b> IXLRNCODEUSERMISMATCH</p> <p><b>Meaning:</b> Environmental error. The system did not expect a confirmation request from the responding connector. The connector probably has already confirmed the user event.</p> <p><b>Action:</b> Check code to determine why the confirmation request was issued more than once.</p> <p><b>This return code is not issued on systems at z/OS V1R8 and higher.</b></p>
C	xxxx0C3F	<p><b>Equate Symbol:</b> IXLRNCODECONNNOTDEFINED</p> <p><b>Meaning:</b> Environmental error. The requesting connection is not defined.</p> <p><b>Action:</b> Ensure that the IXLUSYNC request is issued by an active connector.</p>
C	xxxx0C40	<p><b>Equate Symbol:</b> IXLRNCODECONNNOTACTIVE</p> <p><b>Meaning:</b> Environmental error. The requesting connection is not active.</p> <p><b>Action:</b> Ensure that the IXLUSYNC request is issued by an active connector.</p>
C	xxxx0C48	<p><b>Equate Symbol:</b> IXLRNCODESUBJCONNNOTDEFINED</p> <p><b>Meaning:</b> Environmental error. The connection identified by SUBJCONTOKEN is in the not defined state.</p> <p><b>Action:</b> Verify that your parameter list has not been overlaid. Verify that SUBJCONTOKEN is from EEPLSUBJCONTOKEN from the EEPL passed to the event exit.</p>
C	xxxx0C4B	<p><b>Equate Symbol:</b> IXLRNCODEEUSYNCEVENTNOTSET</p> <p><b>Meaning:</b> Environmental error. When setting an event with REQUEST=SET, the system rejects the new user event because another event is still set. Either all confirmations have not been received for the current event or all connectors have not been notified of the previously completed event through the event exit.</p> <p><b>Action:</b> Only one event can be set at a time.</p>
C	xxxx0C4F	<p><b>Equate Symbol:</b> IXLRNCODEEUSYNCSNOEVENTSET</p> <p><b>Meaning:</b> Environmental error. A REQUEST=CONFIRM or REQUEST=CONFIRMSET is rejected because no user event is currently set.</p> <p><b>Action:</b> Ensure that REQUEST=SET is issued before REQUEST=CONFIRM or REQUEST=CONFIRMSET.</p>

Table 90. Return and Reason Codes for the IXLUSYNC Macro (continued)

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
C	xxxx0C6D	<p><b>Equate Symbol:</b> IXLRSNCODESUBJCONNNOTFAILING</p> <p><b>Meaning:</b> Environmental error. An attempt to respond by proxy on behalf of the connector identified by the value provided for the SUBJCONTOKEN keyword failed because that connector is not in the failing state.</p> <p><b>Action:</b> Ensure that you do not respond by proxy on behalf of a failing connector until after you have been presented with the EEPLDISCFAILCONNECTION event exit.</p>
C	FFFFFFFF	<p><b>Equate Symbol:</b> IXLRSNCODENOTAVAILABLE</p> <p><b>Meaning:</b> Environmental error. XES functions are not available. This can occur because the coupling facility hardware necessary to provide XES functions is not present.</p> <p><b>Action:</b> Re-IPL the system, or follow your particular failure management protocol.</p>
10	xxxx10xx	<p><b>Meaning:</b> XES processing has failed.</p> <p><b>Action:</b> Save the reason code information and contact the IBM support center.</p>



## Chapter 85. IXLVECTR – Check or Modify Local Cache Vector or List Notification Vector

### Description

List structure users can use the IXLVECTR macro to check or modify their list notification vectors. Cache structure users can use the IXLVECTR macro to check or modify their local cache vectors.

**Note:** The following information assumes that you are familiar with either a list or cache services macro and that you are using either a list or cache structure.

**If you are using a list structure:** Use the IXLVECTR macro to perform the following operations on your list notification vector.

- Modify the number of entries in your list notification vector.
- Test whether a list or an event queue is empty or nonempty.
- Test whether a list is full or not-full.
- Test a range of vector entries to determine whether the monitored objects are empty or non-empty, or full or not-full for lists.

Testing whether a vector entry for a list represents the full or non-full state of a list is available when the CFLEVEL of the coupling facility in which the list structure is allocated is CFLEVEL=22 or higher.

**If you are using a cache structure:** Use the IXLVECTR macro to perform the following operations on your local cache vector:

- Modify the number of entries in your local cache vector.
- Determine the validity of data items in your local cache buffer.
- Test a range of vector entries to determine whether local cache vector entries are valid or invalid.

**For more information:** The *z/OS MVS Programming: Sysplex Services Guide* contains a detailed description of the functions provided by the IXLVECTR macro as well as guidance for using the macro and examples of serialization protocols for use with the TESTLOCALCACHE request. Before using the IXLVECTR macro, be sure to read this information. The reference material provided here assumes you have done so.

### Environment

The following are the environment requirements for the caller.

Environment	Environment requirement
Minimum authorization:	For the MODIFYVECTORSIZE parameter, supervisor state. For all other parameters, problem state.
Dispatchable unit mode:	Task or SRB
Cross memory mode:	Any HASN, any PASN, any SASN
AMODE:	31-bit
ASC mode:	Primary or AR
Interrupt status:	Enabled or disabled for I/O and external interrupts
Locks:	If your program is disabled, you may hold the CPU lock. Otherwise, no locks may be held.

Environment	Environment requirement
Control parameters:	Control parameters must be in the primary address space

## Programming Requirements

---

1. If your program is in AR mode, issue SYSSTATE ASCENV=AR before you issue the IXLVECTR macro. ASCENV=AR causes the system to generate code appropriate for AR mode. For more information about the SYSSTATE macro, see *z/OS MVS Programming: Assembler Services Reference ABE-HSP*.
2. All virtual storage areas passed to the IXLVECTR macro must reside in your primary address space.
3. Include the IXLYCON mapping macro to generate the equate symbols for the return codes.
4. The correct use of the TESTLOCALCACHE request requires the implementation of a serialization protocol. Be sure to read the description of the necessary serialization protocol in *z/OS MVS Programming: Sysplex Services Guide*.

## Restrictions

---

Registers 2-6 are used by the macro expansion. Therefore, registers 2-6 should not be used as base registers that will be used to resolve macro parameter addresses.

## Input Register Information

---

Before issuing the IXLVECTR macro, the caller does not have to place any information into any register unless using it in register notation for a particular parameter, or using it as a base register.

## Output Register Information

---

When control returns to the caller, the GPRs contain:

### Register Contents

- 0-1**  
Unpredictable
- 2-13**  
Unchanged
- 14**  
Unpredictable
- 15**  
Return code

When control returns to the caller, the ARs contain:

### Register Contents

- 0-1**  
Used as work registers by the system
- 2-13**  
Unchanged
- 14-15**  
Used as work registers by the system

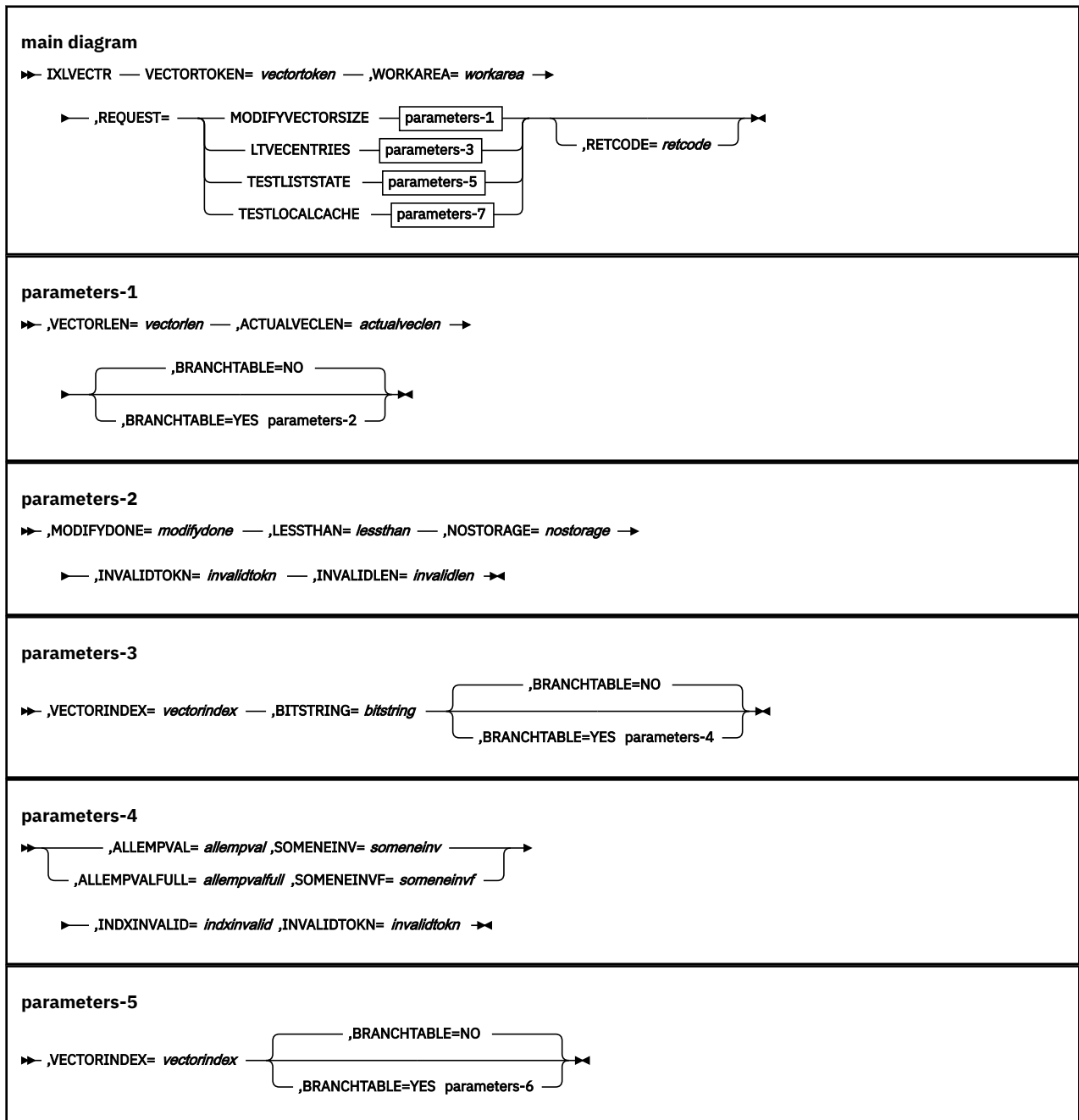
Some callers depend on register contents remaining the same before and after issuing a service. If the system changes the contents of registers on which the caller depends, the caller must save them before issuing the service, and restore them after the system returns control.

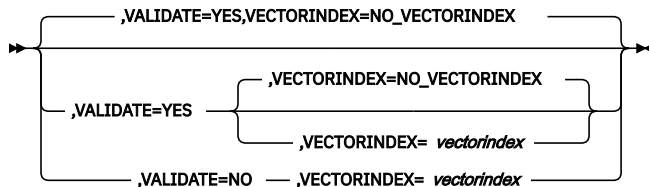
## Performance Implications

1. You should specify VALIDATE=YES only with the first vector index that you hold serialization for. The remaining entries, under the same serialization, should be checked with VALIDATE=NO. Performance is significantly slower with VALIDATE=YES.
2. The vector indexes are local to the CPC you are running on and do not reside on the coupling facility.

## Syntax Diagram

The syntax of the IXLVECTR macro is as follows:



**parameters-6****parameters-7**

## Parameter Descriptions

The parameter descriptions are listed in alphabetical order. Default values are underlined:

**,ACTUALVECLEN=actualveclen**

Use this output parameter to specify a fullword field to receive the count of the new number of entries for the list notification vector or local cache vector after your MODIFYVECTORSIZE request has been processed. This field is valid only when a return code of X'4' is returned.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the fullword field to contain the new vector length if it differs from the requested vector length.

**,ALLEMPVAL=allempval**

Use this input parameter to specify the label to branch to if all monitored objects (comprised of lists and/or the user's event queue) in the range of vector entries are EMPTY (list notification vector), or if all vector entries are VALID (local cache vector).

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the label.

**,ALLEMPVALFULL=allempvalfull**

Use this input parameter to specify the label to branch to if all monitored objects (comprising of lists and/or the user's event queue) in the range of vector entries are EMPTY when monitoring for empty/non-empty state or FULL when monitoring a list for full/non-full state (list notification vector), or if all vector entries are VALID (local cache vector).

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the label.

**,BITSTRING=bitstring**

BITSTRING is a required output field that contains vector entry state information for the range of vector entries specified. This field contains 32 bits. The first bit represents the first vector index entry specified on the VECTORINDEX parameter and continues up to a maximum of 32 vector index entries. The bits are interpreted as follows:

- 0 - the vector entry corresponding to this bit position indicates that the monitored list is NONEMPTY, when monitoring for empty/not-empty, or NONFULL, when monitoring for full/not-full. It can also indicate that the monitored event queue is NONEMPTY (list vector), or that the local cache entry is INVALID (local cache vector).
- 1 - the vector entry corresponding to this bit position indicates that the monitored list or event queue is EMPTY, when monitoring for empty/non-empty, or FULL, when monitoring for full/non-full. It can also indicate that the monitored event queue is EMPTY (list vector), or that the local cache entry is VALID (local cache vector).

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword field.



**,BRANCHTABLE=NO****,BRANCHTABLE=YES**

Use BRANCHTABLE=NO to specify that IXLVECTR should not generate a branch table.

With the MODIFYVECTORSIZE request, use BRANCHTABLE=YES to specify that IXLVECTR should generate a branch table using the labels you specify for MODIFYDONE, LESSTHAN, NOSTORAGE, INVALIDTOKN, and INVALIDLEN.

With the LTVECENTRIES request, use BRANCHTABLE=YES to specify that IXLVECTR should generate a branch table using the labels you specify for ALLEMPVAL, ALLEMPVALFULL, SOMENEINV, SOMENEINVNF, INDXINVALID, and INVALIDTOKN.

With the TESTLISTSTATE request, use BRANCHTABLE=YES to specify that IXLVECTR should generate a branch table for you using the labels you specify for LSTEMPTY, LSTFULL, LSTNONEMPTY, LSTNONFULL, INDXINVALID, and INVALIDTOKN.

**Note:** When BRANCHTABLE=YES is specified, the return code will only be available in register 15.

**,INDXINVALID=*indxinvalid***

Use this input parameter to specify the label to branch to if the IXLVECTR service detects that the index value you specified is not valid.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the label.

**,INVALIDLEN=*invalidlen***

Use this input parameter to specify the label to branch to if the value you specify using the VECTORLEN parameter is not valid.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the label.

**,INVALIDTOKN=*invalidtokn***

Use this input parameter to specify the label to branch to if the vector token you specified was not valid.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the label.

**,LESSTHAN=*lessthan***

Use this input parameter to specify the label to branch to if the IXLVECTR service cannot obtain sufficient storage to enlarge your list notification vector or local cache vector to the size you requested.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the label.

**,LSTEMPTY=*lstempty***

Use this input parameter to specify the label to branch to if the IXLVECTR service finds the list or event queue of interest empty.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the label.

**,LSTFULL=*lstfull***

Use this input parameter to specify the label to branch to if the IXLVECTR service finds the list of interest full. Use the LSTFULL and LSTNONFULL parameters when the list associated with the vector index that is specified on the VECTORINDEX parameter is being monitored for list full/not-full states.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the label.

**,LSTNONEMPTY=*lstnonempty***

Use this input parameter to specify the label to branch to if the IXLVECTR service finds the list or event queue of interest non-empty.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the label.

**,LSTNONFULL=*lstnonfull***

Use this input parameter to specify the label to branch to if the IXLVECTR service finds the list of interest non-full. Use the LSTFULL and LSTNONFULL parameters when the list associated with the vector index that is specified on the VECTORINDEX parameter is being monitored for list full/not-full states.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the label.

**,MODIFYDONE=modifydone**

Use this input parameter to specify the label to branch to if the IXLVECTR service is able to modify the list notification vector or local cache vector as requested.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the label.

**,NOSTORAGE=nostorage**

Use this input parameter to specify the label to branch to if the IXLVECTR service cannot obtain any storage to enlarge your list notification vector or local cache vector to the size you requested. The vector's size remains the same.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the label.

**,REQUEST=MODIFYVECTORSIZE**

**,REQUEST=LTVECENTRIES**

**,REQUEST=TESTLISTSTATE**

**,REQUEST=TESTLOCALCACHE**

Use REQUEST=MODIFYVECTORSIZE to modify the size of your list notification vector or local cache vector.

Use REQUEST=LTVECENTRIES to load and test a range of vector entries associated with a local vector for either a list or cache structure.

Use REQUEST=TESTLISTSTATE to test the following:

- Whether a list you are monitoring is empty or non-empty when monitoring for non-empty.
- Whether a list you are monitoring is full or non-full when monitoring for non-full.
- Whether an event queue you are monitoring is empty or non-empty.

Use REQUEST=TESTLOCALCACHE to test whether a data item in your local cache buffer is valid.

**,RETCODE=retcode**

Use this output parameter to specify a fullword field to contain the return code, which is also returned in register 15.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the fullword field.

**,SOMENEINV=someneinv**

Use this input parameter to specify the label to branch to if some monitored objects (comprised of lists and/or the user's event queue) in the range of vector entries are NON-EMPTY (list notification vector), or if some entries in the range of vector entries are INVALID (local cache vector).

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the label.

**,SOMENEINVf=someneinvf**

Use this input parameter to specify the label to branch to if some monitored objects (comprised of lists and/or the user's event queue) in the range of vector entries are NON-EMPTY when monitoring for empty/non-empty state or NON-FULL when monitoring a list for full/non-full state (list notification vector), or if some entry in the range of vector entries is INVALID (local cache vector).

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the label.

**,VALIDATE=YES**

**,VALIDATE=NO**

Use VALIDATE=YES with TESTLOCALCACHE as follows:

- When you omit VECTORINDEX, to request that the system validate connectivity to the coupling facility
- When you specify VECTORINDEX, to request that the system validate connectivity to the coupling facility and check the validity of a particular data item

Use VALIDATE=NO with TESTLOCALCACHE and VECTORINDEX to request that the system check the validity of a particular data item without validating connectivity to the coupling facility. You

would choose this option only if a previous serialized request was done with `VALIDATE=YES` and the serialization has not yet been released.

**,VECTORINDEX=vectorindex**

With the `LTVECENTRIES` request, use this input parameter to specify the starting index of the range of vector entries that are to be loaded and tested. The index specified must be evenly divisible by 32. Thirty two consecutive vector entries will be loaded and tested. The vector index for a given vector size of `N` goes from zero to `N-1`.

With the `TESTLISTSTATE` request, use this input parameter to specify a fullword field that contains the vector index entry associated with the list or event queue of interest. For a vector with `N` entries, valid vector index values range from zero to `N-1`. The association between the particular index number and the list or event queue must already be established using the `IXLLSTC` macro.

With the `TESTLOCALCACHE` request, use this input parameter to specify a fullword field that contains the vector index entry to be tested for validity. For a vector with `N` entries, valid vector index values range from zero to `N-1`. The association between the particular index number and the data item must already be established using the `IXLCACHE` macro.

For more information about the `IXLLSTC` or `IXLCACHE` macros, see *z/OS MVS Programming: Sysplex Services Guide*.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the fullword field.

**,VECTORLEN=vectorlen**

Use this input parameter to specify a fullword field containing the new total number of entries in the list notification vector or local cache vector. The new number of entries can be greater than or less than the current number of entries but must be greater than zero and be a multiple of 32. If you specify a number that is not a multiple of 32, the system will round up the number to a multiple of 32.

The system will attempt to expand or contract your vector to the size you specify.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the fullword field that contains the requested vector length.

**VECTORTOKEN=vectortoken**

Use this input parameter to identify the list notification vector or local cache vector on which the specified request is to be performed.

The local vector was initially created by the `IXLCONN` service, if requested, and the vector token was returned in the connect answer area (`IXLYCONA`) in the field `CONAVECTORTOKEN`.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 12-character field that contains the vector token.

**,WORKAREA=workarea**

Use this input parameter to specify a 20-byte save area to be used by the `IXLVECTR` service.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 20-character field that begins on a word boundary.

## ABEND Codes

---

Abend X'026' (See *z/OS MVS System Codes* for more information on this abend).

## Return Codes

---

When the `IXLVECTR` macro returns control to your program, GPR 15 (and *retcode* if you coded `RETCODE`) contains a hexadecimal return code with the following exception. If you coded `BRANCHTABLE=YES`, the return code will be available only in GPR 15.

The `IXLYCON` macro provides equate symbols for the return and reason codes.

The following tables contain hexadecimal return codes, their associated equate symbols, and the meaning and suggested action for each return code issued by each type of `IXLVECTR REQUEST`.

Table 91. Return Codes for the IXLVECTR Macro with the MODIFYVECTORSIZE Parameter

Hexadecimal Return Code	Equate Symbol Meaning and Action
0	<p><b>Equate Symbol:</b> IXLRETCODEMODIFYDONE</p> <p><b>Meaning:</b> The list notification vector or local cache vector was modified as requested.</p> <p><b>Action:</b> None.</p>
4	<p><b>Equate Symbol:</b> IXLRETCODELESSTHAN</p> <p><b>Meaning:</b> System error. The list notification vector or local cache vector is smaller than the size you requested because insufficient storage was available. The new number of vector entries is returned in the field specified by ACTUALVECLEN.</p> <p><b>Action:</b> If you require a larger list notification vector, retry the request later when more storage might be available. If the problem persists, notify the system programmer so that the cause of the problem can be determined and corrected.</p>
8	<p><b>Equate Symbol:</b> IXLRETCODENOSTORAGE</p> <p><b>Meaning:</b> System error. Storage could not be obtained to increase the list notification vector or local cache vector size. The size remains unchanged.</p> <p><b>Action:</b> Retry the request later when more storage might be available. If the problem persists, notify the system programmer so that the cause of the problem can be determined and corrected.</p>
C	<p><b>Equate Symbol:</b> IXLRETCODEINVALIDTOKN</p> <p><b>Meaning:</b> Program error. The vector token you specified is not valid.</p> <p><b>Action:</b> Check your program for errors such as:</p> <ul style="list-style-type: none"> <li>• You specified the address of the vector token incorrectly.</li> <li>• The vector token has been overlaid or corrupted between the time you received it and the time you specified it on the IXLVECTR macro.</li> <li>• You disconnected from the cache structure (using the IXLDISC macro) before issuing the IXLVECTR macro. You must be a connected user to issue IXLVECTR.</li> <li>• The connector's task failed and the vector was cleaned up.</li> </ul> <p>Correct the error, and rerun the program.</p>
10	<p><b>Equate Symbol:</b> IXLRETCODEINVALIDLEN</p> <p><b>Meaning:</b> Program error. The vector length you specified is not valid.</p> <p><b>Action:</b> The vector length must be greater than, or equal to, 1. Correct the error, and rerun the program.</p>

Table 92. Return Codes for the IXLVECTR Macro with the LTVECENTRIES Parameter

Hexadecimal Return Code	Equate Symbol Meaning and Action
0	<p><b>Equate Symbol:</b> IXLRETCODEALLEMPVALFULL</p> <p><b>Meaning:</b> All monitored objects (comprising of lists and/or the user's event queue) in the range of vector entries are EMPTY when monitoring for empty/non-empty state or FULL when monitoring a list for full/non-full state (list notification vector), or all local cache entries in the range of vector entries are valid (local cache vector).</p> <p><b>Action:</b> None.</p>
4	<p><b>Equate Symbol:</b> IXLRETCODESOMENEINVF</p> <p><b>Meaning:</b> Some monitored object (comprising of lists and/or the user's event queue) in the range of vector entries is NON-EMPTY when monitoring for empty/non-empty state or NON-FULL when monitoring a list for full/non-full state (list notification vector), or some local cache entry in the range of vector entries is INVALID (local cache vector).</p> <p><b>Action:</b> None.</p>
8	<p><b>Equate Symbol:</b> IXLRETCODEINDXINVALID</p> <p><b>Meaning:</b> Program error. The vector index you specified is not valid.</p> <p><b>Action:</b> Check your program for errors such as:</p> <ul style="list-style-type: none"> <li>• You specified a vector index that is not a multiple of 32.</li> <li>• You specified a vector index value less than zero.</li> <li>• You specified a vector index value greater than, or equal to, the number of vector entries. Vector index values for a vector with N entries range from zero to N-1.</li> <li>• You specified the address of the vector index value incorrectly.</li> </ul> <p>Correct the error, and rerun the program.</p>
C	<p><b>Equate Symbol:</b> IXLRETCODEINVALIDTOKN</p> <p><b>Meaning:</b> Program error. The vector token you specified is not valid.</p> <p><b>Action:</b> Check your program for errors such as:</p> <ul style="list-style-type: none"> <li>• You specified the address of the vector token incorrectly.</li> <li>• The vector token has been overlaid or corrupted between the time you received it and the time you specified it on the IXLVECTR macro.</li> <li>• You disconnected from the list structure (using the IXLDISC macro) before issuing the IXLVECTR macro. You must be a connected user to issue IXLVECTR.</li> <li>• The connector's task failed and the vector was cleaned up.</li> </ul> <p>Correct the error, and rerun the program.</p>

Table 93. Return Codes for the IXLVECTR Macro with the TESTLISTSTATE Parameter

Hexadecimal Return Code	Equate Symbol Meaning and Action
0	<b>Equate Symbol:</b> IXLRETCODELSTEMPTY <b>Meaning:</b> The list or event queue is empty. <b>Action:</b> None.
0	<b>Equate Symbol:</b> IXLRETCODELSTFULL <b>Meaning:</b> The monitored list is in the full state. <b>Action:</b> None.
4	<b>Equate Symbol:</b> IXLRETCODELSTNONEMPTY <b>Meaning:</b> The list or event queue is not empty. <b>Action:</b> None.
4	<b>Equate Symbol:</b> IXLRETCODELSTNONFULL <b>Meaning:</b> The monitored list is in the non-full state. <b>Action:</b> None.
8	<b>Equate Symbol:</b> IXLRETCODEINDXINVALID <b>Meaning:</b> Program error. The vector index you specified is not valid. <b>Action:</b> Check your program for errors such as: <ul style="list-style-type: none"> <li>• You specified a vector index value greater than, or equal to, the number of vector entries. Vector index values for a vector with N entries range from zero to N-1.</li> <li>• You specified the address of the vector index value incorrectly.</li> <li>• Another unit of work modified the vector size while this TESTLISTSTATE was being processed.</li> </ul> Correct the error, and rerun the program.
C	<b>Equate Symbol:</b> IXLRETCODEINVALIDTOKN <b>Meaning:</b> Program error. The vector token you specified is not valid. <b>Action:</b> Check your program for errors such as: <ul style="list-style-type: none"> <li>• You specified the address of the vector token incorrectly.</li> <li>• The vector token has been overlaid or corrupted between the time you received it and the time you specified it on the IXLVECTR macro.</li> <li>• You disconnected from the list structure (using the IXLDISC macro) before issuing the IXLVECTR macro. You must be a connected user to issue IXLVECTR.</li> <li>• The connector's task failed and the vector was cleaned up.</li> </ul> Correct the error, and rerun the program.

Table 94. Return Codes for the IXLVECTR Macro with the TESTLOCALCACHE Parameter

Hexadecimal Return Code	Equate Symbol Meaning and Action
0	<p><b>If you specified the VECTORINDEX parameter:</b></p> <p><b>Equate Symbol:</b> IXLRETCODEBUFVALID  <b>Meaning:</b> The data item in the local cache buffer is valid.  <b>Action:</b> None.</p> <p><b>If you omitted the VECTORINDEX parameter:</b></p> <p><b>Equate Symbol:</b> IXLRETCODECONNECTED  <b>Meaning:</b> There has been no interruption of connectivity to the coupling facility. Cross invalidation is reflected for data items in the cache structure.  <b>Action:</b> None.</p>
4	<p><b>If you specified the VECTORINDEX parameter:</b></p> <p><b>Equate Symbol:</b> IXLRETCODEBUFNOTVALID  <b>Meaning:</b> The data item in the local cache buffer is not valid.  <b>Action:</b> Obtain new copy if the data item.</p> <p><b>If you omitted the VECTORINDEX parameter:</b></p> <p><b>Equate Symbol:</b> IXLRETCODENOTCONNECTED  <b>Meaning:</b> All vector index entries are invalid.  <b>Action:</b> Obtain new copies of data items.</p>
8	<p><b>Equate Symbol:</b> IXLRETCODEINDXINVALID  <b>Meaning:</b> Program error. TESTLOCALCACHE was specified with a VECTORINDEX value that is not valid.  <b>Action:</b> Check your program for errors such as:</p> <ul style="list-style-type: none"> <li>• You specified a vector index value greater than, or equal to, the number of vector entries.</li> <li>• You specified the address of the vector index value incorrectly.</li> <li>• Another unit of work modified the vector size while this TESTLOCALCACHE was being processed.</li> </ul> <p>Correct the error, and rerun the program.</p>
C	<p><b>Equate Symbol:</b> IXLRETCODEINVALIDTOKEN  <b>Meaning:</b> Program error. The vector token you specified is not valid.  <b>Action:</b> Check your program for errors such as:</p> <ul style="list-style-type: none"> <li>• You specified the address of the vector token incorrectly.</li> <li>• You disconnected from the cache structure (using the IXLDISC macro) before issuing the IXLVECTR macro. You must be a connected user to issue IXLVECTR.</li> <li>• The vector token has been overlaid or corrupted between the time you received it and the time you specified it on the IXLVECTR macro.</li> <li>• The connector's task failed and the vector was cleaned up.</li> </ul> <p>Correct the error, and rerun the program.</p>





## Chapter 86. IXLZSTR – Coupling Facility Structure Data Access Service

### Description

The IXLZSTR macro allows you to request coupling facility structure data from a dump containing that information. The macro returns the data requested in an answer area that you provide. You can use IXLZSTR only in an IPCS environment.

The TYPE parameter on the IXLZSTR macro determines what type of data will be returned to the caller:

- TYPE=STRUCTURE returns structure information.
- TYPE=CLASS returns class information for a requested cache structure.
- TYPE=LISTNUM returns list number information for a requested list structure.
- TYPE=LOCKENTRIES returns lock table information for a requested list structure.
- TYPE=USERCNTLS returns user control information for a requested structure.
- TYPE=ENTRY returns entry information for a requested entry in a requested structure.
- TYPE=EMCONTROLS returns event monitor control information for a requested list structure. The requested list structure must be a keyed list structure allocated in a coupling facility with CFLEVEL=4 or higher.
- TYPE=EVENTQS returns event queue control information for a requested connection ID. The requested connection ID must be connected to a keyed list structure in a coupling facility with CFLEVEL=4 or higher.

To use the macro, follow these guidelines:

- Ensure that you include the proper structure attributes on the applicable structure type. For example, you cannot request a range of list numbers for a cache structure or a castout class for a list structure. If you specify structure attributes that do not apply, IXLZSTR fails the request and returns a return code of 8 and a reason code of X'14'.

**Note:** To determine the structure type of a given structure in the dump, issue the IXLZSTR macro with TYPE=STRUCTURE,STRLEVEL=SUMMARY. IXLZSTR returns a list of all structures requested to be dumped and their structure type in the answer area. Using the StrBStrSummary mapping in IXLZSTRB, you can find the structure name and check its corresponding type.

To determine the correct dump identifier for a given structure, issue the same macro as above. IXLZSTR returns the structure dump ID for each structure in the list of structures requested to be dumped, as well as the name of the coupling facility in which the structure was allocated.

- On requests that specify a range of values, ensure that the starting range value is not greater than the ending range value. If the value is greater, IXLZSTR fails the request and returns a return code of 8 and a reason code of X'18'.
- Use the RESTOKEN keyword to retrieve information when IXLZSTR cannot return all information at once in the answer area. When this volume of information exists, you must issue the IXLZSTR macro more than once, each time specifying the RESTOKEN as input.
- **Determining an Answer Area Size**

The IXLZSTR macro puts the requested information in an answer area that you provide. The answer area contains a header (STRBHEADER) that describes the remainder of the information retrieved, followed by one or more entries for the requested information.

The amount of storage that you can specify for the answer area depends on the type of request. The minimum answer area size is 4096 bytes (4K). If you supply less than this amount, IXLZSTR fails the request and returns a return code of 8 and a reason code of X'10'.

A minimum storage allocation of 8192 bytes (8K) is required for a TYPE=STRUCTURE STRLEVEL=DETAIL request and for any request that specifies the ENTRYDATA keyword. If you provide less than 8K of storage for the answer area, IXLZSTR fails the request and returns a return code of 8 and a reason code of X'10'.

You might not be able to provide an answer area large enough to contain all the information returned by IXLZSTR for a single request. To allow for this situation, IXLZSTR provides the RESTOKEN parameter. You specify this token, initialized to binary zeros, on each invocation of IXLZSTR. If all the information can be returned in the answer area, IXLZSTR sets a return code of 0 and a reason code of 0. However, if all the information cannot be returned at once, IXLZSTR sets a return code of 4 and a reason code of 4 to indicate that there is more data to be retrieved from the dump data set. To retrieve the remainder of the data, you must invoke the IXLZSTR macro again with the same keywords and with RESTOKEN as input. (Do not reset RESTOKEN to binary zeros until all until all data for this request is retrieved.) When all data has been retrieved, IXLZSTR sets a return code of 0 and a reason code of 0 to indicate successful completion. IXLZSTR also sets the RESTOKEN to binary zeros.

#### • Determining the Size of Each Entry Returned

Each time you retrieve information into your answer area, the header record (STRBHEADER) specifies the length of the table entry (STRBTABLEENTRYLEN). Each table entry also contains a header (STRBENTRY), which contains additional length specifications for the data.

- STRBENTRYADJLEN — length of the adjunct data
- STRBENTRYEDATALEN — length of the entry data
- STRBENTRYCNTLEN — length of the entry control information

By calculating the sum of these four lengths, you can determine the length of the table entry and calculate the location of the next entry in the answer area.

When the information requested cannot all fit in one answer area, you must again calculate the length for the entry. Additional entries that can fit in the answer area after the first is completed, are preceded by the STRBENTRY header, which contains the appropriate lengths for the data entry that follows.

## Environment

The following are the environment requirements for the caller.

Environment	Environment requirement
Minimum authorization:	One of the following: <ol style="list-style-type: none"> <li>1. Problem state</li> <li>2. PKM allowing key 8</li> </ol>
Dispatchable unit mode:	Task
Amode:	31-bit
ASC mode:	Primary
Serialization:	Enabled
Control parameters:	Control parameters must be in the primary address space

## Programming Requirements

The IXLZSTRB macro provides the format of the area that ANSAREA points to. If you intend to use that area, include that macro in your program. IXLZSTRB maps each of the following types of data to be returned to the caller:

- STRBHEADER maps the header section of the answer area.
- STRBSTRSUMMARY maps summary information about the requested structure.
- STRBSTRDETAIL maps detail information about a requested structure.
- STRBSUMMARY maps summary information for a specified CLASS, LISTNUM, or event monitor controls request for a particular structure.
- STRBDETAIL maps detail information for a specified CLASS, LISTNUM, or event queue controls request for a particular structure.
- STRBENTRY maps entry information for a specified CLASS or LISTNUM request for a particular structure.
- STRBEMCDetail maps EMC information for event monitor controls associated with a specified LISTNUM and maps EMC information for event queue controls associated with a specified CONID.

Depending on the request type, you might need additional macros to map the information returned by IXLZSTR.

Table 95. Answer Area Macros for IXLZSTR		
Macro Name	Descriptive Name	Macro Use
IHAARB	Associated Request Block Mapping	Provides a map of the list of all the ranges of objects that were validly requested to be dumped.
IXLYDCAC	Dumping Coupling Facility Cache Structure Controls Mapping	Provides a map of the dumping cache structure controls.
IXLYDCCC	Dumping Cast-out Class Controls Mapping	Provides a map of the dumping castout class controls.
IXLYDDIB	Dumping Directory Information Block Mappings	Provides mappings for: <ul style="list-style-type: none"> <li>• Lock table entry (LTE), which contains the lock table entry information associated with a structure.</li> <li>• List-entry control block (LECB), which contains the entry controls associated with a list structure.</li> <li>• Directory information block (DIFB), which contains the element controls associated with a cache structure.</li> <li>• List-user control block (LUCB), which contains the list user controls.</li> <li>• Local-cache control block (LCCB), which contains the local cache controls.</li> <li>• Event monitor controls block (EMC), which contains the event monitor controls associated with a list structure.</li> </ul>
IXLYDEQC	Dumping Event Queue Controls Mapping	Provides a map of the dumping event queue controls.
IXLYDLC	Dumping List Controls Mapping	Provides a map of the dumping list header controls and the list monitor table entries found in the list controls.

Table 95. Answer Area Macros for IXLZSTR (continued)

Macro Name	Descriptive Name	Macro Use
IXLYDLCC	Dumping Local Cache Controls Mapping	Provides a map of the dumping local cache controls.
IXLYDLIC	Dumping Coupling Facility List Structure Controls Mapping	Provides a map of the dumping list structure controls.
IXLYDLUC	Dumping List User Controls Mapping	Provides a map of the dumping list user controls.
IXLYDSCC	Dumping Storage Class Controls Mapping	Provides a map of the dumping storage class controls.
IXLYSTRC	Coupling Facility Structure Data Access Service Dump Reason Code Constants	Provides the constants for interpreting the dump reason code.
IXLZSTRB	Coupling Facility Structure Data Access Service Answer Area Mappings	Maps the answer area data that was requested and provides constants to interpret any return and reason codes issued.

## Restrictions

You can invoke IXLZSTR only in an IPCS environment. Include the BLSABDPL mapping macro.

*z/OS MVS IPCS User's Guide* provides general information about an IPCS environment.

## Input Register Information

Before issuing the IXLZSTR macro, the caller does not have to place any information into any register unless using it in register notation for a particular parameter, or using it as a base register.

## Output Register Information

When control returns to the caller of the IXLZSTR macro, the general purpose registers (GPRs) contain:

### Register

#### Contents

**0**

Reason code if GPR15 return code is non-zero

**1**

Used as work register by the system

**2 - 13**

Unchanged

**14**

Used as work register by the system

**15**

Return code

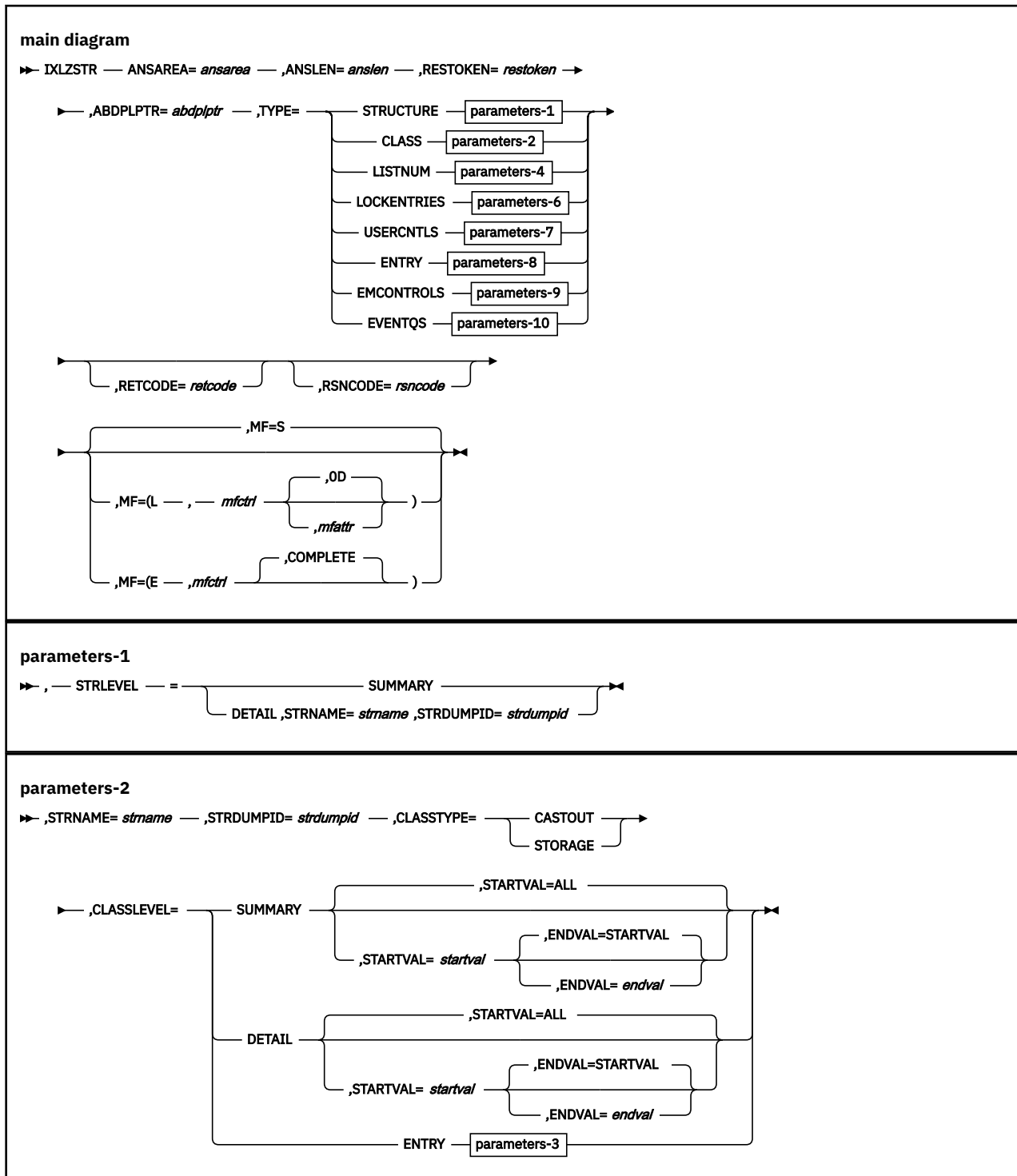
**For registers that the system changes,** a caller who depends on these registers containing the same value before and after issuing the macro must save these registers before issuing the macro and restore them after the system returns control.

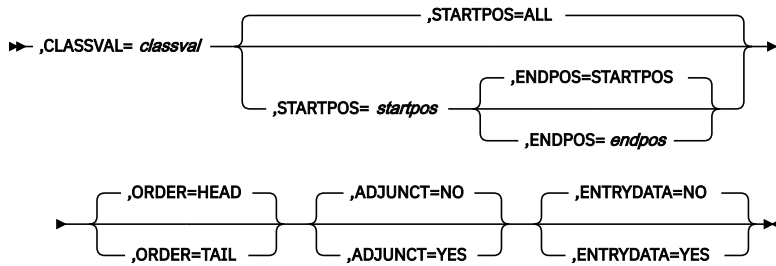
## Performance Implications

None.

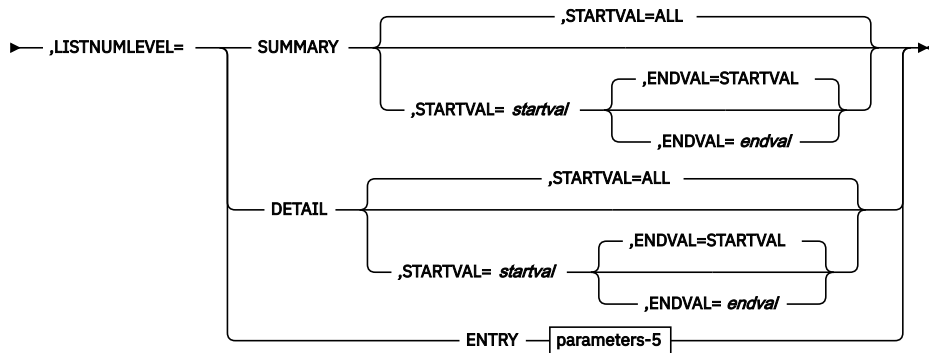
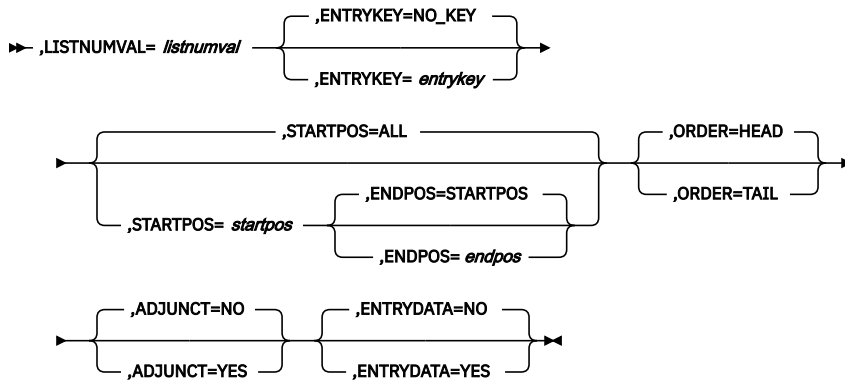
## Syntax Diagram

The syntax of the IXLZSTR macro is as follows:

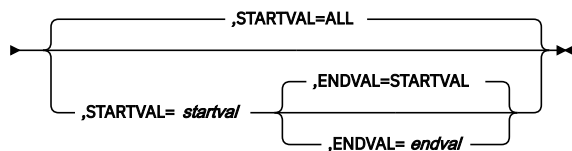


**parameters-3****parameters-4**

►► ,STRNAME= *strname* — ,STRDUMPID= *strdumpid* →

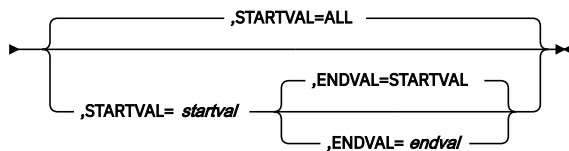
**parameters-5****parameters-6**

►► ,STRNAME= *strname* — ,STRDUMPID= *strdumpid* →



**parameters-7**

►► ,STRNAME= *strname* — ,STRDUMPID= *strdumpid* →

**parameters-8**

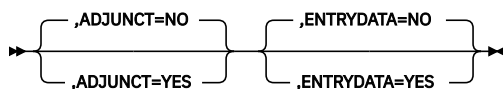
►► ,STRNAME= *strname* — ,STRDUMPID= *strdumpid* →

,

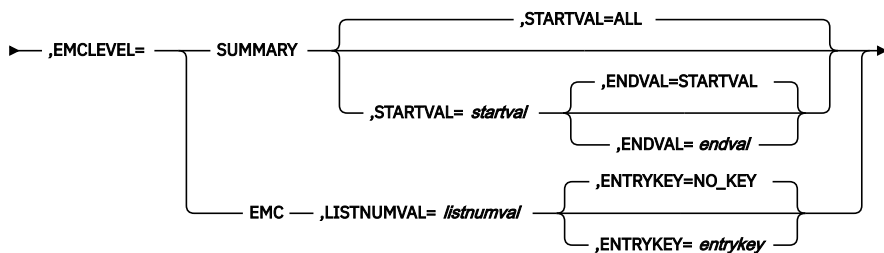
►► ENTRYNAME — = — *entryname* →

,

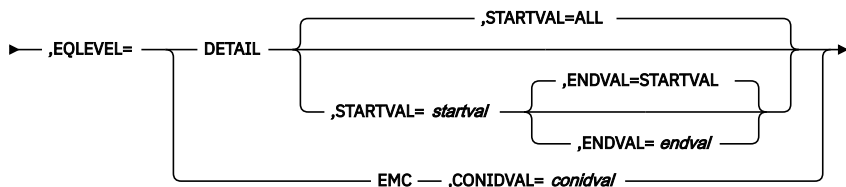
►► ENTRYID — = — *entryid* →

**parameters-9**

►► ,STRNAME= *strname* — ,STRDUMPID= *strdumpid* →

**parameters-10**

►► ,STRNAME= *strname* — ,STRDUMPID= *strdumpid* →



## Parameter Descriptions

The parameter descriptions are listed in alphabetical order. Default values are underlined>.

### **,ABDPLPTR=***abdplptr*

Use this input parameter to specify the address of the ABDUMP parameter list (ABDPL) that the system currently is using.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword field that contains the address of the ABDPL.

**,ADJUNCT=NO****,ADJUNCT=YES**

Use this input parameter to specify whether the adjunct data associated with the requested entries should be returned with the entries. The STRBENTRY mapping for each entry points to the adjunct data and also provides the length of the adjunct data.

**NO**

Do not return adjunct data.

**YES**

Do return adjunct data.

**ANSAREA=ansarea**

Use this output parameter to identify the answer area.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the answer area.

**,ANSLEN=anslen**

Use this input parameter to specify the length of the answer area.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword field that contains the length of the answer area.

**,CLASSLEVEL=SUMMARY****,CLASSLEVEL=DETAIL****,CLASSLEVEL=ENTRY**

Use this input parameter to specify the level of the CLASS type request.

**SUMMARY**

Information about a requested class, a range of classes, or all classes (castout or storage) in the dump. The STRBSUMMARY mapping of IXLZSTRB maps the entries in the answer area.

**DETAIL**

Detailed information about a requested class, a range of classes, or all classes (castout or storage) in the dump. The STRBDETAIL mapping of IXLZSTRB maps the entries in the answer area. The entry also contains a pointer to the class controls information. For storage class information, the DSCC mapping in IXLYDSCC maps the storage class controls. For castout class information, the DCCC mapping in IXLYDCCC maps the castout class controls.

**ENTRY**

An entry at a requested position, a group of entries at a requested range of positions, or the entries at all entry positions that were dumped in a requested class. The STRBENTRY mapping of IXLZSTRB maps the entries in the answer area. The entry also contains a pointer to the entry control information. The DDIC mapping in IXLYDDIB maps the entry controls.

**,CLASSTYPE=CASTOUT****,CLASSTYPE=STORAGE**

Use this input parameter to specify the type of class that should be retrieved.

**CASTOUT**

Castout classes

**STORAGE**

Storage classes

**,CLASSVAL=classval**

Use this input parameter to identify the class for which data entries will be retrieved.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the fullword field that contains the class for which data entries will be retrieved.

**,CONIDVAL=conidval**

Use this input parameter to specify the connection ID for which the system will retrieve event monitor control entries on the event queue.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a one-byte field that contains the connection ID for which event monitor control information is being requested.



**,EMCLEVEL=SUMMARY****,EMCLEVEL=EMC**

Use this input parameter to specify the level of the EMCONTROLS type request.

**SUMMARY**

The list of event monitor controls in the list structure that was requested to be dumped. The system returns information for a list number, a range of list numbers, or all list numbers in the dump. The STRBSUMMARY mapping of the IXLZSTRB macro maps the entries in the answer area.

**EMC**

Event monitor controls that were dumped for the requested list structure. The STRBEMCDetail mapping of the IXLZSTRB macro maps the entries in the answer area. The STRBEMCDetail contains a pointer to the event monitor controls in the answer area. The DEMC mapping of the IXLYDDIB macro maps the event monitor controls entry.

**,ENDPOS=STARTPOS****,ENDPOS=*endpos***

Use this input parameter to specify the end of the requested entry position range. If you do not specify this keyword, the ending range position will be equal to the starting range position specified on the STARTPOS keyword.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword field that contains the end of the requested entry position range. ENDPOS is an optional keyword. The default is STARTPOS.

**,ENDVAL=STARTVAL****,ENDVAL=*endval***

Use this input parameter to specify the end of the requested class range. If you do not specify this keyword, the ending range value will be equal to the starting range value specified on the STARTVAL keyword.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the fullword field that contains the value indicating the end of the requested class range.

**,ENTRYDATA=NO****,ENTRYDATA=YES**

Use this input parameter to specify whether the entry data associated with the requested entries should be returned with the entries. The STRBENTRY mapping for each entry points to the entry data and also provides the total length of the entry data. If the buffer fills up before all entry data is returned, the STRBENTRY mapping provides the length of the entry data that has been returned.

**NO**

Do not return entry data.

**YES**

Do return entry data.

**,ENTRYID=*entryid***

Use this input parameter to specify the list entry identifier of the list entry to be retrieved. This keyword should be used only if the requested structure is a list structure.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the 12-character field that contains the list entry identifier of the list entry to be retrieved.

**,ENTRYKEY=*entrykey***

Use this input parameter to identify the key to be used for retrieving list entries. If this keyword is specified, only list entries with the requested key will be returned.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the 16-character field that contains the key to be used for retrieving list entries.

**,ENTRYNAME=*entryname***

Use this input parameter to specify the name of the data entry or list entry to be retrieved.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the 16-character field that contains the name of the data entry or list entry to be retrieved.

**,EQLEVEL=DETAIL**

**,EQLEVEL=EMC**

Use this input parameter to specify the level of the EVENTQS type request.

#### **DETAIL**

Detailed information about a requested connection ID, a range of connection IDs, or all connection IDs in the dump. The STRBDETAIL mapping of the IXLZSTRB macro maps the entries in the answer area. In the STRBDETAIL entry, there is a pointer to the event queue controls in the answer area. The DEQC mapping of the IXLYDEQC macro maps the event queue controls. The STRBDETAIL entry also contains the length of the event queue controls.

#### **EMC**

Event monitor controls entries associated with the event queue of the requested connection ID. The STRBEMCDETAIL mapping of the IXLZSTRB macro maps the entries in the answer area. In the STRBEMCDETAIL entry, there is a pointer to the event monitor controls in the answer area. The DEMC mapping of the IXLYDDIB macro maps the event monitor controls.

**,LISTNUMLEVEL=SUMMARY**

**,LISTNUMLEVEL=DETAIL**

**,LISTNUMLEVEL=ENTRY**

Use this input parameter to specify the level of the LISTNUM request. You can request summary information about all list numbers, detail information about specific list numbers, or information about entry positions.

#### **SUMMARY**

Information about the list numbers in the requested structure that was dumped. You can specify a specific list number, a range of list numbers, or all list numbers in the dump. The STRBSUMMARY mapping of IXLZSTRB maps the entries in the answer area.

#### **DETAIL**

Detailed information about a specific list number, a range of list numbers, or all list numbers in the dump. The STRBDETAIL mapping of IXLZSTRB maps the entries in the answer area. The STRBDETAIL mapping includes a pointer to the list controls in the answer area and the length of the list controls. The DLC mapping of IXLYDLC maps the list controls.

#### **ENTRY**

Information for a particular list number about an entry at a specific position, a group of entries at a range of positions, or the entries at all positions in the dump. The STRBENTRY mapping of IXLZSTRB maps the entries in the answer area. The STRBENTRY also contains a pointer to the entry controls in the answer area. The DDIL mapping of IXLYDDIB maps the entry controls.

**,LISTNUMVAL=*listnumval***

Use this input parameter to specify the list number for which list entries are to be retrieved.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword field that contains the list number for which entries are to be retrieved.

**,MF=S**

**,MF=(L,*mfctrl*)**

**,MF=(L,*mfctrl*,*mfattr*)**

**,MF=(L,*mfctrl*,0D)**

**,MF=(M,*mfctrl*)**

**,MF=(M,*mfctrl*,COMPLETE)**

**,MF=(M,*mfctrl*,NOCHECK)**

**,MF=(E,*mfctrl*)**

**,MF=(E,*mfctrl*,COMPLETE)**

**,MF=(E,*mfctrl*,NOCHECK)**

Use MF=S to specify the standard form of the macro, which builds an inline parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the

execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=M to specify the modify form of the macro. Use the modify form to generate code to put the parameters into the parameter list.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form, and generates the macro invocation to transfer control to the service.

#### **,mfctrl**

Use this output parameter to specify a storage area to contain the parameters.

**To Code:** Specify the RS-type name or address (by using a register 2 - 12) of the parameter list.

#### **,mfattr**

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembly language Data Studio pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

#### **,COMPLETE**

#### **,NOCHECK**

Use this input parameter to specify the degree of macro parameter syntax checking the system is to do.

#### **COMPLETE**

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

**Note:** In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if SMILE=*var* were an optional parameter and the default is SMILE=NO\_SMILE then it would not be documented. However, if the default was SMILE=-), then it would be documented because a value would be the default.

#### **NOCHECK**

Use this parameter to specify that the system is not to check for required parameters nor to supply defaults for omitted optional parameters.

#### **,ORDER=HEAD**

#### **,ORDER=TAIL**

Use this input parameter to specify the order in which entries for the requested class are to be returned. For a storage class, the head of the queue represents the least recently referenced entry and the tail of the queue represents the most recently referenced entry. For a castout class, the head of the queue represents the most recently changed entry and the tail of the queue represents the least recently changed entry.

#### **HEAD**

Entries are in head-to-tail order. If you also specify the keywords STARTPOS/ENDPOS, the system returns the entries in head-to-tail order.

#### **TAIL**

Entries are in tail-to-head order. If you also specify the keywords STARTPOS/ENDPOS, the system returns the entries in tail-to-head order.

#### **,RETCODE=retcode**

Use this output parameter to specify the location in which the system is to copy the return code from GPR 15.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword field to contain the return code.

**,RESTOKEN=restoken**

Use this input parameter to specify a 64-byte token that must be passed on all IXLZSTR requests. The token allows a request to be continued across multiple calls if all the requested data could not be returned in the answer area. Before your first request to access the data, you must initialize the field to binary zeros. On every request, the IXLZSTR service initializes RESTOKEN with information required on subsequent IXLZSTR requests. Therefore, the user who is requesting the data must not modify the contents of RESTOKEN.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the 64-character field that must be passed on all IXLZSTR requests.

**,RSNCODE=rsncode**

Use this output parameter to specify the location in which the system is to copy the reason code from GPR 0.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword field to contain the reason code.

**,STARTPOS=ALL****,STARTPOS=startpos**

Use this input parameter to specify the start of the requested entry position range. If you do not specify this keyword, all of the entries in the dump for the requested class will be returned.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a fullword field that contains the start of the requested entry position range.

**,STARTVAL=ALL****,STARTVAL=startval**

Use this input parameter to specify the start of the requested class range. If you do not specify this keyword, all of the classes requested to be dumped will be returned.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of the fullword field that contains the value indicating the start of the requested class range.

**,STRDUMPID=strdumpid**

Use this input parameter to specify the structure dump ID of structure being requested.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a halfword field that contains the structure dump ID.

**,STRLEVEL=SUMMARY****,STRLEVEL=DETAIL**

Use this input parameter to specify the level of the STRUCTURE request.

**SUMMARY**

A list of all the structures and their types that are in the dump. STRBSTRSUMMARY maps the entries in the answer area.

**DETAIL**

Detailed information about a requested structure. STRBSTRDETAIL maps the entry in the answer area. The entry also contains a pointer to the structure control information. For a cache structure, the DCAC mapping in IXLYDCAC maps the structure controls. For a list structure, the DLIC mapping in IXLYDLIC maps the structure controls.

**,STRNAME=strname**

Use this input parameter to specify the name of the structure for which information is being requested.

**To Code:** Specify the RS-type name or address (using a register from 2 to 12) of a 16-character field that contains the name of the structure for which information is being requested.

**,TYPE=STRUCTURE**  
**,TYPE=CLASS**  
**,TYPE=LISTNUM**  
**,TYPE=LOCKINDEX**  
**,TYPE=USER**  
**,TYPE=ENTRY**  
**,TYPE=EMCONTROLS**  
**,TYPE=EVENTQS**

Use this input parameter to specify the type of information to be retrieved by the IXLZSTR macro from the dump data set.

#### **STRUCTURE**

Structure information.

#### **CLASS**

Castout class or storage class information. Use CLASS only for a cache structure.

#### **LISTNUM**

List number information. Use LISTNUM only for a list structure.

#### **LOCKENTRIES**

Detailed information about a requested lock entry, a range of lock entries, or all entries in the lock table for a requested list structure. IXLZSTR returns only the non-zero lock table entries. Use LOCKENTRIES only for a list structure. The DLTE mapping in IXLYDDIB maps the entries in the answer area.

#### **USERCNTLS**

Detailed information about a requested connection ID, a range of connection IDs, or all connection IDs for a requested structure. For a cache structure, the DLCCB mapping in IXLYDDIB maps the entries in the answer area. For a list structure, the DLUCB mapping in IXLYDDIB maps the entries in the answer area.

#### **ENTRY**

Detailed information about a requested entry name or entry ID in a requested structure. The STRBENTRY mapping in IXLZSTRB maps the entry in the answer area. The entry also contains a pointer to the entry controls. For a cache structure, the DDIC mapping in IXLYDDIB maps the entry controls. For a list structure, the DDIL mapping in IXLYDDIB maps the entry controls.

#### **EMCONTROLS**

Event monitor control information. Use EMCONTROLS only for a keyed list structure allocated in a coupling facility with CFLEVEL=4 or higher. The DEMC mapping in IXLYDDIB maps the event monitor controls.

#### **EVENTQS**

Event queue control information. Use EVENTQS only for a keyed list structure allocated in a coupling facility with CFLEVEL=4 or higher. IXLYDEQC maps the event queue controls.

## ABEND Codes

---

None.

## Return and Reason Codes

---

When the system returns control to the caller, GPR 15 (and *retcode*, if you coded RETCODE) contains the return code and GPR 0 (and *rsncode*, if you coded RSNCODE) contains the reason code.

Table 96. Return and Reason Codes for the IXLZSTR Macro

Hexadecimal Return Code	Hexadecimal Reason Code	Meaning and Action
00	None	<b>Meaning:</b> IXLZSTR completed successfully and returned the requested information in the data area provided. <b>Action:</b> None
04	04	<b>Meaning:</b> Not all requested data could be returned in the data area provided because the data area was not large enough. <b>Action:</b> To retrieve the remainder of the data, invoke IXLZSTR again with the same keywords and the RESTOKEN as input to the macro.
08	04	<b>Meaning:</b> The STRNAME specified on the IXLZSTR macro does not appear in the dump. <b>Action:</b> Correct STRNAME, if incorrect, and reissue the IXLZSTR macro.
08	08	<b>Meaning:</b> The STRNAME specified on the IXLZSTR macro appears in the dump, but the STRDUMPID specified does not appear in the dump <b>Action:</b> Correct STRDUMPID, if incorrect, and reissue the IXLZSTR macro.
08	0C	<b>Meaning:</b> No coupling facility data appears in the dump. <b>Action:</b> Ensure that you have specified the correct dump.
08	10	<b>Meaning:</b> ANSAREA specified on the IXLZSTR macro does not meet the minimum storage requirement for the request. <b>Action:</b> Recalculate the amount of storage required for the type of request you are submitting and reissue the IXLZSTR macro.
08	14	<b>Meaning:</b> The data does not appear in the dump because the attributes of the requested data does not match the attributes of the structure type. <b>Action:</b> Correct the attributes specified and reissue the IXLZSTR macro.
08	18	<b>Meaning:</b> The range specification on the IXLZSTR macro is not valid. The starting value is greater than the ending value. <b>Action:</b> Correct the range values and reissue the IXLZSTR macro.
0C	04	<b>Meaning:</b> Environmental error. Unable to obtain system storage. <b>Action:</b> Retry the request one or more times. If the problem persists, record the return and reason codes and supply them to the appropriate IBM support personnel.
10	04	<b>Meaning:</b> Failure in IXLZSTR processing. Some storage could not be obtained in the dump data set. <b>Action:</b> Save the return code information and contact the IBM support center.

## Example

```

        TITLE  'XETPUB02-Sample IPCS exit using IXLZSTR with RESTOKEN'
*01* FUNCTION =
*       Sample program to illustrate use of IXLZSTR macro
*       in an IPCS environment to access coupling facility
*       structure data from a dump containing that information.  Usage
*       of the RESTOKEN keyword is shown for handling requests for
*       which all of the information can not be returned in the
*       user-provided answer area.
*
*02* OPERATION =
*       (1) Obtain a 4K (4096 byte) buffer to be passed to the
*       IXLZSTR macro for use as the answer area for a summary
*       type request.
*
*       (2) Issue the IXLZSTR request to obtain summary structure
*       information from the input dump which is expected to
*       contain coupling facility structure data for a cache
*       structure.  The dump also contains multiple entries for
*       a given castout class value and the entries have adjunct
*       and entry data that was also dumped.
*
*       (3) The answer area is searched looking for summary structure
*       information for a specific CACHE structure name that is
*       expected to have been dumped.  If it is found then the
*       structure dump id is saved for subsequent use
*       in obtaining class entry information for a given castout
*       class value.
*
*       (4) Free 4K (4096 byte) buffer previously obtained for use as
*       a summary data answer area.
*
*       (5) Obtain a 8K (8192 byte) buffer to be passed to the
*       IXLZSTR macro for use as the answer area for a entry data
*       type request.
*
*       (6) Issue the IXLZSTR request to obtain all entries with both
*       adjunct and entry data returned for a given castout class
*       value for the specific Cache structure.
*
*       As parts of the answer area are "analyzed", WTOs will
*       be issued:
*       - Indicating start of analysis of the answer area
*       - Identifying the entry name from entry controls
*       - Displaying first 16 bytes of adjunct and entry data
*       for each entry
*       - Indicating going to next table entry in answer area
*       - Indicating finished with answer area
*
*       (7) Process all of the table entries returned in the answer
*       area.  Since TYPE(CLASS) data with adjunct and entry data
*       is requested and size of entry data is quite large,
*       multiple invocations of IXLZSTR are made using the
*       RESTOKEN returned to get more data that can not fit in
*       the returned answer area for the first and subsequent
*       IXLZSTR invocations. (I.e. Step 6 and 7 is repeated as
*       many times as necessary to get all of the data.)
*
*       (8) Free 8K (8192 byte) buffer previously obtained for use as
*       a entry data answer area.
*
*       (9) Free dynamic storage previously obtained and return to
*       caller.
*****
*02* RECOVERY-OPERATION = This program functions without recovery.
*
*****
EJECT

*****
*       Standard entry linkage
*
*****
        STM      R14,R12,12(R13)
        BALR     BASEREG1,0           Establish addressability
        USING    *,BASEREG1

```

```

MODID  BR=YES
LR     R2,R1          Save input parameter
LR     R3,R13         Save callers savearea address
STORAGE OBTAIN,ADDR=(DATAREG1),SP=0,LENGTH=DYNASIZE
LA     DATAREG2,4095(,DATAREG1) Set Second Data Register
USING  DYNA,DATAREG1   First Data Register
USING  DYNA+4095,DATAREG2 Second Data Register
ST     R3,SAVEAREA+4   Save @ of callers savearea
ST     DATAREG1,8(,R3) Chain our savearea to callers
EJECT

*****
*
*   Initialize variables
*
*****
MVC    EXITRC,=AL4(GOODRETC) * Initialize return code
ST     R2,ABDPLPTR      Save input pointer to ABDPL
MVC    WTOEXEC(LENWTOS),WTOS * Copy static parm list to dynamic
MVC    WTOTXTD1(L'WTOTXTD1),WTOTXTS1 * Prime WTO text length
EJECT

*****
* (1) Obtain a 4K (4096 byte) buffer to be passed to the
*      IXLZSTR macro for use as the answer area for a summary
*      type request.
*****
L      R0,SUMMSIZE      Size of summary buffer to obtain
STORAGE OBTAIN,ADDR=ANSAREA_PTR,SP=0,LENGTH=(R0)
USING  AREAMAP,R2
L      R2,ANSAREA_PTR   Get addressability to Answer area
EJECT

*****
* (2) Issue the IXLZSTR request to obtain summary structure
*      information from the input dump which is expected to contain
*      Coupling Facility structure data for a Cache structure. The
*      dump should also contain multiple entries for a given castout
*      class value and the entries should have adjunct and entry data
*      that was also dumped.
*****
XC     MYRESTOKEN(64),MYRESTOKEN Initialize RESTOKEN to
*      binary zeros
IXLZSTR ANSAREA=AREAMAP,      Output answer area
        ANSLEN=SUMMSIZE,     Answer area length
        RESTOKEN=MYRESTOKEN, Input/Output token for IXLZSTR
        ABDPLPTR=ABDPLPTR,   IPCS common parameter List address
        TYPE=STRUCTURE,      Get Structure information
        STRLEVEL=SUMMARY,    Get summary information
        RETCODE=SAVERET,     Return Code
        RSNCODE=SAVERSN,     Reason Code
        MF=(E,STREXEC)
*****
*      Check for the requested data being successfully accessed
*
*****
L      R3,SAVERET       Get return code
C      R3,=AL4(STRBRETCODESUCC) * All data accessed ?
BNE    BADACC           Data was not successfully accessed
L      R3,SAVERSN       Get reason code
C      R3,=AL4(STBRNSNCODESUCC) * All data accessed ?
BNE    BADACC           Data was not successfully accessed
EJECT

*****
*      Summary data was successfully accessed in the dump
*
*****
* (3) Search the answer area table entries (which contain summary
*      structure information for the input Cache structure that this
*      sample IPCS exit is interested in. If the structure is found
*      then the corresponding structure dump id is saved for subsequent
*      use in obtaining class entry information for the input castout
*      class value.
*
* Note: If the Cache structure of interest was in Structure Rebuild
*      there may be more than one dumped instant of the Cache
*      structure. For purposes of this sample exit, the installation
*      is assumed to be interested in the first dumped instance of the
*      structure.
*****
WTO 'XETPUB02 ANALYZING SUMMARY ANSWER AREA',ROUTCODE=11

```



```

        USING STRBHEADER,R2      Base Header section of ANSAREA
        USING STRBSTRSUMMARY,R4  Base Structure Summary table entry
        L      R4,STRBFIRSTTABLEENTRY@ * Get addressability to first
*
        NI      WRKFLG,NOTFOUND   Did not find Cache structure yet
        LA      R6,STRBSTRSUMMARY_LEN * Size of summary table entry
        LA      R8,1              Loop increment
        LR      R9,R8             Processing first table entry
NEXTENT    ST      R9,I           Remember loop count
        L      R5,STRBNUMTABLEENTRIES * Get number of table entries
        CR      R9,R5             All table entries searched ?
        BH      DONESRH           Yes, stop searching

```

```

*****
*      Check to see if this table entry is for the input Cache      *
*      structure                                                    *
*****
        CLI     STRBSTRSUMMARYTYPE,STRBSTRTYPECACHE * Cache str type ?
        BNE     TRYNEXT      No, Try next table entry
*****
*      Check to see if the structure name matches the input Cache  *
*      structure name                                              *
*****
        CLC     STRBSTRSUMMARYNAME,APPSTRNM * Same str name ?
        BNE     TRYNEXT      No, Try next table entry
*****
*      Found the table entry for the input Cache structure name.    *
*      (NOTE: More code needed to handle cases where structure is  *
*      dumped more than once -- e.g. Structure in rebuild)        *
*****
        OI      WRKFLG,FOUNDIT   Indicate Cache structure found
        LH      R3,STRBSTRSUMMARYSTRDUMPID * Save structure dump id
        STH     R3,DUMPID_STR    Remember structure dump id
        B       DONESRH         Stop searching for structure
TRYNEXT    EQU      *           Try next table entry (if any)
        ALR     R4,R6            Point to next table entry
        ST      R4,SUMMARY_PTR   Remember new table entry address
        ALR     R9,R8            Processing another table entry
        B       NEXTENT         Go to process the next table entry
        SPACE   1
*****
*      Finished searching returned structure summary information    *
*****
DONESRH    EQU      *           Finished search for structure
        TM      WRKFLG,FOUNDIT   Found the input Cache structure ?
        BZ      NOCACHE         No, tell user structure not found
        WTO     'XETPUB02 CACHE STRUCTURE FOUND IN DUMP',ROUTCDE=11
        B       FREESUM         Go free answer area buffer
NOCACHE    EQU      *           Did not find the Cache structure
        WTO     'XETPUB02 CACHE STRUCTURE NOT FOUND IN DUMP',ROUTCDE=11
        MVC     EXITRC,=AL4(BADRETC) Set bad return code
        B       FREESUM         Go free answer area buffer
        EJECT
*****
*      Summary data was not successfully accessed in the dump      *
*
*****
BADACC     EQU      *           Data was not accessed successfully
        MVC     WTOTXTD2(L'WTOTXTD2),WTOBADAC * Set message text
        MVC     MAPSERV(8),=CL8'IXLZSTR ' * Service that failed
* Convert hex return code to printable hex
        MVC     PHEXIN(4),SAVERET Hex return code to convert
        UNPK    PHEXOUT,PHEXIN   Unpack the data
        MVC     MAPRETC(8),PHEXOUT+1 Store unpacked data into target
        TR      MAPRETC(8),TRTBL-240 Translate to printable hex
* Convert hex reason code to printable hex
        MVC     PHEXIN(4),SAVERSN Hex reason code to convert
        UNPK    PHEXOUT,PHEXIN   Unpack the data
        MVC     MAPRSNC(8),PHEXOUT+1 Store unpacked data into target
        TR      MAPRSNC(8),TRTBL-240 Translate to printable hex
        BAL     R14,ISSUEWTO     Tell user access failed
        MVC     EXITRC,=AL4(BADRETC) Set bad return code
        EJECT

```

```

*****
* (4) Free 4K (4096 byte) buffer previously obtained for a summary *
*      data answer area.                                           *
*
*****
FREESUM    L      R0,SUMMSIZE    Size of summary buffer to be freed

```

```

        STORAGE RELEASE,ADDR=((R2)),SP=0,LENGTH=(R0)
        DROP    R2
*****
*   Only continue processing if input cache structure was dumped   *
*****
        TM      WRKFLG,FOUNDIT      Found the input Cache structure ?
        BZ      COMPLETE           No, exit is finished
        SPACE 2
*****
* (5) Obtain a 8K (8192 byte) buffer to be passed to the          *
*   IXLZSTR macro for use as the answer area for a Entry data      *
*   type request.                                                  *
*                                                                    *
*****
        L        R0,BIGSIZE           Size of big buffer to be obtained
        STORAGE OBTAIN,ADDR=ANSAREA_PTR,SP=0,LENGTH=(R0)
        USING AREAMAP,R2
        L        R2,ANSAREA_PTR      Get addressability to Answer area
        EJECT
*****
* (6) Issue the IXLZSTR request to obtain all entries with both     *
*   adjunct and entry data returned for a given input castout class *
*   value for the input Cache structure.                             *
*                                                                    *
*   This exit is interested in listing the first 16 bytes of the    *
*   entry data and adjunct data for each entry. For illustration    *
*   purposes a WTO will be issued to job log for each matching      *
*   entry found displaying this data.                                 *
*                                                                    *
*   The expected amount of data that will be returned in the answer *
*   area is quite large and will not fit in the coded 8K answer     *
*   area buffer. The following code illustrates how to use          *
*   IXLZSTR macro with RESTOKEN keyword in a loop to retrieve all   *
*   of the requested data.                                          *
*                                                                    *
*****
        LA      R5,ENTBUF           Point to start of entry data WTO
*                                                                    *
*   ST      R5,ENTBUF_PTR          Remember location in buffer
*   L        R5,MIN_ENTDATA_SIZE    Get size of buffer
*   ST      R5,LEN_NEEDED           Remember size of buffer still to
*                                                                    *
*   NI      WRKFLG,NOENTWTO         Remember that WTO has not been
*                                                                    *
*   XC      MYRESTOKEN(64),MYRESTOKEN Initialize RESTOKEN to
*                                                                    *
*   binary zeros
*
*****
*   Continue to request more of the entry data until all of the     *
*   data has been returned. (Start of loop)                         *
*                                                                    *
*****
GETDATA IXLZSTR ANSAREA=AREAMAP,      Output answer area      +
          ANSLEN=BIGSIZE,             Answer area length        +
          RESTOKEN=MYRESTOKEN,         Input/Output token for IXLZSTR +
          ABDPLPTR=ABDPLPTR,           IPCS common parameter List addr +
          STRNAME=APPSTRNM,            Structure name to get info on  +
          STRDUMPID=DUMPID_STR,        Structure dump id to get info on +
          TYPE=CLASS,                  Get Structure class information  +
          CLASSTYPE=CASTOUT,           Get castout class information  +
          CLASSLEVEL=ENTRY,            Get entry information          +
          CLASSVAL=GETCLASS,           Class value to get info on      +
          ADJUNCT=YES,                Request adjunct data            +
          ENTRYDATA=YES,              Request entry data              +
          ORDER=TAIL,                 Return data in tail-to-head order +
          RETCODE=SAVERET,             Return Code                    +
          RSNCODE=SAVERSN,             Reason Code                    +
          MF=(E,STREXEC)
        SPACE 1
*****
*   Check for the requested data being successfully accessed        *
*****
        L        R3,SAVERET          Get return code
        C        R3,=AL4(STRBRETCODESUCC) * All data accessed ?
        BNE     CHKSOME              No, Only part of data accessed ?
        L        R3,SAVERSN          Get reason code
        C        R3,=AL4(STRBRSNCODESUCC) * All data accessed ?
        BNE     BADACC2              Data was not successfully accessed
        B       GOODACC              Data was successfully accessed
        CHKSOME C R3,=AL4(STRBRETCODEMOREDATA) * More data to get ?
        BNE     BADACC2              Data was not successfully accessed

```

```

L      R3,SAVERSN          Get reason code
C      R3,=AL4(STRBRNCODEANSANOTLGE) * Answer area too small?
BNE    BADACC2            Data was not successfully accessed
EJECT

*****
* (7) Process all of the table entries returned in the answer area *
* from the IXLZSTR invocation. Since TYPE(CLASS) was requested *
* with adjunct and entry data also returned for a given castout *
* class value and the size of the entry data is quite large, the *
* contents of the answer area just returned may contain different *
* pieces of information. *
* *
* The first thing that appears in the answer area is always the *
* answer area header section mapped by the IXLZSTRB mapping macro *
* (section STRBHEADER). The header section will point to the *
* first table entry returned in the current answer area. *
* *
* The format of each table entry returned for a TYPE(CLASS) *
* CLASSLEVEL(ENTRY) request is also mapping by IXLZSTRB in the *
* mapping section beginning with field STRBENTRY. *
* *
*****

```

```

GOODACC EQU *              Data was successfully accessed
WTO 'XETPUB02 ANALYZING ANSWER AREA',ROUTCDE=11
USING STRBHEADER,R2        Base Header section of ANSAREA
USING STRBENTRY,R4         Base TYPE(CLASS) CLASSLEVEL(ENTRY)
*                           table entry
L      R4,STRBFIRSTTABLEENTRY@ * Get addressability to first
*                           table entry in answer area
LA     R8,1                Loop increment
LR     R9,R8               Processing first table entry
NEXTENT2 ST R9,I           Remember number of table entries
*                           processed
*****
* The STRBENTRY will indicate if the current table entry has any *
* entry controls present, where it is located and its size. The *
* presence, location, and size of any adjunct data is also in the *
* STRBENTRY. Also, the presence, location, and size of any entry *
* data will be indicated in the STRBENTRY information. *
* *
*****
SPACE 1
*****
* The first returned answer area associated with a given table *
* entry will point to any entry controls and adjunct data that is *
* going to be returned for the entry. *
*****
EJECT
*****
* Check for entry control data *
*****
L      R3,STRBENTRYCNTL@    Get pointer to entry control data
C      R3,ZERO              Entry control data present ?
BZ     SKIPCNTL             No, Skip entry control processing
*****
* Process the entry controls associated with the current table *
* entry. The entry controls for a cache structure are mapped by *
* DDIC mapping in mapping macro IXLYDDIB. *
* *
* The code below will access the DDIC to get the value specified *
* for the entry when the data was registered in the cache. This *
* information is surfaced by a WTO to the job log. *
*****
USING DDIC,R3              Base the cache entry controls
MVC    WTOTXTD2(L'WTOTXTD2),WTOCNT * Set message text
MVC    MAPCNTNM,DDICNAME    Copy the entry's name
BAL    R14,ISSUEWTO         Issue WTO
SKIPCNTL EQU *             Label skips entry control code
SPACE 1
*****
* Check for adjunct data *
*****
L      R3,STRBENTRYADJ@    Get pointer to adjunct data
C      R3,ZERO              Adjunct data present ?
BZ     SKIPADJ             No, Skip adjunct data processing

```

```

*****
* Process the adjunct data associated with the current table *
* entry. Adjunct data is application specific --- the following *

```

```

*      section of code just issues a WTO to show the first 16 bytes.  *
*****
      USING  ADJDATA,R3          Base adjunct data area
      MVC    WTOTXTD2(L'WTOTXTD2),WTOADJ * Set message text
      MVC    MAPADJ,ADJ16        First 16 bytes of adjunct data
      BAL     R14,ISSUEWTO       Issue WTO
SKIPADJ EQU    *                Label skips adjunct data code
      EJECT
*****
*      Process the entry data associated with the current table      *
*      entry. Entry data is application specific --- the following   *
*      section of code just issues a WTO to show the first 16 bytes  *
*      (see constant MIN_ENTDATA_SIZE).                               *
*                                                                      *
*      NOTE: This code assumes that entry data for the structure is  *
*      always at least 16 bytes. If an entry is found with less entry *
*      data than the expected application minimum size, then a WTO is *
*      issued (which would have been proceeded by a WTO that showed  *
*      the object name associated with the entry from entry controls). *
*                                                                      *
*      There may or may not be room for a table entry's entry data in *
*      the first answer area returned for the entry. Entry data may  *
*      also span one or more answer areas. The code below accumulates *
*      entry data in a WTO buffer until MIN_ENTDATA_SIZE bytes of    *
*      entry data have been returned. Even though there may be more  *
*      entry data than MIN_ENTDATA_SIZE only MIN_ENTDATA_SIZE bytes  *
*      are saved for purposes of the WTO showing the first part of the *
*      entry data.                                                    *
*                                                                      *
*****
*      NOTE: Number of IXLZSTR invocations needed to get all of the  *
*      entry data for a given table entry can be minimized by taking *
*      advantage of the STRBENTRYEDATALENLEFT2PROC field to          *
*      determine how big of an answer area to provide to IXLZSTR.    *
*****
      TM      WRKFLG,ENTWTO      Processed entry data for this
*                               table entry yet ?
      BNZ     SKIPENT           Yes, Skip Entry data processing
      L       R7,STRBENTRYEDATALEN Get total entry data length
*                               associated with the table entry
      C       R7,MIN_ENTDATA_SIZE Total entry data size is large
*                               enough ?
      BNL     OKTOTSZ           Yes, Process entry data
*****
*      Issue WTO indicating that the entry data for a table entry was *
*      too small. Previous WTO may have already been issued with    *
*      object name associated with this table entry.                 *
*****
      MVC     WTOTXTD2(L'WTOTXTD2),WTOBADSZ * Set message text
      BAL     R14,ISSUEWTO       Issue WTO
      MVC     EXITRC,=AL4(BADRETC) Set bad return code
      OI      WRKFLG,ENTWTO      Indicate WTO issued about this
*                               table entry's entry data
      B       SKIPENT           Skip Entry data processing
*                               table entry's entry data
      EJECT

OKTOTSZ EQU    *                Label total entry data size OK
      L       R3,STRBENTRYEDATA@ Get pointer to entry data
      C       R3,ZERO           Entry data present ?
      BZ      SKIPENT           No, Skip Entry data processing
*                               associated with the table entry
*****
*      Some entry data is present in this answer area for the current *
*      table entry being processed                                     *
*****
      L       R5,STRBENTRYEDATALEN Get amount of entry data returned
      L       R6,LEN_NEEDED      Get amount of entry data still
*                               needed before entry data WTO can
*                               be issued
      CLR     R5,R6             Was enough entry data returned to
*                               issue the entry data WTO ?
      BL      MOVEPART          No, move the part of entry data
*                               in this answer area to WTO buffer
      SPACE 1
*****
*      Enough entry data is present in this answer area to fill up the *
*      WTO buffer and issue the entry data WTO.                       *
*****
      L       R7,ENTBUF_PTR      Point to where to move data to

```

```

        BCTR    R6,0           Set length of data to move
        EX      R6,@MOVEENT    Move entry data to WTO buffer
        MVC     WTOTXTD2(L'WTOTXTD2),WTOENT * Set message text
        MVC     MAPENT,ENTBUF   Move entry data into WTO
        BAL     R14,ISSUEWTO    Issue WTO
        OI      WRKFLG,ENTWTO   Indicate WTO issued about this
*                                     table entry's entry data
        B       SKIPENT
MOVEPART EQU *               Label to move only part of needed
*                                     entry data to WTO buffer
        SLR     R6,R5          Calculate amount of entry data
*                                     still needed for this table entry
        ST      R6,LEN_NEEDED   Remember how much is still needed
        L       R7,ENTBUF_PTR   Point to where to move data to
        BCTR    R5,0           Get size of data to be moved
        EX      R5,@MOVEENT    Move entry data to WTO buffer
        ALR     R7,R5          Point to next byte to move entry
        ALR     R7,R8          Add one to adjust for prior BCTR
        ST      R7,ENTBUF_PTR   Remember next byte in WTO buffer
*                                     to move entry data to
        EJECT
SKIPENT EQU *               Skip entry data processing for
*                                     current table entry
*****
*   Determine if all of the entry data for current table entry has
*   been seen.
*****
        L       R5,STRBENTRYEDATALENLEFT2PROC * Get amount of entry
*                                     data left to process for last
*                                     table entry in this answer area
        C       R5,ZERO
        BNE     CHKLAST        More entry data for this entry ?
*                                     Yes, check that it is last one

*****
*   All of the entry data for this table entry has been seen.
*   Reset entry data buffer indicators for next table entry.
*****
        LA      R5,ENTBUF      Point to start of entry data WTO
*                                     buffer
        ST      R5,ENTBUF_PTR   Remember location in buffer
        L       R5,MIN_ENTDATA_SIZE Get size of buffer
        ST      R5,LEN_NEEDED   Remember size of buffer still to
*                                     be copied
        NI      WRKFLG,NOENTWTO Remember that WTO has not been
*                                     issued for current table entry
CHKLAST EQU *               Check for last table entry
*****
*   Determine if there is another table entry to process in the
*   current answer area
*****
        L       R9,I           Get number of entries processed
        ALR     R9,R8          Increment number of table entries
        L       R5,STRBNUMTABLEENTRIES * Get number of table entries
        CR      R9,R5          All table entries processed ?
        BH      DONEENT        Yes, stop processing table entries
        WTO     'XETPUB02 GOING TO NEXT TABLE ENTRY',ROUTCDE=11
        EJECT
*****
*   Point to the next table entry. The following calculation
*   will always give you the length to add to get to the next
*   table entry:
*
*   size table entry +size adjunct  +size entry data +size entry cntls
*   (i.e.)
*   STRBTABLEENTRYLEN+STRBENTRYADJLEN+STRBENTRYEDATLEN+STRBENTRYCNTLLEN
*
*   NOTE: Some of the sizes above can be different for each
*   table entry and different for the "same table entry"
*   for a table entry for which not all of the entry data
*   can fit in the provided answer area in one IXLZSTR
*   macro invocation.
*****
        L       R5,STRBTABLEENTRYLEN * Get size of table entry
        L       R6,STRBENTRYADJLEN  Get size of returned adjunct data
        ALR     R5,R6              Add to table entry size
        L       R6,STRBENTRYEDATALEN Get size of returned entry data
        ALR     R5,R6              Add to table entry size
        L       R6,STRBENTRYCNTLLEN Get size of returned entry cntls
        ALR     R5,R6              Add to table entry size
        ALR     R4,R5              Point to next table entry

```

```

      ST      R4,ENTRY_PTR      Remember new table entry address
      B      NEXTENT2          Go process the next table entry
      EJECT

*****
*      Finished processing returned TYPE(CLASS) CLASSLEVEL(ENTRY) data *
*      in the current answer area buffer.                               *
*****
DONEENT EQU      *      Finished processing table entries
      WTO 'XETPUB02 FINISHED WITH THIS ANSAREA',ROUTCDE=11

*****
*      Check to see if there is more data still to get from dump      *
*****
      L      R3,SAVERET        Is there more data to get ?
      C      R3,=AL4(STRBRETCODEMOREDATA) * More data to get ?
      BE     GETDATA           Go get more of the data
*****
*      (End of loop to request more of the entry data in the dump    *
*      until of the data has been returned)                          *
*                                                                      *
*****
      WTO 'XETPUB02 FINISHED ACCESSING ALL DATA',ROUTCDE=11
      B      FREEENT           Go free answer area buffer
      EJECT

*****
*      Entry data was not successfully accessed in the dump          *
*                                                                      *
*****
BADACC2 EQU      *      Data was not accessed successfully
      MVC     WTOTXTD2(L'WTOTXTD2),WTOBADAC * Set message text
      MVC     MAPSERV(8),=CL8'IXLZSTR ' * Service that failed
* Convert hex return code to printable hex
      MVC     PHEXIN(4),SAVERET   Hex return code to convert
      UNPK    PHEXOUT,PHEXIN      Unpack the data
      MVC     MAPRETC(8),PHEXOUT+1 Store unpacked data into target
      TR      MAPRETC(8),TRTBL-240 Translate to printable hex
* Convert hex reason code to printable hex
      MVC     PHEXIN(4),SAVERSN   Hex reason code to convert
      UNPK    PHEXOUT,PHEXIN      Unpack the data
      MVC     MAPRSNC(8),PHEXOUT+1 Store unpacked data into target
      TR      MAPRSNC(8),TRTBL-240 Translate to printable hex
      BAL     R14,ISSUEWTO        Tell user access failed
      MVC     EXITRC,=AL4(BADRETC) Set bad return code
      EJECT

*****
* (8) Free 8K (8192 byte) buffer previously obtained for a entry    *
* data answer area.                                                 *
*                                                                      *
*****
FREEENT L      R0,BIGSIZE        Size of entry buffer to be freed
      STORAGE RELEASE,ADDR=((R2)),SP=0,LENGTH=(R0)
      DROP    R2
      EJECT

*****
* (9) Free up dynamic storage and return to caller                  *
*                                                                      *
*****

*****
COMPLETE EQU      *
      L      R2,SAVEAREA+4      Save caller's save area address
      L      R3,EXITRC          Save IPCS exit return code
      STORAGE RELEASE,ADDR=((DATAREG1)),LENGTH=DYNASIZE
      LR     R13,R2             Restore caller's save area address
      L      R14,12(R13)        Restore Return address
      LR     R15,R3             Set IPCS exit return code
      LM     R0,R12,20(R13)      Restore Registers R0-R12
      BR     R14                Return to caller
      SPACE  2

*****
*      Special EX target instructions                                *
*****
@MOVEENT MVC     0(0,R7),0(R3)   Move entry data to WTO buffer
      EJECT

*****
*      Subroutine: ISSUEWTO                                          *
*      Function : This routine is called whenever contention is     *

```

```

*          detected on a dataset, and the dataset owner is a      *
*          job on this system. It will take whatever action it    *
*          can to attempt to relieve the contention.              *
*          *
*          Input      : WTOTXTD1 contains text for WTO message to be issued *
*          *
*          *
*****
ISSUEWTO EQU *
          STM      R14,R12,SAVE1          Save callers regs
          LA       R5,WTOTXTD1          Address WTO parmlist
          WTO TEXT=(R5),ROUTCDE=(11),MF=(E,WTOEXEC) * Issue WTO
          LM       R14,R12,SAVE1          Restore callers regs
          BR       R14                  Return to caller
          EJECT

*****
*          Register declares                                     *
*          *
*****
R0      EQU      0
R1      EQU      1
R2      EQU      2
R3      EQU      3
R4      EQU      4
R5      EQU      5
R6      EQU      6
R7      EQU      7
R8      EQU      8
R9      EQU      9
R10     EQU      10
*
DATAREG2 EQU      11
BASEREG1 EQU      12
R12     EQU      12
DATAREG1 EQU      13
R13     EQU      13
R14     EQU      14
R15     EQU      15
          EJECT
*
*          Static data                                         *
*          *
*****
          DS      0F
TRTBL   DC      CL16'0123456789ABCDEF' * Translate table
ZERO    DC      F'0' * Constant zero for comparisons
          SPACE   1

*****
*          Static WTO data                                     *
*          *
*****
WTOS     WTO TEXT=,ROUTCDE=(11),MF=L * Static form of WTO
LENWTOS  EQU *-WTOS * Length of WTO parmlist
WTOTXTS1 DC      AL2(L'WTOTXTD2) * WTO text length
WTOBADAC DC CL65'XETPUB02 mmmmmmmmm RETCODE=rrrrrrrr RSNCODE=ssssssss'
MAPSERV  EQU      WTOTXTD2+9,8,C'C' * Map service WTO insert
MAPRETC  EQU      WTOTXTD2+26,8,C'C' * Map service RETCODE insert
MAPRSNC  EQU      WTOTXTD2+43,8,C'C' * Map service RETCODE insert
WTOADJ   DC CL65'XETPUB02 FIRST 16 CHARS ADJUNCT IS: aaaaaaaaaaaaaaaaaa'
MAPADJ   EQU      WTOTXTD2+36,16,C'C' * Map adjunct data insert
WTOENT   DC CL65'XETPUB02 FIRST 16 CHARS ENTRY DATA: eeeeeeeeeeeeeeeee'
MAPENT   EQU      WTOTXTD2+36,16,C'C' * Map entry data insert
WTOBADSZ DC CL65'XETPUB02 ENTRY DATA SIZE WAS TOO SMALL
WTOCNT   DC CL65'XETPUB02 ENTRY NAME FROM DDICNAME: ccccccccccccccccc'
MAPCNTNM EQU      WTOTXTD2+36,16,C'C' * Map entry's object name insert
          EJECT
*****
*          Constants used in accessing Coupling Facility structure data. *
*          *
*****
APPSTRNM DC      CL16'CACHE02 * Application structure name to
*          be accessed in the dump
GETCLASS DC      F'2' * Castout class value for which
*          class entry data is to be

```

**1640** z/OS: z/OS MVS Sysplex Services Reference

```

*****
*                                                                 *
*   Dynamic data                                                                 *
*                                                                 *
*****
DYNAREA DSECT
SAVEAREA DS      18F      Standard savearea (first field)
ABDPLPTR DS      AL4      Pointer to ABDPL
ANSAREA_PTR DS    AL4      Answer area storage pointer
LEN_NEEDED DS     F        Length of entry data still
*                          needed before WTO is issued
CLASSVAL DS      1F        Class value
DETAIL_PTR DS     A        ANSAREA detail area data pointer
DUMPID_STR DS     1H        Structure dump id
DUMP_STRNAME DS   CL16      Structure name in dump
ENTBUF      DS    CL16      Entry data buffer for WTO
          DS      0F
ENTBUF_PTR DS     A        Pointer into entry data buffer
*                          to copy entry data into
ENTRY_PTR   DS     A        ANSAREA Entry area pointer
I           DS      1F      Loop index
PHEXIN      DS      CL5     Work area for printable hex conv
PHEXOUT      DS     CL10     Work area for printable hex conv
MYRESTOKEN DS   CL64        RESTOKEN returned by IXLZSTR
SAVERET      DS      F      Save macro service return code
SAVERSN      DS      F      Save macro service reason code
SAVE1        DS     15F     First level subroutine savearea
SAVE2        DS     15F     Second level subroutine savearea
SUMMARY_PTR DS     A        ANSAREA summary data pointer
EXITRC       DS      F      Module Return code
BADRET      EQU     8        Bad return code from IPCS exit
GOODRET      EQU     0        Good return code from IPCS exit
          SPACE 1
WRKFLG      DS     BL'008    Work Flags
FOUNDIT      EQU     X'80'    Indicates Input Cache structure
*                          summary info found in the dump
ENTWTO       EQU     X'40'    Indicates WTO issued about entry
*                          data for current entry
NOENTWTO     EQU     X'BF'    Indicates WTO not issued about
*                          entry data for current entry
NOTFOUND     EQU     X'7F'    Indicates summary info not found
*                          in the dump
          EJECT
*****
*                                                                 *
*   List forms of macros (Dynamic storage)                                     *
*                                                                 *
*****
          IXLZSTR MF=(L,STREXEC)
          EJECT

```

[illegible]



```

DYNASIZE EQU    *-DYNA                Total size of dynamic storage
EJECT
*****
*                                     *
*   Other mappings                     *
*                                     *
*****
AREAMAP EQU     0,,C'C'                Map Answer area
SPACE 2
ADJDATA DSECT                      Map Adjunct data
ADJ16   DS CL16                     First 16 bytes adjunct data
ADJREST DS CL48                     Rest of adjunct data
EJECT
*****
*                                     *
*   Mapping macros                     *
*                                     *
*****
BLSABDPL
IXLZSTRB
IXLYDDIB
END

```



---

## Appendix A. Accessibility

Accessible publications for this product are offered through [IBM Documentation for z/OS \(www.ibm.com/docs/en/zos\)](http://www.ibm.com/docs/en/zos).

If you experience difficulty with the accessibility of any z/OS documentation see [How to Send Feedback to IBM](#) to leave documentation feedback.



## Notices

---

This information was developed for products and services that are offered in the USA or elsewhere.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

*IBM Director of Licensing  
IBM Corporation  
North Castle Drive, MD-NC119  
Armonk, NY 10504-1785  
United States of America*

For license inquiries regarding double-byte character set (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

*Intellectual Property Licensing  
Legal and Intellectual Property Law  
IBM Japan Ltd.  
19-21, Nihonbashi-Hakozakicho, Chuo-ku  
Tokyo 103-8510, Japan*

**The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:** INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

This information could include missing, incorrect, or broken hyperlinks. Hyperlinks are maintained in only the HTML plug-in output for IBM Documentation. Use of hyperlinks in other output formats of this information is at your own risk.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

*IBM Corporation  
Site Counsel  
2455 South Road*

Poughkeepsie, NY 12601-5400  
USA

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

#### COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

## Terms and conditions for product documentation

---

Permissions for the use of these publications are granted subject to the following terms and conditions.

### **Applicability**

These terms and conditions are in addition to any terms of use for the IBM website.

### **Personal use**

You may reproduce these publications for your personal, noncommercial use provided that all proprietary notices are preserved. You may not distribute, display or make derivative work of these publications, or any portion thereof, without the express consent of IBM.

### **Commercial use**

You may reproduce, distribute and display these publications solely within your enterprise provided that all proprietary notices are preserved. You may not make derivative works of these publications, or

reproduce, distribute or display these publications or any portion thereof outside your enterprise, without the express consent of IBM.

## **Rights**

Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the publications or any information, data, software or other intellectual property contained therein.

IBM reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the publications is detrimental to its interest or, as determined by IBM, the above instructions are not being properly followed.

You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations.

IBM MAKES NO GUARANTEE ABOUT THE CONTENT OF THESE PUBLICATIONS. THE PUBLICATIONS ARE PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, AND FITNESS FOR A PARTICULAR PURPOSE.

## **IBM Online Privacy Statement**

---

IBM Software products, including software as a service solutions, ("Software Offerings") may use cookies or other technologies to collect product usage information, to help improve the end user experience, to tailor interactions with the end user, or for other purposes. In many cases no personally identifiable information is collected by the Software Offerings. Some of our Software Offerings can help enable you to collect personally identifiable information. If this Software Offering uses cookies to collect personally identifiable information, specific information about this offering's use of cookies is set forth below.

Depending upon the configurations deployed, this Software Offering may use session cookies that collect each user's name, email address, phone number, or other personally identifiable information for purposes of enhanced user usability and single sign-on configuration. These cookies can be disabled, but disabling them will also eliminate the functionality they enable.

If the configurations deployed for this Software Offering provide you as customer the ability to collect personally identifiable information from end users via cookies and other technologies, you should seek your own legal advice about any laws applicable to such data collection, including any requirements for notice and consent.

For more information about the use of various technologies, including cookies, for these purposes, see IBM's Privacy Policy at [ibm.com/privacy](http://ibm.com/privacy) and IBM's Online Privacy Statement at [ibm.com/privacy/details](http://ibm.com/privacy/details) in the section entitled "Cookies, Web Beacons and Other Technologies," and the "IBM Software Products and Software-as-a-Service Privacy Statement" at [ibm.com/software/info/product-privacy](http://ibm.com/software/info/product-privacy).

## **Policy for unsupported hardware**

---

Various z/OS elements, such as DFSMSdfp, JES2, and MVS, contain code that supports specific hardware servers or devices. In some cases, this device-related element support remains in the product even after the hardware devices pass their announced End of Service date. z/OS may continue to service element code; however, it will not provide service related to unsupported hardware devices. Software problems related to these devices will not be accepted for service, and current service activity will cease if a problem is determined to be associated with out-of-support devices. In such cases, fixes will not be issued.

## **Minimum supported hardware**

---

The minimum supported hardware for z/OS releases identified in z/OS announcements can subsequently change when service for particular servers or devices is withdrawn. Likewise, the levels of other software products supported on a particular release of z/OS are subject to the service support lifecycle of those

products. Therefore, z/OS and its product publications (for example, panels, samples, messages, and product documentation) can include references to hardware and software that is no longer supported.

- For information about software support lifecycle, see: [IBM Lifecycle Support for z/OS \(www.ibm.com/software/support/systemsz/lifecycle\)](http://www.ibm.com/software/support/systemsz/lifecycle)
- For information about currently-supported IBM hardware, contact your IBM representative.

## Programming Interface Information

---

This book documents intended Programming Interfaces that allow the customer to write programs to obtain services of z/OS.

## Trademarks

---

IBM, the IBM logo, and [ibm.com](http://www.ibm.com) are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at [Copyright and Trademark information \(www.ibm.com/legal/copytrade.shtml\)](http://www.ibm.com/legal/copytrade.shtml).

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Java™ and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

The registered trademark Linux® is used pursuant to a sublicense from the Linux Foundation, the exclusive licensee of Linus Torvalds, owner of the mark on a worldwide basis.

UNIX is a registered trademark of The Open Group in the United States and other countries.



---

# Index

## A

accessibility  
    contact IBM [1643](#)  
assistive technologies [1643](#)  
asynchronously duplexed structure  
    synchronizing [429](#)

## C

cache  
    delete NAME data items [543](#)  
    invalidate data item copies [507](#)  
    read data [649](#)  
    services [457](#)  
    vector, check or modify [1605](#)  
cached data items, process [587](#)  
capacity planning data [109](#)  
cast-out  
    data [463](#)  
    locks [757](#), [777](#)  
    retrieve statistical information [629](#)  
command  
    syntax diagrams [1](#)  
contact  
    z/OS [1643](#)  
coupling facility  
    connect to structure [847](#)

## D

data item  
    delete by NAME [543](#)  
    invalidate copies of [507](#)  
    process cached [587](#)  
delete NAME data items [543](#)  
delete single list entry [967](#)  
directory entry  
    names [605](#)  
    user data [605](#)  
directory information, retrieve [675](#)

## E

events, synchronize processing for [1593](#)

## F

failed-persistent connection, delete [951](#)

## I

IHABLDP macro [7](#)  
invalidate data item copies [507](#)  
IXCARM macro [15](#)  
IXCCFCM macro [51](#)

IXCCREAT macro [59](#)  
IXCDELET macro [71](#)  
IXCJOIN macro [79](#)  
IXCLEAVE macro  
    XCF member  
        place in not-defined state [101](#)  
IXCMG macro [109](#)  
IXCMOD macro [125](#)  
IXCMMSGC macro [133](#)  
IXCMMSGI macro [155](#)  
IXCMMSGIX macro [157](#)  
IXCMMSGO macro [181](#)  
IXCMMSGOX macro [183](#)  
IXCNOTE macro [229](#)  
IXCQUERY macro [279](#)  
IXCQUIES macro [303](#)  
IXCRECVY macro [313](#)  
IXCREQ macro [327](#)  
IXCSEND macro [371](#)  
IXCSETUS macro [373](#)  
IXCSRVR macro [385](#)  
IXCSYSCL macro [405](#)  
IXCTERM macro [421](#)  
IXLADUPX macro [429](#)  
IXLALTER macro [437](#)  
IXLAXISN macro [449](#)  
IXLCACHE macro  
    CASTOUT\_DATA request [463](#)  
    CROSS\_INVAL request [507](#)  
    DELETE\_NAME request [543](#)  
    DELETE\_NAMELIST request [561](#)  
    general information [457](#)  
    mapping macros  
        IXLYCAA [459](#)  
        IXLYCANB [459](#)  
        IXLYCCIH [459](#)  
        IXLYCRRB [459](#)  
        IXLYCSCS [459](#)  
        IXLYCUNB [459](#)  
        IXLYDEIB [459](#)  
        IXLYNSB [459](#)  
    PROCESS\_REFLIST request [587](#)  
    READ\_COCLASS request [605](#)  
    READ\_COSTATS request [629](#)  
    READ\_DATA request [649](#)  
    READ\_DIRINFO request [675](#)  
    READ\_STGSTATS request [697](#)  
    REG\_NAMELIST request [709](#)  
    RESET\_REFBIT request [727](#)  
    SET\_RECLVCTR request [743](#)  
    UNLOCK\_CASTOUT request [757](#)  
    UNLOCK\_CO\_NAME request [777](#)  
    WRITE\_DATA request [791](#)  
IXLCONN macro  
    cache structure connection [851](#)  
    CACHE structure connection [871](#)  
IXLCONN macro

## IXLCONN macro (*continued*)

### IXLCONN macro (*continued*)

general information [847](#)

LIST structure connection [874](#)

LOCK structure connection [879](#)

## IXLCSP macro [901](#)

## IXLDISC macro [923](#)

## IXLEERSP macro

connection event

disconnected [931](#)

failed [931](#)

rebuilding events

Rebuild Cleanup [931](#)

Rebuild connection failure [931](#)

Rebuild Quiesce [931](#)

Rebuild Stop [931](#)

## IXLFCOMP macro [943](#)

## IXLFORCE macro [951](#)

## IXLLIST macro

DELETE request [967](#)

DELETE\_ENTRYLIST request [995](#)

DELETE\_MULT request [1017](#)

DEQ\_EVENTQ request [1035](#)

general information [961](#)

LOCK request [1053](#)

mapping macros

IXLYEMC [963](#)

IXLYLAA [963](#)

IXLYLCTL [963](#)

IXLYLMI [963](#)

IXLYMSRI [963](#)

MONITOR\_EVENTQ [1067](#)

MONITOR\_LIST request [1081](#)

MONITOR\_SUBLIST request [1095](#)

MONITOR\_SUBLISTS request [1109](#)

MOVE request [1129](#)

READ request [1169](#)

READ\_EMCONTROLS request [1197](#)

READ\_EQCONTROLS request [1209](#)

READ\_MULT request [1265](#)

WRITE request [1291](#)

WRITE\_LCONTROLS request [1323](#)

## IXLLOCK macro

general information [1339](#)

## IXLLSTC macro [1371](#)

## IXLMG macro [1521](#)

## IXLPURGE macro [1533](#)

## IXLREBLD macro [1539](#)

## IXLRT macro [1565](#)

## IXLSYNCH macro [1583](#)

## IXLUSYNC macro [1593](#)

## IXLVETR macro [1605](#)

## IXLZSTR macro [1617](#)

## K

### keyboard

navigation [1643](#)

PF keys [1643](#)

shortcut keys [1643](#)

## L

### list

delete multiple entries [1017](#)

modify controls [1323](#)

monitor [1081](#)

move a list entry [1129](#)

services [961](#)

list notification vector, check or modify [1605](#)

LOCALREGCNTL [462](#)

### lock

services [1339](#)

structure recovery [1565](#)

locking functions [1053](#)

## M

### macro version

specifying with PLISTVER [5](#)

### mapping macros

IHAASCB [250](#)

IHAASXB [231](#)

IKJTCB [250](#)

IXCYAMDA [109](#), [110](#), [113](#), [117](#), [118](#)

IXCYARAA [15](#)

IXCYCON [260](#)

IXCYMEPL [158](#)

IXCYMQAA [133](#)

IXCYNOTE [261](#)

IXCYQUAA [59](#), [77](#), [80](#), [108](#), [131](#), [227](#), [279](#), [281](#), [285](#), [295](#), [311](#), [384](#), [428](#)

IXLYAMDA [1521](#), [1522](#), [1525](#)

IXLYCAA [459](#), [466](#), [489](#), [509](#), [525](#), [545](#), [564](#), [589](#), [607](#), [631](#), [652](#), [677](#), [698](#), [711](#), [729](#), [744](#), [759](#), [779](#), [795](#), [796](#), [824](#)

IXLYCANB [459](#), [605](#), [608](#), [609](#)

IXLYCCIH [459](#), [632](#), [633](#)

IXLYCON [944](#), [1340](#), [1606](#)

IXLYCONA [850](#), [857](#), [858](#)

IXLYCRRB [459](#)

IXLYCSCS [459](#), [701](#)

IXLYCUNB [459](#), [757](#), [760](#), [761](#), [777](#)

IXLYDEIB [459](#), [680](#)

IXLYEMC [963](#)

IXLYLAA [963](#), [971](#), [998](#), [1019](#), [1036](#), [1055](#), [1069](#), [1083](#), [1096](#), [1111](#), [1172](#), [1198](#), [1210](#), [1222](#), [1325](#)

IXLYLCTL [963](#)

IXLYLMI [963](#), [1223](#), [1224](#)

IXLYMRTD [1566](#)

IXLYMSRI [963](#)

IXLYNDE [281](#)

IXLYNEPL [1583](#)

IXLYNSB [459](#)

IXLYRTAA [1566](#)

measurement data, coupling facility [1521](#)

modify list controls [1323](#)

monitor list [1081](#)

multiple entries

entry list [995](#)

## N

### navigation

navigation (*continued*)  
keyboard [1643](#)

## O

obtain status, IXLLIST or IXLCACHE request [943](#)

## P

persistent structure, delete [951](#)  
planning data  
    capacity requirements [109](#)  
    tuning activities [109](#)  
PLISTVER keyword  
    to specify macro version 5  
processing, synchronize for events [1593](#)  
purge operations to coupling facility [1533](#)

## R

read  
    cast-out class statistics [629](#)  
    directory information [675](#)  
    names, directory entry [605](#)  
    storage statistics [697](#)  
    user data, directory entry [605](#)  
rebuild a structure [1539](#)  
reclaim vector [743](#)  
release cast-out locks [757](#), [777](#)  
request information [279](#), [313](#), [327](#), [385](#)  
reset reference bit [727](#)  
respond to event [931](#)  
retrieve directory information [675](#)

## S

serialization, delete [951](#)  
shortcut keys [1643](#)  
signalling services [155](#), [157](#)  
single list entry [1169](#)  
statistical information  
    cast-out class [629](#)  
    storage class [697](#)  
status-checking interval [125](#)  
structure dump, delete [951](#)  
summary of changes [xxxi](#)  
synchronize processing for events [1593](#)  
synchronous update to a lock structure [1583](#)  
syntax diagrams  
    how to read [1](#)

## T

trademarks [1648](#)  
tuning planning data [109](#)

## U

update user state field [373](#)  
user interface  
    ISPF [1643](#)  
    TSO/E [1643](#)

## V

vector  
    list notification [1605](#)  
    local cache [1605](#)  
    reclaiming [743](#)

## W

wait for completion, IXLLIST or IXLCACHE request [943](#)  
write data [791](#)

## X

XCF member  
    automatic restart manager services [15](#), [405](#)  
    change state to not-defined [71](#)  
    define  
        group name [59](#)  
        member name [59](#)  
        user state field [59](#)  
    place in active state [79](#)  
    place in quiesced state [303](#)







Product Number: 5655-ZOS

SA38-0658-70

