

z/OS Communications Server
3.2

SNA Customization



Note:

Before using this information and the product it supports, be sure to read the general information under [“Notices” on page 287](#).

This edition applies to 3.1 of z/OS® (5655-ZOS), and to subsequent releases and modifications until otherwise indicated in new editions.

Last updated: 2025-09-20

© **Copyright International Business Machines Corporation 2000, 2025.**

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

Contents

Figures.....	ix
Tables.....	xi
About this document.....	xvii
Who should read this document.....	xvii
How this document is organized.....	xvii
How to use this document.....	xviii
How to provide feedback to IBM.....	xviii
Conventions and terminology that are used in this information.....	xviii
Prerequisite and related information.....	xx
Summary of changes for SNA Customization.....	xxv
Summary of changes for z/OS 3.2.....	xxv
Changes made in z/OS Communications Server 3.1.....	xxv
Chapter 1. Writing VTAM installation-wide exit routines.....	1
Session management exit routine.....	1
Initial register contents.....	3
Final register contents.....	3
Design requirements.....	4
Parameter lists.....	5
Parameter list structure.....	6
Parameter descriptions.....	34
Virtual route selection exit routine.....	70
Initial register contents.....	71
Final register contents.....	71
Design requirements.....	71
Virtual route selection exit routine parameter list.....	72
Virtual route pacing window size calculation exit routine.....	77
Initial register contents.....	78
Final register contents.....	78
Design requirements.....	78
VR pacing window size calculation exit routine parameter list.....	79
Using the VR window size calculation exit routine for IMS.....	80
Session accounting exit routine.....	81
Initial register contents.....	81
Final register contents.....	81
Design requirements.....	81
Session authorization exit routine.....	82
Initial register contents.....	83
Final register contents.....	83
Design requirements.....	83
Session authorization exit routine parameter list.....	84
Configuration services XID exit routine.....	85
Initial register contents.....	86
Final register contents.....	87
Design requirements.....	87
Configuration services XID exit routine parameter list.....	88

Configuration services XID exit routine output parameter list.....	89
Vector header.....	89
Vector data formats.....	89
Selection of definitions for dependent LUs exit routine.....	97
Naming dynamically defined dependent LUs.....	98
Specifying a model LU.....	98
Failing dynamic definition requests.....	99
Initial register contents.....	99
Final register contents.....	99
Design requirements.....	100
SDDL exit routine parameter list.....	100
SDDL exit routine vectors.....	101
Command verification exit routine.....	105
Initial register contents.....	106
Final register contents.....	106
Design requirements.....	106
Command verification exit parameter list.....	107
USERVAR exit routine.....	109
Initial register contents.....	110
Final register contents.....	111
Design requirements.....	111
USERVAR exit routine parameter list.....	112
USERVAR parameters.....	113
Directory services management exit routine.....	115
Initial register contents.....	115
Final register contents.....	116
Design requirements.....	116
Diagnostic information.....	117
Parameter lists.....	117
Parameter list structure.....	118
Parameter descriptions.....	126
CDS list structure.....	141
Generic resource resolution exit routine.....	142
Initial register contents.....	142
Final register contents.....	143
Design requirements.....	143
Generic resource resolution exit routine parameter list.....	144
Generic resource resolution processing.....	147
Performance monitor exit routine.....	148
Multiple exit support.....	148
Initial register contents.....	149
Final register contents.....	149
Design requirements.....	149
Performance monitor exit routine parameter list.....	150
Performance data parameter list.....	151
Installing VTAM exit routines.....	173
Operator commands for VTAM exit routines.....	174
Activating and deactivating VTAM exit routines.....	175
Replacing VTAM exit routines.....	175
Passing user-defined data to installation-wide exit routines.....	175
Chapter 2. Writing logon manager installation-wide exit routines.....	177
Logon manager exit routines.....	177
Criteria for contending CLUs.....	177
TPF sessions.....	178
Session requests from dependent SLUs.....	178
Session requests from independent and dependent PLUs.....	179

Sessions initiated by TPF.....	179
USERVAR exit routine for TPF sessions.....	180
Activation processing.....	180
Processing a USERVAR.....	180
Adding a USERVAR.....	180
Updating a USERVAR.....	180
Deleting a USERVAR.....	180
Translating a USERVAR.....	181
CLU search exit routine.....	181
Initial register contents.....	182
Final register contents.....	182
Design requirements.....	183
CLU search exit parameter descriptions.....	183
Function code parameter lists.....	184
REQTAIL macroinstruction.....	189
Session partners list.....	190
Chapter 3. Writing TSO/VTAM installation-wide exit routines.....	193
TSO/VTAM exit routines.....	193
IKTCASX1: Error handling for unsupported terminals.....	194
IKTCASX2: User message language-hardware verification.....	195
IKTGETXT: Editing on unsupported terminals.....	197
IKTIDSX1: Output editing for IBM 3270 terminals.....	197
IKTIDSX2: Input editing for IBM 3270 terminals.....	198
IKTIDSX3: Attention handler for IBM 3270 terminals.....	198
IKTIDSX4: TGET edit for IBM 3270 terminals.....	198
IKTINX1: Logon edit.....	199
IKTINX2: I/O manager initialization.....	200
IKTRTX1: Output edit for IBM 3767, 3770, and 2741 terminals.....	200
IKTRTX2: Input edit for IBM 3767 and 3770 terminals.....	201
IKTRTX3: Attention handler for IBM 3767 and 3770 terminals.....	201
IKTRTX4: Edit for IBM 3767, 3770, and 2741 terminals.....	201
IKTWTX1: Output edit for WTTY and TWX terminals.....	202
Installing TSO/VTAM exit routines.....	202
Control blocks.....	203
IKTIPARM.....	203
IKTMPL.....	204
IKTOPARM.....	204
IKTWESTD.....	205
IKTXSA.....	208
Chapter 4. Defining user modules and tables.....	211
Directory size of symbol resolution table for the host network.....	211
Directory size of symbol resolution tables for other networks.....	212
CNM routing table.....	213
Installing the CNM routing table.....	214
Structure of the CNM routing table.....	214
Logon-interpret routine requirements.....	216
Initial register contents.....	216
Final register contents.....	217
Logon-interpret routine parameter list.....	217
Operation.....	218
Appendix A. IBM-supplied CNM routing table.....	219
Appendix B. VTAM session flows.....	221
Session flows for subarea.....	221

Cross-network session for CDINIT.....	221
Cross-network for INIT_OTHER_CD (third-party initiated).....	222
Session flows for alias selection.....	223
Virtual route selection for boundary function LUs.....	235
Session flows for APPN.....	235
Mixed subarea/APPN ILU-initiated session.....	235
Mixed subarea/APPN PLU-initiated network broadcast.....	237
Mixed subarea/APPN SLU-initiated directed search.....	239
Appendix C. Sample session management exit.....	241
Initialization.....	242
Function selection.....	243
Begin function.....	244
Obtaining user storage.....	245
Loading NETID registration table.....	245
Returning to VTAM.....	248
Secondary authorization function.....	248
Gateway path selection function.....	251
Examining LU names.....	252
Modifying gateway path selection list.....	254
Alias selection function.....	256
Translating DLU name.....	256
Translating OLU name.....	257
Final accounting function.....	260
End function.....	261
VTAM exit services.....	262
Design considerations.....	262
Example invocation of VTAM exit services.....	262
Problem determination.....	263
Appendix D. Sample configuration services XID exit routine.....	267
Generating a dynamic switched major node name.....	267
Device identification.....	267
Device definition files.....	268
Name generation function.....	271
Invocation.....	271
Name generation table.....	271
Tailoring the name generation table.....	273
Connection status records.....	274
Appendix E. Command verification exit routine.....	275
Sample command verification exit routine.....	275
Appendix F. Sample USERVAR exit routine for TPF sessions.....	277
Function code X'04' processing.....	277
Return codes.....	280
Appendix G. Sample CLU search exit routine for TPF sessions.....	281
Function code X'04' processing.....	281
Sample CLU search exit routine for TPF sessions.....	282
Architectural specifications.....	283
Accessibility.....	285
Notices.....	287
Terms and conditions for product documentation.....	288
IBM Online Privacy Statement.....	289

Policy for unsupported hardware.....	289
Minimum supported hardware.....	289
Programming interface information.....	290
Policy for unsupported hardware.....	290
Trademarks.....	290
Bibliography.....	291
Index.....	295

Figures

1. Network address before VTAM Version 4 Release 2.....	5
2. Network address effective in VTAM Version 4 Release 2 and later releases.....	5
3. Format of the IKTCASX1 message.....	195
4. Sample cross-network session for CDINIT.....	222
5. Sample cross-network session for INIT_OTHER_CD (third-party initiated).....	223
6. SLU-initiated session for alias selection.....	224
7. PLU-initiated session for alias selection.....	226
8. HPR across a VR-based TG.....	227
9. HPR across a VR-based TG.....	227
10. ILU-initiated session flows (third-party initiated).....	228
11. ILU-initiated session flows with PLU real name and NETID returned (third-party initiated).....	229
12. SLU-initiated flows with USERVAR name.....	230
13. Session flows for DSRLST.....	231
14. PLU-initiated flows for ONLY-SHR.....	232
15. PLU-initiated flows for SHR-SHR.....	234
16. Virtual route selection for boundary function LUs.....	235
17. Mixed subarea/APPN ILU-initiated session, part 1.....	236
18. Mixed subarea/APPN ILU-initiated session, part 2.....	237
19. Mixed subarea/APPN PLU-initiated network broadcast, part 1.....	238
20. Mixed subarea/APPN PLU-initiated network broadcast, part 2.....	239
21. Mixed subarea/APPN SLU-initiated directed search.....	240
22. Network environment for sample session management exit routine.....	242
23. User data storage area format.....	245

24. Network identifier registration table.....	246
25. Sample portion of PLU resource identification control vector.....	249
26. Gateway path selection list for sample session management exit routine.....	254
27. Real and alias network-qualified names.....	256
28. Alias terminal name pool—3-digit suffix.....	259
29. NIDDEF and CPNAME example network.....	270

Tables

1. Example of session management exit routine functions.....	2
2. Summary of parameter lists for session management exit routine - function codes X'FE', X'00' - X'03'	6
3. Summary of parameter lists for session management exit routine - function codes X'04' - X'07'.....	8
4. Summary of parameter lists for session management exit routine - function codes X'08' - X'0B'.....	10
5. Summary of parameter lists for session management exit routine - function codes X'0C' and X'FF'.....	12
6. Begin function parameter list.....	13
7. Initial authorization function parameter list.....	15
8. Secondary authorization function parameter List.....	17
9. Accounting function parameter list.....	19
10. Gateway path selection function parameter list.....	20
11. XRF session switch function parameter list.....	21
12. Adjacent SSCP selection function parameter list.....	22
13. Alias selection function parameter list.....	23
14. ALS selection function parameter list.....	26
15. Exit replacement function parameter list.....	28
16. Exit replaced function parameter list.....	29
17. Virtual route selection function parameter list.....	31
18. HPR virtual route selection function parameter list.....	32
19. End function parameter list.....	34
20. Environment vectors.....	35
21. Function code and related session information.....	37
22. Session management exit options.....	43

23. PLU, SLU, ILU, and USERVAR resource identifier control vectors.....	46
24. PLU and SLU hierarchy control vector.....	49
25. SLU table name vector.....	51
26. PLU and SLU fully qualified associated resource name control vector.....	51
27. PLU and SLU adjacent link control vector.....	52
28. Format of OLU gateway information vector.....	53
29. Format of gateway path selection list.....	53
30. Format of SSCP name list.....	55
31. Format of DLU gateway information vector.....	55
32. Format of OLU adjacent SSCP vector.....	56
33. Format of DLU adjacent SSCP vector.....	56
34. Alias selection input parameter list.....	57
35. Constants for the alias selection input parameter list.....	60
36. Alias selection output parameter list.....	61
37. Format of session initiation user data field.....	63
38. ALS list information vector.....	63
39. ALS name information vector.....	65
40. VR/TP list information vector.....	65
41. Session authorization data vector.....	66
42. VTAM Exit Services parameter list.....	68
43. VTAM Exit Services supported functions bitmap.....	68
44. VTAM Exit Services message parameter list.....	69
45. Session management data area parameter list.....	69
46. Virtual route selection exit routine parameter list.....	72
47. Virtual route descriptor block format.....	74

48. VR pacing window size calculation exit routine parameter list.....	79
49. Explicit route characteristics table.....	79
50. Session authorization exit routine parameter list.....	84
51. Configuration services XID exit routine parameter list.....	88
52. Configuration services exit routine output parameter list.....	89
53. Vector header format.....	89
54. Begin vector format.....	90
55. XID vector format.....	92
56. Build vector format.....	93
57. Configuration services XID exit resource entry block format.....	94
58. Connection status format.....	96
59. Failure vector format.....	97
60. End vector format.....	97
61. SDDLU exit routine parameter list.....	100
62. Dynamic definition request vector format.....	102
63. SDDLU resource entry block format.....	103
64. Command verification exit parameter list for X'04'.....	107
65. Command verification exit parameter list for X'10', X'18', and X'20'.....	108
66. USERVAR exit parameter list for X'04'.....	112
67. USERVAR exit parameter list for X'10', X'18', and X'20'.....	112
68. USERVAR parameters for USERVAR addition, update, or deletion.....	113
69. USERVAR parameters for USERVAR translation.....	113
70. Session partners list.....	114
71. Directory services management exit parameter summary for function codes X'FE', X'00', X'01', X'02', X'03'.....	118
72. Directory services management exit parameter summary for function codes X'04', X'05', X'08', X'09', X'FF'.....	119

73. Exit return codes, exit sense codes, and VTAM search steps.....	121
74. Environment vectors.....	127
75. Function code and related search information.....	128
76. Exit options.....	133
77. Border node and interchange node options.....	134
78. Interchange node list structure.....	135
79. OLU information structure.....	136
80. DLU information structure.....	136
81. Network-qualified adjacent CP name vector.....	137
82. Search correlator structure.....	138
83. PCID modifier structure.....	138
84. Search task list.....	138
85. Subnetwork routing list structure.....	139
86. Central directory server list.....	141
87. Generic resource resolution exit parameter list for entry code X'04'.....	144
88. Generic resource resolution exit parameter list for entry codes X'10', X'18', and X'20'.....	144
89. Input parameters for generic resource resolution.....	145
90. Generic resource member list.....	145
91. Output parameters for generic resource resolution.....	146
92. Performance monitor exit parameter list for entry code X'04'.....	150
93. Performance monitor exit parameter list for entry codes X'10', X'18', and X'20'.....	150
94. Performance list header format.....	151
95. Performance data vector.....	153
96. Performance data vector identifiers.....	154
97. Global environment vector.....	154

98. Global installation-wide exit vector.....	159
99. Global buffer pool vector.....	159
100. Global storage private storage vector.....	161
101. Global storage CSA vector.....	161
102. Global storage GETBLK vector.....	162
103. Global session vector.....	162
104. APPN directory services vector.....	164
105. Global APPN directory services border node vector.....	165
106. Global APPN topology data vector.....	166
107. CSM buffer pool vector.....	167
108. CSM storage usage vector.....	167
109. Basic route data vector.....	168
110. RTP data vector.....	170
111. MNPS application recovery data.....	172
112. MNPS Application Data.....	173
113. MNPS structure data vector.....	173
114. Libraries for VTAM exit routines.....	173
115. VTAM exit names for exit routines.....	174
116. Modifiable VTAM exit routines.....	175
117. CLU search exit parameter list header.....	184
118. Function codes X'01' and X'07' parameters.....	185
119. Function codes X'02' and X'06' parameters.....	185
120. Function codes X'03' and X'05' parameters.....	186
121. Function code X'04' parameters.....	186
122. REQTAIL macro parameter list for function code X'04'.....	189

123. Session partners list.....	191
124. Summary of the TSO/VTAM exit routines for VTIOC processing.....	193
125. Summary of the TSO/VTAM exit routines for TCAS processing.....	194
126. Summary of the TSO/VTAM exit routines for VTIOC and TCAS processing.....	194
127. IKTCASX2 input and output parameter list.....	196
128. Routine-module cross-reference.....	203
129. IKTIPARM control block.....	203
130. IKTMPL control block.....	204
131. IKTOPARM control block.....	204
132. IKTWESTD control block.....	205
133. IKTXSA control block.....	208
134. Format of CNM routing table header.....	214
135. Format of CNM routing table.....	215
136. Logon-interpret routine parameter list.....	217
137. NIDDEF and CPNAME definition file example.....	270

About this document

This document is designed to help users customize VTAM®. Customization is the process of tailoring VTAM by enhancing or extending it to suit the needs of your installation.

VTAM provides default constants, tables, modules, and exit routines. VTAM also provides functional sample exit routines. These defaults and samples are usually sufficient to meet the needs of most installations. However, because each installation is different, the VTAM defaults and samples might not be able to handle every situation. If the defaults and samples are not appropriate for your installation, you can modify or replace the defaults, or write your own exits. *z/OS Communications Server: SNA Customization* is a reference document for users who need to perform these tasks to customize VTAM.

You can also use this document as background reading to help you understand the function of installation-wide exit routines, the Communication Network Management (CNM) routing table, and logon-interpret routine requirements.

Who should read this document

This document is for system programmers, system analysts, or anyone who customizes VTAM. Before using this document, you should be familiar with the information in the following publications:

- [z/OS Communications Server: SNA Network Implementation Guide](#)
- [z/OS Communications Server: SNA Resource Definition Reference](#)

How this document is organized

This document contains the following topics:

- Chapter 1, “[Writing VTAM installation-wide exit routines,](#)” on page 1 describes how to code and install the following VTAM exit routines:
 - Session management exit routine
 - Virtual route selection exit routine
 - VR pacing window size calculation exit routine
 - Session accounting exit routine
 - Session authorization exit routine
 - Configuration services XID exit routine
 - Selection of definitions for dependent LUs exit routine
 - Command verification exit routine
 - USERVAR exit routine
 - Directory services management exit routine
 - Generic resources resolution exit routine
 - Performance monitor exit routine
- Chapter 2, “[Writing logon manager installation-wide exit routines,](#)” on page 177 describes how to code and install the following exit routines:
 - USERVAR exit routine for TPF sessions
 - Control logical unit search exit routine
- Chapter 3, “[Writing TSO/VTAM installation-wide exit routines,](#)” on page 193 describes how to code and install exit routines for TSO/VTAM.
- Chapter 4, “[Defining user modules and tables,](#)” on page 211 describes how to create or modify the communication network management (CNM) routing table and the Logon-Interpret Routine.

- [Appendix A, “IBM-supplied CNM routing table,” on page 219](#) lists the default CNM routing table for IBM® supplied CNM application programs.
- [Appendix B, “VTAM session flows,” on page 221](#) illustrates session flow information required for writing VTAM installation-wide exit routines. Included are flows for both subarea and APPN VTAM.
- [Appendix C, “Sample session management exit,” on page 241](#) illustrates how the initial session management exit routine environment is established, the type of information available in each parameter list for the specific sample environment, and some techniques used to examine the various parameter lists.
- [Appendix D, “Sample configuration services XID exit routine,” on page 267](#) describes how the sample configuration services XID exit routine identifies and names unknown switched resources. The appendix also describes the connection status monitoring feature of the sample exit.
- [Appendix E, “Command verification exit routine,” on page 275](#) is an example of some of the code required in an installation-wide exit routine for screening commands that might impact critical nodes in a network.
- [Appendix F, “Sample USERVAR exit routine for TPF sessions,” on page 277](#) describes sections of code from the USERVAR exit routine that is available on SYS1.SAMPLIB.
- [Appendix G, “Sample CLU search exit routine for TPF sessions,” on page 281](#) describes sections of code from the CLU search exit routine that is available on SYS1.SAMPLIB.
- [“Architectural specifications” on page 283](#) lists documents that provide architectural specifications for the SNA protocol.
- [“Accessibility” on page 285](#) describes accessibility features to help users with physical disabilities.
- [“Notices” on page 287](#) contains notices and trademarks that are used in this information.
- [“Bibliography” on page 291](#) contains descriptions of the information in the z/OS Communications Server library.

How to use this document

To use this document, you should be familiar with the basic concepts of telecommunication, SNA, and VTAM.

How to provide feedback to IBM

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

Conventions and terminology that are used in this information

Commands in this information that can be used in both TSO and z/OS UNIX environments use the following conventions:

- When describing how to use the command in a TSO environment, the command is presented in uppercase (for example, NETSTAT).
- When describing how to use the command in a z/OS UNIX environment, the command is presented in bold lowercase (for example, **netstat**).
- When referring to the command in a general way in text, the command is presented with an initial capital letter (for example, Netstat).

All the exit routines described in this information are *installation-wide exit routines*. The installation-wide exit routines also called installation-wide exits, exit routines, and exits throughout this information.

The TPF logon manager, although included with VTAM, is an application program; therefore, the logon manager is documented separately from VTAM.

Samples used in this information might not be updated for each release. Evaluate a sample carefully before applying it to your system.

z/OS no longer supports mounting HFS data sets (The POSIX style file system). Instead, a z/OS File System (zFS) can be implemented. The term hierarchical file system, abbreviated as HFS, is defined as a data structure that has a hierarchical nature with directories and files. References to hierarchical file systems or HFS might still be in use in z/OS Communications Server publications.

Network Express and Open Systems Adapter-Express (OSA-Express) terminology:

- The Network Express feature is introduced with the IBM z17 processor family. The Network Express feature is the next generation of Open Systems Adapter (OSA) technology. The term OSA (Open Systems Adapter) is carried forward with Network Express. The IBM z17 processor supports both the Network Express and the OSA-Express7S features. In this information, when a general reference is made to OSA that applies to all these features, then the term OSA is used, and the acronym will appear in italics. This formatting style and guideline for usage for the term OSA is used throughout this document. When a distinction is necessary, then the specific feature name is used such as the Network Express feature
- The Network Express feature is defined as channel (CHPID) type OSH (Open System Adapter for Hybrid networks) that might operate in either 10 GbE or 25 GbE link speed. When this term is used in this information, the processing being described applies to either link speed. If processing is applicable to only one link speed, the full terminology, for instance, IBM 25 GbE Network Express will be used.
- Network Express is defined with new system architecture called Enhanced Queued Direct I/O (EQDIO). In this information there are many references to QDIO or OSA/QDIO. When the reference applies to both QDIO and EQDIO the reference just indicates OSA. When the reference is specific to the QDIO or EQDIO architecture, then the specific architecture is referenced, for example, OSA/QDIO or OSA/EQDIO. Some OSA references also use or include the channel type for OSA such as OSD (QDIO). When the reference applies to both features, then the term OSA is used. When a distinction is necessary then the specific channel or architecture type is used, OSD/QDIO or OSH/EQDIO.

Shared Memory Communications over Remote Direct Memory Access (SMC-R) terminology

- *RoCE* , which is a generic term representing IBM® 10 GbE RoCE Express, IBM 10 GbE RoCE Express2, IBM 25 GbE RoCE Express2, IBM 10 GbE RoCE Express3, IBM 25 GbE RoCE Express3, IBM 10 GbE Network Express and IBM 25 GbE Network Express feature capabilities. When this term is used in this information, the processing being described applies to all of these features. If processing is applicable to only one feature, the full terminology, for instance, Network Express will be used.
- RoCE Express2, which is a generic term representing an IBM RoCE Express2 feature that might operate in either 10 GbE or 25 GbE link speed. When this term is used in this information, the processing being described applies to either link speed. If processing applies to only one link speed, the full terminology, for instance, IBM 25 GbE RoCE Express2 will be used.
- RoCE Express3, which is a generic term representing an IBM RoCE Express3 feature that might operate in either 10 GbE or 25 GbE link speed. When this term is used in this information, the processing being described applies to either link speed. If processing applies to only one link speed, the full terminology, for instance, IBM 25 GbE RoCE Express3 will be used.
- Network Express, which is a generic term representing an Network Express feature that might operate in either 10 GbE or 25 GbE link speed. When this term is used in this information, the processing being described applies to either link speed. If processing is applicable to only one link speed, the full terminology, for instance, IBM 25 GbE Network Express will be used. When configured with a CHPID type of NETH, the Network Express feature may operate as an RDMA network interface card.
- RDMA network interface card (RNIC), which is used to refer to the IBM 10 GbE RoCE Express, IBM 10 GbE RoCE Express2, IBM 25 GbE RoCE Express2, IBM 10 GbE RoCE Express3, or IBM 25 GbE RoCE Express3, IBM 10 GbE Network Express or IBM 25 GbE Network Express feature.
- Shared RoCE environment, which means that the *RoCE* feature can be used concurrently, or shared, by multiple operating system instances. The feature is considered to operate in a shared RoCE environment even if you use it with a single operating system instance.

Clarification of notes

Information traditionally qualified as Notes is further qualified as follows:

Attention

Indicate the possibility of damage

Guideline

Customary way to perform a procedure

Note

Supplemental detail

Rule

Something you must do; limitations on your actions

Restriction

Indicates certain conditions are not supported; limitations on a product or facility

Requirement

Dependencies, prerequisites

Result

Indicates the outcome

Tip

Offers shortcuts or alternative ways of performing an action; a hint

Prerequisite and related information

z/OS Communications Server function is described in the z/OS Communications Server library. Descriptions of those documents are listed in “Bibliography” on page 291, in the back of this document.

Required information

Before using this product, you should be familiar with TCP/IP, VTAM, MVS, and UNIX System Services.

Softcopy information

Softcopy publications are available in the following collection.

Titles	Description
<i>IBM Z Redbooks</i>	The IBM Z® subject areas range from e-business application development and enablement to hardware, networking, Linux®, solutions, security, parallel sysplex, and many others. For more information about the Redbooks® publications, see http://www.redbooks.ibm.com/ and http://www.ibm.com/systems/z/os/zos/zfavorites/ .

Other documents

This information explains how z/OS references information in other documents.

When possible, this information uses cross-document links that go directly to the topic in reference using shortened versions of the document title. For complete titles and order numbers of the documents for all products that are part of z/OS, see *z/OS Information Roadmap (SA23-2299)*. The Roadmap describes what level of documents are supplied with each release of z/OS Communications Server, and also describes each z/OS publication.

To find the complete z/OS library, visit the *z/OS library* in *IBM Documentation* (<https://www.ibm.com/docs/en/zos>).

Relevant RFCs are listed in an appendix of the IP documents. Architectural specifications for the SNA protocol are listed in an appendix of the SNA documents.

The following table lists documents that might be helpful to readers.

Title	Number
<i>DNS and BIND</i> , Fifth Edition, O'Reilly Media, 2006	ISBN 13: 978-0596100575
<i>Routing in the Internet</i> , Second Edition, Christian Huitema (Prentice Hall 1999)	ISBN 13: 978-0130226471
<i>sendmail</i> , Fourth Edition, Bryan Costales, Claus Assmann, George Jansen, and Gregory Shapiro, O'Reilly Media, 2007	ISBN 13: 978-0596510299
<i>SNA Formats</i>	GA27-3136
<i>TCP/IP Illustrated, Volume 1: The Protocols</i> , W. Richard Stevens, Addison-Wesley Professional, 1994	ISBN 13: 978-0201633467
<i>TCP/IP Illustrated, Volume 2: The Implementation</i> , Gary R. Wright and W. Richard Stevens, Addison-Wesley Professional, 1995	ISBN 13: 978-0201633542
<i>TCP/IP Illustrated, Volume 3: TCP for Transactions, HTTP, NNTP, and the UNIX Domain Protocols</i> , W. Richard Stevens, Addison-Wesley Professional, 1996	ISBN 13: 978-0201634952
<i>TCP/IP Tutorial and Technical Overview</i>	GG24-3376
<i>Understanding LDAP</i>	SG24-4986
z/OS Cryptographic Services System SSL Programming	SC14-7495
z/OS IBM Tivoli Directory Server Administration and Use for z/OS	SC23-6788
z/OS JES2 Initialization and Tuning Guide	SA32-0991
z/OS Problem Management	SC23-6844
z/OS MVS Diagnosis: Reference	GA32-0904
z/OS MVS Diagnosis: Tools and Service Aids	GA32-0905
z/OS MVS Using the Subsystem Interface	SA38-0679
z/OS Program Directory	GI11-9848
z/OS UNIX System Services Command Reference	SA23-2280
z/OS UNIX System Services Planning	GA32-0884
z/OS UNIX System Services Programming: Assembler Callable Services Reference	SA23-2281
z/OS UNIX System Services User's Guide	SA23-2279
z/OS C/C++ Runtime Library Reference	SC14-7314
OSA-Express Customer's Guide and Reference	SA22-7935

Redbooks publications

The following Redbooks publications might help you as you implement z/OS Communications Server.

Title	Number
<i>IBM z/OS Communications Server TCP/IP Implementation, Volume 1: Base Functions, Connectivity, and Routing</i>	SG24-8096
<i>IBM z/OS Communications Server TCP/IP Implementation, Volume 2: Standard Applications</i>	SG24-8097
<i>IBM z/OS Communications Server TCP/IP Implementation, Volume 3: High Availability, Scalability, and Performance</i>	SG24-8098

Title	Number
<i>IBM z/OS Communications Server TCP/IP Implementation, Volume 4: Security and Policy-Based Networking</i>	SG24-8099
<i>IBM Communication Controller Migration Guide</i>	SG24-6298
<i>IP Network Design Guide</i>	SG24-2580
<i>Managing OS/390 TCP/IP with SNMP</i>	SG24-5866
<i>Migrating Subarea Networks to an IP Infrastructure Using Enterprise Extender</i>	SG24-5957
<i>SecureWay Communications Server for OS/390 V2R8 TCP/IP: Guide to Enhancements</i>	SG24-5631
<i>SNA and TCP/IP Integration</i>	SG24-5291
<i>TCP/IP in a Sysplex</i>	SG24-5235
<i>TCP/IP Tutorial and Technical Overview</i>	GG24-3376
<i>Threadsafe Considerations for CICS</i>	SG24-6351

Where to find related information on the Internet

z/OS

This site provides information about z/OS Communications Server release availability, migration information, downloads, and links to information about z/OS technology

<http://www.ibm.com/systems/z/os/zos/>

z/OS Internet Library

Use this site to view and download z/OS Communications Server documentation

<http://www.ibm.com/systems/z/os/zos/library/bkserv/>

z/OS Communications Server product

The page contains z/OS Communications Server product introduction

<https://www.ibm.com/products/zos-communications-server>

IBM Communications Server product support

Use this site to submit and track problems and search the z/OS Communications Server knowledge base for Technotes, FAQs, white papers, and other z/OS Communications Server information

<https://www.ibm.com/mysupport>

IBM Communications Server performance information

This site contains links to the most recent Communications Server performance reports

<http://www.ibm.com/support/docview.wss?uid=swg27005524>

IBM Systems Center publications

Use this site to view and order Redbooks publications, Redpapers, and Technotes

<http://www.redbooks.ibm.com/>

z/OS Support Community

Search the z/OS Support Community Library for Techdocs (including Flashes, presentations, Technotes, FAQs, white papers, Customer Support Plans, and Skills Transfer information)

[z/OS Support Community](#)

Tivoli® NetView for z/OS

Use this site to view and download product documentation about Tivoli NetView for z/OS

<http://www.ibm.com/support/knowledgecenter/SSZJDU/welcome>

RFCs

Search for and view Request for Comments documents in this section of the Internet Engineering Task Force website, with links to the RFC repository and the IETF Working Groups web page

<http://www.ietf.org/rfc.html>

Internet drafts

View Internet-Drafts, which are working documents of the Internet Engineering Task Force (IETF) and other groups, in this section of the Internet Engineering Task Force website

<http://www.ietf.org/ID.html>

Information about web addresses can also be found in information APAR II11334.

Note: Any pointers in this publication to websites are provided for convenience only and do not serve as an endorsement of these websites.

DNS websites

For more information about DNS, see the following USENET news groups and mailing addresses:

USENET news groups

comp.protocols.dns.bind

BIND mailing lists

<https://lists.isc.org/mailman/listinfo>

BIND Users

- Subscribe by sending mail to bind-users-request@isc.org.
- Submit questions or answers to this forum by sending mail to bind-users@isc.org.

BIND 9 Users (This list might not be maintained indefinitely.)

- Subscribe by sending mail to bind9-users-request@isc.org.
- Submit questions or answers to this forum by sending mail to bind9-users@isc.org.

The z/OS Basic Skills Information Center

The z/OS Basic Skills Information Center is a web-based information resource intended to help users learn the basic concepts of z/OS, the operating system that runs most of the IBM mainframe computers in use today. The Information Center is designed to introduce a new generation of Information Technology professionals to basic concepts and help them prepare for a career as a z/OS professional, such as a z/OS systems programmer.

Specifically, the z/OS Basic Skills Information Center is intended to achieve the following objectives:

- Provide basic education and information about z/OS without charge
- Shorten the time it takes for people to become productive on the mainframe
- Make it easier for new people to learn z/OS

To access the z/OS Basic Skills Information Center, open your web browser to the following website, which is available to all users (no login required): <https://www.ibm.com/support/knowledgecenter/zosbasics/com.ibm.zos.zbasics/homepage.html?cp=zosbasics>

Summary of changes for SNA Customization

This document contains terminology, maintenance, and editorial changes, including changes to improve consistency and retrievability. 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.

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

- None.

Deleted

The following content is deleted.

September 2025 release

- None.

Changes made in z/OS Communications Server 3.1

This information contains no technical change for this release.

Chapter 1. Writing VTAM installation-wide exit routines

This topic describes VTAM installation-wide exit routines that control session management, virtual route selection, virtual route pacing window size, session accounting, session authorization, dynamic definition of switched connections, dynamic definition of dependent logical units (LUs), command verification, USERVAR processing, directory services management, generic resource resolution, and performance monitoring.

This topic also provides information for installing VTAM installation-wide exit routines, and describes the roles of the DISPLAY and MODIFY commands for displaying, activating, deactivating, and replacing the exit routines described throughout the SNA Customization information.

Session management exit routine

The session management exit is a multi-function exit that you can use to control and manage LU-LU session-related functions. You can use the exit to authorize session establishments, obtain session accounting data, and better manage SSCP and GWPATH selection. The exit can also interact with, or replace, the functions of the NetView alias application program. You can use the exit for adjacent link station (ALS) selection for independent LUs as well.

VTAM can invoke the session management exit at certain points in processing:

- When VTAM initialization has completed
- When normal VTAM termination occurs
- During XRF session switch
- SSCP takeover
- MNPS recovery

VTAM can also invoke the session management exit to provide certain functions:

- For each session establishment before any cross-domain flows
- For each session establishment after the destination resource for the session has been determined
- To determine ordering for the adjacent SSCPs to try for each cross-network session establishment
- To perform name translation, owning SSCP determination, or CoS and logon mode translation (some of these functions are for cross-domain session establishments, others are for cross-network session establishments)
- To limit the number of interfaces with the NetView alias application program
- To determine the appropriate ALS to use as the connection for an independent LU that is the destination LU (DLU) for the session
- To modify the virtual routes (VRs) and the associated transmission priorities (TPs) that are to be used in session or HPR pipe establishment

Although the functions are provided for a specific purpose, there is no requirement that you use a function only for that specific purpose. For example, you can use the authorization function for a purpose other than authorization. Although IBM does not supply a session management exit routine, there is a sample exit in [Appendix C, “Sample session management exit,” on page 241](#).

The session management exit is useful for same-domain, cross-domain, or cross-network sessions. Some functions of the exit are used only for cross-domain or cross-network sessions.

Write a session management exit routine if you want to use session management exit functions. The exit routine operates as an internal VTAM subroutine and is either initialized when VTAM is initialized, or is activated by issuing a `MODIFY EXIT,ID=exitname,OPTION=ACT` command. See the `MODIFY EXIT`

command in [z/OS Communications Server: SNA Operation](#) for more information about this command. Install your exit in each VTAM that will use the exit. Depending on the functions that you request, your exit might be called for every session establishment. The exit is called only in those VTAMs that are active participants in the session path. The exit is called in gateway SSCPs used by the session establishment procedures, but is not called in intermediate routing nodes (IRNs).

You do not have to write a session management exit. If there is no exit, VTAM uses the internal processing supplied by IBM to perform the necessary functions. The exit capabilities are for you to use when you want to modify normal VTAM actions. For example, you might want to enable your resource naming convention to provide an algorithmic SSCP selection function. VTAM does not normally supply this function, but you can obtain this function by coding the SSCP selection function of the session management exit routine.

Even if you do write a session management exit routine, you do not have to use every possible function. For example, you might need only session authorization. You can use this function independently of any other session management exit functions.

You can use the DISPLAY EXIT command to display information regarding a session management exit. If you specify ISTEXCAA on the ID operand of the DISPLAY EXIT command, you can see which functions are active for the session management exit. You also can use a MODIFY EXIT command to activate, deactivate, or replace the session management exit routine without interrupting VTAM processing. See [“Operator commands for VTAM exit routines” on page 174](#) for more information about using the MODIFY EXIT command to modify VTAM exit routines. See [z/OS Communications Server: SNA Operation](#) for more information about these commands.

If you need to trace information relative to session management exit routine activity, use the MODIFY TRACE and MODIFY NOTRACE commands. These commands allow you to trace the session management exit's input and output. You can specify which exit functions to trace on the OPTIONS operand of the MODIFY TRACE and MODIFY NOTRACE commands. To display the trace, use the DISPLAY TRACES command.

Note: Because of the amount of generated data, the tracing of all exit functions might negatively affect performance. For this reason, use extensive tracing selectively.

For information about trace output, see [z/OS Communications Server: SNA Diagnosis Vol 1, Techniques and Procedures](#). For information about the MODIFY TRACE command, the MODIFY NOTRACE command, and the DISPLAY TRACES command, see [z/OS Communications Server: SNA Operation](#).

Examples of when the session management exit routine functions are driven are shown in [Table 1 on page 2](#). Sample flows illustrating when the functions are driven are shown in [Appendix B, “VTAM session flows,” on page 221](#). These samples include cross-network and APPN session flows. The samples do not include exit replacement or exit replaced functions.

Table 1. Example of session management exit routine functions

Function	Driven
Begin	Once immediately following VTAM initialization, when the MODIFY EXIT command is activated.
ALS selection	Once, in each host, during session initiation (if the DLU resource could be an independent LU served by this SSCP).
Adjacent SSCP selection	Once during session initiation when the SSCP table used to route the CDINIT or DSRLST is being built.
Gateway path selection	During session initiation each time a new cross-network adjacent SSCP is selected and a gateway path list is built.

Table 1. Example of session management exit routine functions (continued)

Function	Driven
Alias selection	During session initiation each time a name needs to be translated and the NetView alias application program might be called. During session initiation if the start option TRANSLAT=USERVAR is coded, indicating that alias selection should be called even when translation is not required, so that USERVAR information can be passed to the exit.
Initial authorization	Once during session initiation after the first adjacent SSCP selection. For INIT OTHER CD, this occurs after the SSCP is selected in the secondary logical unit (SLU) direction.
Secondary authorization	Once during session initiation when the DLU has been located and more information is known.
Initial accounting	Once during session initiation when all session started signals have arrived and the session is active.
Final accounting	Once during session termination.
Exit replacement	Once immediately before the exit is replaced.
Exit replaced	Once as the first invocation following exit replacement.
Virtual route selection	Called during session initiation each time VTAM builds a VR list.
HPR Virtual Route Selection	Called during session initiation each time VTAM builds a VR list for the activation of an HPR pipe.
End	Once during VTAM termination, when the MODIFY EXIT command is deactivated.

The following information covers what you need to know to write a session management exit routine. You might also want to see [Appendix C, “Sample session management exit,”](#) on page 241.

Initial register contents

When VTAM passes control to the session management exit routine, register contents are as follows:

Register 1:

Address of a variable-length list of virtual storage addresses that point to fields of parameter data.

Register 13:

Address of a standard 72-byte save area.

Register 14:

Return address.

Register 15:

Address of the entry point of the session management exit routine.

Final register contents

The following return codes are the same for all functions of the session management exit routine.

Register 15:

Return code:

Greater than X'7F' (127)

Invocation failed. The session management exit routine is deleted and deactivated.

VTAM deactivates the exit if a return code is passed back higher than X'7F'.

X'F0'–X'FF' (240–255)

Reserved.

If you use a number that is reserved, the exit response is the same as for return codes greater than X'7F'.

Return codes are valid only if they are defined for a particular function.

The other final register contents vary, depending on which function the exit performs. Final register contents are described in the information pertaining to the individual functions.

If any function other than the end function appends, VTAM issues message IST793E, and continues as though the exit did not exist. To restart the exit, restart VTAM or use the MODIFY EXIT command to reactivate the exit.

Design requirements

Follow these procedures when writing this routine:

- Use standard linkage.
- Save registers 0–14.

Consider the following restrictions when writing this routine:

- The name of the session management exit routine module should be ISTEEXCAA; however, if you have written an alternate load module, use the load module name you assigned to your replacement module.
- The entry point for this routine must be at offset zero of the load module.
- Do not issue any SVCs if this exit routine is running in SRB mode.
- This exit routine should use only conditional storage invocations. You can reduce the possibility of a VTAM abend during a storage shortage by coding conditional storage invocations.
- Data is addressable in 24- or 31-bit mode. It is advisable that you use 31-bit addressing.
- This exit routine can be above or below the 16M line. Data is always presented below the line.
- All functions in this exit routine run under abend protection. If an error occurs, VTAM issues message IST793E and continues as though the exit routine had not been coded.
- This exit routine must be re-entrant.
- This exit routine must be link-edited into the appropriate library. See [“Installing VTAM exit routines” on page 173](#) for more information.
- This exit routine operates as an internal VTAM subroutine. VTAM performance might be degraded if the routine requires lengthy processing time. While this exit routine has control, VTAM does not process any new operator requests, session initiation requests, or session termination requests for any resource.
- System waits, including implied waits for I/O operations, should be avoided. System waits can cause VTAM failure in some timing-dependent situations.
- This exit routine should not contain VTAM macroinstructions.
- This exit routine operates enabled in pageable storage. The routine gains control in supervisor state with a VTAM storage key (key 6). Errors in the routine could damage VTAM or system control blocks and modules.
- If your installation permits parallel LU-LU sessions, the session management exit routine must be capable of processing more than one request for the same LU-LU pair. You can distinguish between these sessions by using the session ID pointed to by word 6 of the input parameter list.
- At certain offsets in the parameter lists described in this topic, you will see the designation *reserved*. These reserved fields can contain unpredictable information. Therefore, your code should not refer to, or manipulate, data in these reserved fields.

- If your exit routine refers to network addresses included within the resource identification control vectors, be aware that the format of network addresses changes when using the ENHADDR=YES start option.

With ENHADDR=NO, the first 4 bytes of the 6-byte network address (bytes 0–3) are the subarea number, and the last 2 bytes (bytes 4–5) are the element address. See [Figure 1 on page 5](#) for an illustration. This format limits the number of element addresses available to each subarea to 65,536.

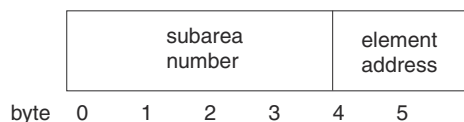


Figure 1. Network address before VTAM Version 4 Release 2

When the ENHADDR=YES start option is specified, the number of element addresses available to VTAM subareas is expanded by changing the network address format. Because the subarea number was never larger than 65,535, the first 2 bytes of the subarea number (bytes 0–1) were always 0. VTAM now uses bytes 0–1 as an index into a table of a maximum of 65,536 sets of 65,536 element addresses. (See [Figure 2 on page 5](#) for an illustration.) Network addresses with an element index of zero are called *low-order element addresses*, while those with a nonzero element index are called *high-order element addresses*. For additional information, see [z/OS Communications Server: SNA Network Implementation Guide](#).

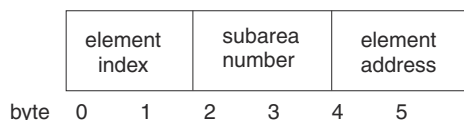


Figure 2. Network address effective in VTAM Version 4 Release 2 and later releases

When switching from ENHADDR=NO to ENHADDR=YES, if your exit routine refers to the subarea number portion of a network address, it must be modified to refer to only bytes 2–3 of the network address because bytes 0–1 are no longer part of the subarea number. If your exit routine refers to the element address portion of a network address, it must be modified to refer to bytes 0–1, as well as bytes 4–5.

Many more high-order addresses are available than low-order addresses and high-order addresses cannot be used for every session. Consequently, VTAM might change the element address and element index portions of a network address before the session becomes active in order to maximize the use of high-order element address. However, the subarea number portion of a network address never changes. This means that your exit routine might be provided with different network addresses during different SME invocations for the same session. For example, the network address of the primary LU provided to the secondary authorization function for a given session might be different than the network address that was provided to the initial authorization function.

Suggestions

When writing your exit routine, remember that any number of vectors can exist and that vector lengths might be different in different releases of VTAM. Therefore, use the length byte in the environment vectors list to identify the end of the parameter list. In addition, use a vector's length field to manipulate the length of any vector in the parameter list. If you use these length fields, it is less likely that you will have to recode your existing exit routine if bytes are added to a vector, or vectors are added to the parameter list.

Parameter lists

Several functions are defined for the session management exit routine. All functions except the begin function are optional. A function code indicates which function the exit is to process and determines which parameters VTAM passes to the exit. Function codes are described in [“Function code and related](#)

session information” on page 36. The parameters VTAM passes to the exit routine are described in the information that follows.

The possible input parameter lists are:

- Begin
- Initial authorization
- Secondary authorization
- Initial accounting
- Final accounting
- Gateway path selection
- Adjacent SSCP selection
- Alias selection
- ALS selection
- Exit replacement
- Exit replaced
- Virtual route selection
- HPR virtual route selection
- End

Table 2 on page 6, Table 3 on page 8, Table 4 on page 10, and Table 5 on page 12 provide a summary of the parameter lists. Descriptions of the parameters begin at “Parameter descriptions” on page 34. The parameters are described in the order in which they appear in the parameter lists.

Parameter list structure

Table 2. Summary of parameter lists for session management exit routine - function codes X'FE', X'00' - X'03'

FUNCTION N		BEGIN	INITIAL AUTHORIZATION	SECONDARY AUTHORIZATION	INITIAL and FINAL ACCOUNTING
Offset		Function Code X'FE'	Function Code X'00'	Function Code X'01'	Function Codes X'02' and X'03'
Dec	Hex				
0	0	Environment vectors address: See Table 20 on page 35	Environment vectors address: See Table 20 on page 35	Environment vectors address: See Table 20 on page 35	Environment vectors address: See Table 20 on page 35
4	4	Function code and related session information address: See Table 21 on page 37	Function code and related session information address: See Table 21 on page 37	Function code and related session information address: See Table 21 on page 37	Function code and related session information address: See Table 21 on page 37
8	8	User Data Field Address	User Data Field Address	User Data Field Address	User Data Field Address
12	C	Exit Options Address: See Table 22 on page 43	PLU Resource ID Control Vector Address: See Table 23 on page 46	PLU Resource ID Control Vector Address: See Table 23 on page 46	PLU Resource ID Control Vector Address: See Table 23 on page 46
16	10	Parameter string address from MODIFY EXIT, PARMS= command	SLU Resource ID Control Vector Address: See Table 23 on page 46	SLU Resource ID Control Vector Address: See Table 23 on page 46	SLU Resource ID Control Vector Address: See Table 23 on page 46

Table 2. Summary of parameter lists for session management exit routine - function codes X'FE', X'00' - X'03' (continued)

FUNCTION BEGIN			INITIAL AUTHORIZATION	SECONDARY AUTHORIZATION	INITIAL and FINAL ACCOUNTING
Offset	Function Code X'FE'		Function Code X'00'	Function Code X'01'	Function Codes X'02' and X'03'
Dec	Hex				
20	14	N/A	Session ID Address	Session ID Address	Session ID Address
24	18	N/A	Reserved	Reserved	Time of Day Address
28	1C	N/A	OLU Gateway Information Vector Address: See Table 28 on page 53	OLU Gateway Information Vector Address: See Table 28 on page 53	OLU Gateway Information Vector Address: See Table 28 on page 53
32	20	N/A	DLU Gateway Information Vector Address: See Table 31 on page 55	DLU Gateway Information Vector Address: See Table 31 on page 55	DLU Gateway Information Vector Address: See Table 31 on page 55
36	24	N/A	OLU Adjacent SSCP Vector Address: See Table 32 on page 56	OLU Adjacent SSCP Vector Address: See Table 32 on page 56	OLU Adjacent SSCP Vector Address: See Table 32 on page 56
40	28	N/A	DLU Adjacent SSCP Vector Address: See Table 33 on page 56	DLU Adjacent SSCP Vector Address: See Table 33 on page 56	DLU Adjacent SSCP Vector Address: See Table 33 on page 56
44	2C	N/A	Reserved	Failing sense code address (for information about the failing sense code, see z/OS Communications Server: IP and SNA Codes)	N/A
48	30	N/A	Reserved	Reserved	N/A
52	34	N/A	Session Initiation User Data Field Address: See Table 37 on page 63	Reserved	N/A
56	38	N/A	ILU Resource ID Control Vector Address: See Table 23 on page 46	ILU Resource ID Control Vector Address: See Table 23 on page 46	N/A
60	3C	N/A	N/A	N/A	N/A
64	40	N/A	N/A	N/A	N/A

Table 2. Summary of parameter lists for session management exit routine - function codes X'FE', X'00' - X'03' (continued)

FUNCTION		BEGIN	INITIAL AUTHORIZATION	SECONDARY AUTHORIZATION	INITIAL and FINAL ACCOUNTING
Offset		Function Code X'FE'	Function Code X'00'	Function Code X'01'	Function Codes X'02' and X'03'
Dec	Hex				
68	44	N/A	Session Authorization Data Vector Address: See Table 41 on page 66	Session Authorization Data Vector Address: See Table 41 on page 66	Session Authorization Data Vector Address: See Table 41 on page 66
72	48	VTAM Exit Services Parameter List Address: See Table 42 on page 68	VTAM Exit Services Parameter List Address: See Table 42 on page 68	VTAM Exit Services Parameter List Address: See Table 42 on page 68	VTAM Exit Services Parameter List Address: See Table 42 on page 68
76	4C	N/A	Session Management Data Area Address: See Table 45 on page 69	Session Management Data Area Address: See Table 45 on page 69	Session Management Data Area Address: See Table 45 on page 69
80	50	N/A	TCP/IP Information Control Vector Address, X'64' Address: See CV64 in <i>SNA Formats</i> , GA27–3136	TCP/IP Information Control Vector Address, X'64' Address: See CV64 in <i>SNA Formats</i> , GA27–3136	TCP/IP Information Control Vector Address, X'64' Address: See CV64 in <i>SNA Formats</i> , GA27–3136
Reg. 15 Return Code (Hex)	0		0,4,6,8,C	0,8	0

Table 3. Summary of parameter lists for session management exit routine - function codes X'04' - X'07'

FUNCTION		GATEWAY PATH SELECTION	XRF SESSION SWITCH	ADJACENT SSCP SELECTION	ALIAS SELECTION
Offset		Function Code X'04'	Function Code X'05'	Function Code X'06'	Function Code X'07'
Dec	Hex				
0	0	Environment Vectors Address: See Table 20 on page 35	Environment Vectors Address: See Table 20 on page 35	Environment Vectors Address: See Table 20 on page 35	Environment Vectors Address: See Table 20 on page 35
4	4	Function Code and Related Session Information Address: See Table 21 on page 37	Function Code and Related Session Information Address: See Table 21 on page 37	Function Code and Related Session Information Address: See Table 21 on page 37	Function Code and Related Session Information Address: See Table 21 on page 37
8	8	User Data Field Address	User Data Field Address	User Data Field Address	User Data Field Address
12	C	PLU Resource ID Control Vector Address: See Table 23 on page 46	PLU Resource ID Control Vector Address: See Table 23 on page 46	PLU Resource ID Control Vector Address: See Table 23 on page 46	PLU Resource ID Control Vector Address: See Table 23 on page 46

Table 3. Summary of parameter lists for session management exit routine - function codes X'04' - X'07'
(continued)

FUNCTION N		GATEWAY PATH SELECTION	XRF SESSION SWITCH	ADJACENT SSCP SELECTION	ALIAS SELECTION
Offset		Function Code X'04'	Function Code X'05'	Function Code X'06'	Function Code X'07'
Dec	Hex				
16	10	SLU Resource ID Control Vector Address: See Table 23 on page 46	SLU Resource ID Control Vector Address: See Table 23 on page 46	SLU Resource ID Control Vector Address: See Table 23 on page 46	SLU Resource ID Control Vector Address: See Table 23 on page 46
20	14	Session ID Address	Session ID Address	Session ID Address	Session ID Address
24	18	Reserved	Time of Day Address	Reserved	Reserved
28	1C	Gateway Path Selection List Address: See Table 29 on page 53	OLU Gateway Information Vector Address: See Table 28 on page 53	SSCP Name List Address: See Table 30 on page 55	OLU Gateway Information Vector Address: See Table 28 on page 53
32	20	N/A	DLU Gateway Information Vector Address: See Table 31 on page 55	N/A	DLU Gateway Information Vector Address: See Table 31 on page 55
36	24	N/A	OLU Adjacent SSCP Vector Address: See Table 32 on page 56	N/A	OLU Adjacent SSCP Vector Address: See Table 32 on page 56
40	28	N/A	DLU Adjacent SSCP Vector Address: See Table 33 on page 56	N/A	DLU Adjacent SSCP Vector Address: See Table 33 on page 56
44	2C	N/A	N/A	N/A	Alias Selection Input Parameter List Address: See Table 34 on page 57
48	30	N/A	N/A	N/A	Alias Selection Output Parameter List Address: See Table 36 on page 61
52	34	N/A	N/A	N/A	Reserved
56	38	N/A	N/A	N/A	USERVAR Resource ID Control Vector Address: See Table 23 on page 46
60	3C	N/A	N/A	N/A	N/A
64	40	N/A	N/A	N/A	N/A

Table 3. Summary of parameter lists for session management exit routine - function codes X'04' - X'07'
(continued)

FUNCTION		GATEWAY PATH SELECTION	XRF SESSION SWITCH	ADJACENT SSCP SELECTION	ALIAS SELECTION
Offset		Function Code X'04'	Function Code X'05'	Function Code X'06'	Function Code X'07'
Dec	Hex				
68	44	Session Authorization Data Vector Address: See Table 41 on page 66	N/A	Session Authorization Data Vector Address: See Table 41 on page 66	Session Authorization Data Vector Address: See Table 41 on page 66
72	48	VTAM Exit Services Parameter List Address: See Table 42 on page 68	VTAM Exit Services Parameter List Address: See Table 42 on page 68	VTAM Exit Services Parameter List Address: See Table 42 on page 68	VTAM Exit Services Parameter List Address: See Table 42 on page 68
76	4C	Session Management Data Area Address: See Table 45 on page 69	Session Management Data Area Address: See Table 45 on page 69	Session Management Data Area Address: See Table 45 on page 69	Session Management Data Area Address: See Table 45 on page 69
80	50	N/A	N/A	N/A	N/A
Reg. 15 Return Code (Hex)		0,4	0	0,4	0,4,8,C,10,14

Table 4. Summary of parameter lists for session management exit routine - function codes X'08' - X'0B'

FUNCTION		ALS SELECTION	EXIT REPLACEMENT	EXIT REPLACED	VIRTUAL ROUTE SELECTION
Offset		Function Code X'08'	Function Code X'09'	Function Code X'0A'	Function Code X'0B'
Dec	Hex				
0	0	Environment Vectors Address: See Table 20 on page 35	Environment Vectors Address: See Table 20 on page 35	Environment Vectors Address: See Table 20 on page 35	Environment Vectors Address: See Table 20 on page 35
4	4	Function Code and Related Session Information Address: See Table 21 on page 37	Function Code and Related Session Information Address: See Table 21 on page 37	Function Code and Related Session Information Address: See Table 21 on page 37	Function Code and Related Session Information Address: See Table 21 on page 37
8	8	User Data Field Address	User Data Field Address	User Data Field Address	User Data Field Address
12	C	PLU Resource ID Control Vector Address: See Table 23 on page 46	N/A	N/A	PLU Resource ID Control Vector Address: See Table 23 on page 46
16	10	SLU Resource ID Control Vector Address: See Table 23 on page 46	Parameter string address from MODIFY EXIT, PARMS=command	Parameter string address from MODIFY EXIT, PARMS=command	SLU Resource ID Control Vector Address: See Table 23 on page 46

Table 4. Summary of parameter lists for session management exit routine - function codes X'08' - X'0B'
(continued)

FUNCTION		ALS SELECTION	EXIT REPLACEMENT	EXIT REPLACED	VIRTUAL ROUTE SELECTION
Offset		Function Code X'08'	Function Code X'09'	Function Code X'0A'	Function Code X'0B'
Dec	Hex				
20	14	Session ID Address	N/A	N/A	Session ID Address
24	18	Reserved	N/A	N/A	Reserved
28	1C	OLU Gateway Information Vector Address: See Table 28 on page 53	N/A	N/A	OLU Gateway Information Vector Address: See Table 28 on page 53
32	20	Reserved	N/A	N/A	DLU Gateway Information Vector Address: See Table 31 on page 55
36	24	OLU Adjacent SSCP Vector Address: See Table 32 on page 56	N/A	N/A	OLU Adjacent SSCP Vector Address: See Table 32 on page 56
40	28	DLU Adjacent SSCP Vector Address: See Table 33 on page 56	N/A	N/A	DLU Adjacent SSCP Vector Address: See Table 33 on page 56
44	2C	Reserved	N/A	N/A	Reserved
48	30	Reserved	N/A	N/A	Reserved
52	34	Session Initiation User Data Field Address: See Table 37 on page 63	N/A	N/A	Reserved
56	38	ILU Resource ID Control Vector Address: See Table 23 on page 46	N/A	N/A	Reserved
60	3C	ALS List Information Vector Address: See Table 38 on page 63	N/A	N/A	VR/TP List Information Vector Address: See Table 40 on page 65
64	40	ALS Name Information Vector Address: See Table 39 on page 65	N/A	N/A	N/A
68	44	Session Authorization Data Vector Address: See Table 41 on page 66	N/A	N/A	Session Authorization Data Vector Address: See Table 41 on page 66

Table 4. Summary of parameter lists for session management exit routine - function codes X'08' - X'0B'
(continued)

FUNCTION		ALS SELECTION	EXIT REPLACEMENT	EXIT REPLACED	VIRTUAL ROUTE SELECTION
Offset		Function Code X'08'	Function Code X'09'	Function Code X'0A'	Function Code X'0B'
Dec	Hex				
72	48	VTAM Exit Services Parameter List Address: See Table 42 on page 68	VTAM Exit Services Parameter List Address: See Table 42 on page 68	VTAM Exit Services Parameter List Address: See Table 42 on page 68	VTAM Exit Services Parameter List Address: See Table 42 on page 68
76	4C	Session Management Data Area Address: See Table 45 on page 69	N/A	N/A	Session Management Data Area Address: See Table 45 on page 69
80	50	N/A	N/A	N/A	N/A
Reg. 15 Return Code (Hex)		0,4,8,C,10,14	0	0	0,4

Table 5. Summary of parameter lists for session management exit routine - function codes X'0C' and X'FF'

FUNCTION		HPR VIRTUAL ROUTE SELECTION	END
Offset		Function Code X'0C'	Function Code X'FF'
Dec	Hex		
0	0	Environment Vectors Address: See Table 20 on page 35	Environment Vectors Address: See Table 20 on page 35
4	4	Function Code and Related Session Information Address: See Table 21 on page 37	Exit Routine Function Code Address
8	8	User Data Field Address	User Data Field Address
12	C	Reserved	N/A
16	10	Reserved	Parameter string address from MODIFY EXIT,PARMS= command
20	14	Session ID Address: See “Session identifier” on page 52	N/A
24	18	Reserved	N/A
28	1C	Reserved	N/A
32	20	Reserved	N/A
36	24	Reserved	N/A

Table 5. Summary of parameter lists for session management exit routine - function codes X'0C' and X'FF' (continued)

FUNCTION		HPR VIRTUAL ROUTE SELECTION	END
Offset		Function Code X'0C'	Function Code X'FF'
Dec	Hex		
40	28	Reserved	N/A
44	2C	Reserved	N/A
48	30	Reserved	N/A
52	34	Reserved	N/A
56	38	Reserved	N/A
60	3C	VR/TP List Information Vector Address: See Table 40 on page 65	N/A
64	40	N/A	N/A
68	44	Session Authorization Data Vector Address: See Table 41 on page 66	N/A
72	48	VTAM Exit Services Parameter List Address: See Table 42 on page 68	VTAM Exit Services Parameter List Address: See Table 42 on page 68
76	4C	Session Management Data Area Address: See Table 45 on page 69	N/A
80	50	N/A	N/A
Reg. 15 Return Code (Hex)		0,4	0

Begin function (function code X'FE')

The begin function of the session management exit routine is required; all other functions are optional. This function is processed only once. It is processed before any of the other functions and is either initialized immediately following VTAM initialization, or is activated by issuing a MODIFY EXIT,ID=*exitname*,OPTION=ACT command. See [z/OS Communications Server: SNA Operation](#) for more information about this command.

During begin function processing, the exit routine selects the other functions to process for all LU-LU sessions. If no other functions are selected at this time, the session management exit routine is not invoked for the other functions. [Table 6 on page 13](#) shows the begin function parameter list pointed to by register 1. The parameters are described in [“Parameter descriptions” on page 34](#).

Table 6. Begin function parameter list

Dec (Hex) Offset	Size (Bytes)	Description
0 (0)	4	Address of environment vectors
4 (4)	4	Address of function code and related session information

Table 6. Begin function parameter list (continued)

Dec (Hex) Offset	Size (Bytes)	Description
8 (8)	4	Address of user data field
12 (C)	4	Address of exit options
16 (10)	4	Address of a parameter string if you specify a value for the PARMS operand on the MODIFY EXIT command. The parameter string is in the form of a 2-byte length field followed by the actual character data. If you do not specify a value, this pointer will be zero.
72 (48)	4	Address of VTAM Exit Services parameter list

Final register contents

The routine must leave the register status as follows:

Registers 1–14:

Restored to entry contents.

Register 15:

Return code:

X'00'

Processing successfully completed.

X'01'–X'7F'

Exit processing did not work. The session management exit remains active.

In addition to these specific return codes, there are other codes that control the exit's disablement. See [“Final register contents” on page 3](#) for more information about those codes.

If the exit routine returns a return code that is not valid (that is, a nonzero return code) or data that is not valid as a result of begin function processing, VTAM issues message IST793E and continues as though no exit routine exists.

If the alias selection function is chosen during begin processing, the alias parameter list is obtained at this time. If the parameter list cannot be obtained, VTAM issues message IST793E and the alias selection function is not invoked to perform translations.

Initial authorization function (function code X'00')

The session management exit routine invokes this function during session initiation and during SSCP takeover. Using this function, the session management exit routine determines whether to allow a session. If initial authorization for INIT OTHER CD is selected during begin function processing, the exit routine invokes initial authorization for both CLSDST PASS (which uses INIT OTHER CD) and normal initiation processing.

If initial authorization for LU session takeover is selected during begin function processing, the exit routine invokes initial authorization for LU session takeover if the sessions being taken over were established using extended BIND protocols.

For normal initiation processing, the initial authorization function can defer the decision of allowing a session to the secondary authorization function when more information is available, for example:

- The DLU's real name
- The DLU's network identifier
- The DLU's owning SSCP

In addition, for normal initiation processing, the initial authorization function can authorize session setup and request that the secondary authorization function be scheduled if session initiation fails.

For SSCP takeover, the initial and secondary authorization features provide equivalent function.

If the initial authorization function is not selected during begin function processing, VTAM defers authorization to the secondary authorization function. If the exit routine does not process secondary authorization either, VTAM authorizes the session.

The authorization process is slightly different for CLSDST PASS (INIT OTHER CD) processing. The secondary authorization function is not driven for CLSDST PASS processing when neither the PLU nor the SLU are in the initiating LU's host; therefore, the initial authorization function cannot defer session authorization for INIT OTHER CD. This limitation applies only to the portion of the new session establishment path that is processed under the INIT OTHER CD, that is, that portion of the path between the initiating logical unit (ILU) and either the new PLU or the SLU. For the portion of the session establishment path between the new PLU and the SLU (inclusive), both initial and secondary authorization occur without limitation.

For CLSDST PASS processing, certain parameters in the initial authorization parameter list point to information about the initiating logical unit (ILU). The OLU and DLU gateway information vectors contain information about the ILU and SLU rather than the OLU and DLU. The ILU resource identifier control (RIC) vector provides the ILU name, the ILU network ID, and its owning SSCP name during CLSDST PASS processing. The ILU and SSCP names provided in the vector are the real names used by the application program; no translations occur on these names.

The ILU RIC vector is present only with CLSDST PASS (third-party initiated sessions). Even with third-party initiated sessions, the ILU RIC vector is present only when the SSCPs on the session establishment path are at a level that supports the provision of this data. When referencing the ILU RIC data, the pointer to the data should be examined to determine whether the information is available for use. The setting of the high-order bit in the last parameter indicates the end of the parameter list; consider this when determining the presence of the ILU RIC.

For CLSDST PASS sessions, the name of the resource issuing CLSDST PASS is supplied to the initial authorization function in the session initiation user data field.

Table 7 on page 15 shows the initial authorization function parameter list pointed to by register 1. The parameters are described in “Parameter descriptions” on page 34.

Table 7. Initial authorization function parameter list

Dec (Hex) Offset	Size (Bytes)	Description
0 (0)	4	Address of environment vectors
4 (4)	4	Address of function code and related session information
8 (8)	4	Address of user data field
12 (C)	4	Address of PLU resource identifier control vector
16 (10)	4	Address of SLU resource identifier control vector
20 (14)	4	Address of session ID
24 (18)	4	Reserved
28 (1C)	4	Address of OLU gateway information vector
32 (20)	4	Address of DLU gateway information vector
36 (24)	4	Address of OLU adjacent SSCP vector (for INIT OTHER CD, address of the ILU adjacent SSCP vector)
40 (28)	4	Address of DLU adjacent SSCP vector (present only for INIT OTHER CD)
44 (2C)	4	Reserved

Table 7. Initial authorization function parameter list (continued)

Dec (Hex) Offset	Size (Bytes)	Description
48 (30)	4	Reserved
52 (34)	4	Address of session initiation user data field
56 (38)	4	Address of ILU resource identifier control vector
68 (44)	4	Address of session authorization data vector
72 (48)	4	Address of VTAM exit services parameter list
76 (4C)	4	Address of session management data area
80 (50) ¹	4	Address of TCP/IP information control vector CV64

1. This offset should be tested for zero before usage. When the offset is nonzero, it represents a pointer to the TCP/IP information control vector. This vector provides the IP characteristics of a TN3270 client associated with the SLU resource.

Final register contents

The routine must leave the register status as follows:

Registers 1–14:

Restored to entry contents.

Register 15:

Return Code:

X'0'

Processing successfully completed, session setup authorized; do not schedule the secondary authorization function.

X'4'

Session setup authorized; secondary authorization is required if an INIT OTHER CD request is not being processed. This return code is valid only if the exit routine processes the secondary authorization function. Otherwise, session setup is not authorized, and the session setup fails. For INIT OTHER CD processing, this return code is equivalent to a return code of 0.

X'6'

Session setup authorized; schedule the secondary authorization function only if the session setup fails at or before CDINIT response time.

X'8'

Session setup not authorized and session setup fails; do not reroute.

X'A'

Session setup authorized; schedule the secondary authorization function only if the session setup fails at any time.

X'C'

Session setup not authorized; allow sending SSCP to reroute.

If this return code is set in the host of the origin LU, or if this return code is set during INIT OTHER CD processing, it is handled as return code 8.

X'01'–X'7F'

For other return code values, exit processing does not work. The session management exit remains active.

In addition to these specific return codes, there are other codes that control the exit's disablement. See [“Final register contents” on page 3](#) for more information about those codes.

Note: A return code that is not valid (that is, a return code not defined here) means that session setup is not authorized and the session setup fails. VTAM issues message IST793E.

Secondary authorization function (function code X'01')

The secondary authorization function can provide authorization for sessions that were deferred by the initial authorization function, or all successful session authorization requests if the exit routine does not process the initial authorization function. Secondary authorization is processed during session initiation. If the begin function specifies that the exit is to be called for LU session takeover, and the session being taken over is established using extended BIND protocols, the secondary authorization function is called. If authorization was deferred by initial authorization and the session fails before secondary authorization, the secondary authorization will be called to inform the exit of the failure. An indicator in the related session information indicates whether the invocation is because of session setup or failure.

The secondary authorization function sometimes receives more information about a session from VTAM than does the initial authorization function. The DLU's real name, network identifier, and owning SSCP might not be known at initial authorization, but are known at secondary authorization.

For normal initiation processing, the initial authorization function does not know the DLU gateway information vector (GIV) at initial authorization time. It can, however, be determined by looking at the parameter list obtained when the secondary authorization function is invoked.

For cross-domain or cross-network sessions, the secondary authorization function is processed:

- In the DLU SSCP before sending CDINIT response
- In the intermediate and OLU SSCPs during CDINIT response processing

For same-domain sessions, the function is processed before the INIT response.

If the exit routine processes neither initial nor secondary authorization functions (because they were not selected during begin function processing), all sessions are authorized.

Table 8 on page 17 shows the secondary authorization function parameter list pointed to by register 1. The parameters are described in [“Parameter descriptions” on page 34](#).

Table 8. Secondary authorization function parameter List

Dec (Hex) Offset	Size (Bytes)	Description
0 (0)	4	Address of environment vectors
4 (4)	4	Address of function code and related session information
8 (8)	4	Address of user data field
12 (C)	4	Address of PLU resource identifier control vector
16 (10)	4	Address of SLU resource identifier control vector
20 (14)	4	Address of session ID
24 (18)	4	Reserved
28 (1C)	4	Address of OLU gateway information vector
32 (20)	4	Address of DLU gateway information vector
36 (24)	4	Address of OLU adjacent SSCP vector
40 (28)	4	Address of DLU adjacent SSCP vector
44 (2C)	4	Failing sense code address ¹
48 (30)	4	Reserved

Table 8. Secondary authorization function parameter List (continued)

Dec (Hex) Offset	Size (Bytes)	Description
52 (34)	4	Reserved
56 (38)	4	Address of ILU resource identifier control vector
68 (44)	4	Address of session authorization data vector
72 (48)	4	Address of VTAM exit services parameter list
76 (4C)	4	Address of session management data area
80 (50) ²	4	Address of TCP/IP information control vector

1. This pointer is valid only if bit X'08' in byte 2 of the function code and related session information is set to one. For additional information about the failing sense code, see [z/OS Communications Server: IP and SNA Codes](#).
2. This offset should be tested for zero before usage. When the offset is nonzero, it represents a pointer to the TCP/IP information control vector. This vector provides the IP characteristics of a TN3270 client associated with the SLU resource.

Final register contents

The routine must leave the register status as follows:

Registers 0–14:

Restored to entry contents.

Register 15:

Return Code:

X'0'

Processing successfully completed, session authorized.

X'8'

Session setup not authorized; session setup failed.

X'A'

Session setup authorized; schedule the secondary authorization function only if the session setup fails at any time.

X'01'–X'7F'

For other return code values, exit processing did not work. The session management exit remains active.

In addition to these specific return codes, there are other codes that control the exit's disablement. See [“Final register contents” on page 3](#) for more information about those codes.

Note: A return code that is not valid (that is, a return code not defined here) means that session setup is not authorized and the session setup fails. VTAM issues message IST793E.

Initial and final accounting functions (function codes X'02' and X'03)

The initial accounting function is processed during session initiation. This includes MNPS session recovery (new) and SSCP takeover (existing) of sessions. The final accounting function is processed during session termination. If the initial accounting function is selected for processing, the final accounting function is also selected for processing. These functions pass accounting information only. No results are returned to VTAM.

Table 9 on page 19 shows the accounting function parameter list (for both initial and final accounting functions) pointed to by register 1. The parameters are described in [“Parameter descriptions” on page 34](#).

Table 9. Accounting function parameter list

Dec (Hex) Offset	Size (Bytes)	Description
0 (0)	4	Address of environment vectors
4 (4)	4	Address of function code and related session information
8 (8)	4	Address of user data field
12 (C)	4	Address of PLU resource identifier control vector
16 (10)	4	Address of SLU resource identifier control vector
20 (14)	4	Address of session ID
24 (18)	4	Address of time of day field
28 (1C)	4	Address of OLU gateway information vector
32 (20)	4	Address of DLU gateway information vector
36 (24)	4	Address of OLU adjacent SSCP vector
40 (28)	4	Address of DLU adjacent SSCP vector
68 (44)	4	Address of session authorization data vector
72 (48)	4	Address of VTAM exit services parameter list
76 (4C)	4	Address of session management data area
80 (50) ¹	4	Address of TCP/IP information control vector

1. This offset should be tested for zero before usage. When the offset is nonzero, it represents a pointer to the TCP/IP information control vector. This vector provides the IP characteristics of a TN3270 client associated with the SLU resource.

Final register contents

The routine must leave the register status as follows:

Registers 0–14:

Restored to entry contents.

Register 15:

Return code:

X'0'

Processing successfully completed.

X'01'–X'7F'

Exit processing did not work. The session management exit remains active.

In addition to these specific return codes, there are other codes that control the exit's disablement. See [“Final register contents” on page 3](#) for more information about those codes.

Note: A return code that is not valid (that is, a nonzero return code) does not affect processing. VTAM issues message IST793E.

Gateway path selection function (function code X'04')

Use the gateway path selection function to shorten or reorder the list of gateway paths determined by GWPATH definition statements. (See [z/OS Communications Server: SNA Resource Definition Reference](#) for an explanation of the GWPATH statement.) If this function is not selected during the begin function processing, VTAM selects the gateway NCP from the GWPATH statements defined for the selected SSCP.

Table 10 on page 20 shows the parameter list pointed to by register 1. The parameters are described in “Parameter descriptions” on page 34.

Table 10. Gateway path selection function parameter list

Dec (Hex) Offset	Size (Bytes)	Description
0 (0)	4	Address of environment vectors
4 (4)	4	Address of function code and related session information
8 (8)	4	Address of user data field
12 (C)	4	Address of PLU resource identifier control vector
16 (10)	4	Address of SLU resource identifier control vector
20 (14)	4	Address of session ID
24 (18)	4	Reserved
28 (1C)	4	Address of gateway path selection list
68 (44)	4	Address of session authorization data vector
72 (48)	4	Address of VTAM exit services parameter list
76 (4C)	4	Address of session management data area

Final register contents

The routine must leave the register status as follows:

Registers 0–14:

Restored to entry contents.

Register 15:

Return Code:

X'0'

Processing successfully completed, proceed with session setup; the gateway path list returned by the exit routine is used. Modification of the gateway path list by the exit routine is optional.

X'4'

Proceed with session setup; the original gateway path list (determined by the GWPATH definition statements) is used. The SSCP-SSCP GWPATH is first in the list.

X'8'

Proceed with session setup:

- If OLU GWPATH selection is being performed, use the original list.
- If DLU GWPATH selection is being performed, defer selection to the adjacent gateway SSCP in the DLU direction. The adjacent SSCP must be capable of performing GWPATH selection for this LU-LU session.

X'01'–X'7F'

For other return code values, exit processing did not work. The session management exit remains active.

In addition to these specific return codes, there are other codes that control the exit's disablement. See “Final register contents” on page 3 for more information about those codes.

Note: If the exit returns a return code that is not valid (that is, a return code not defined here), the original gateway path list determined by the GWPATH statements is used for gateway path selection. The original gateway path list is also used if the returned list is empty or larger than the list passed to the routine. VTAM issues message IST793E.

XRF session switch function (function code X'05')

VTAM calls the session management exit routine with this function to process a session switch by the extended recovery facility (XRF) application program. The exit routine is driven for this function when the application program has completed its takeover of a session. The XRF session switch function notifies the exit routine of a change in session status.

An indicator in the related session information (see [Table 21 on page 37](#)) identifies whether a session is a backup XRF session. This indicator applies to all function codes except the begin and end functions.

[Table 11 on page 21](#) shows the XRF session switch function parameter list that is pointed to by register 1. These parameters are described in [“Parameter descriptions” on page 34](#).

Table 11. XRF session switch function parameter list

Dec (Hex) Offset	Size (Bytes)	Description
0 (0)	4	Address of environment vectors
4 (4)	4	Address of function code and related session information
8 (8)	4	Address of user data field
12 (C)	4	Address of PLU resource identifier control vector
16 (10)	4	Address of SLU resource identifier control vector
20 (14)	4	Address of session ID
24 (18)	4	Address of time of day field
28 (1C)	4	Address of OLU gateway information vector
32 (20)	4	Address of DLU gateway information vector
36 (24)	4	Address of OLU adjacent SSCP vector
40 (28)	4	Address of DLU adjacent SSCP vector
72 (48)	4	Address of VTAM exit services parameter list
76 (4C)	4	Address of session management data area

Final register contents

The routine must leave the register status as follows:

Registers 0–14:

Restored to entry contents.

Register 15:

Return code:

X'0'

Processing successfully completed.

X'01'–X'7F'

Exit processing did not work. The session management exit remains active.

In addition to these specific return codes, there are other codes that control the exit's disablement. See [“Final register contents” on page 3](#) for more information about those codes.

Note: A return code that is not valid (that is, a nonzero return code) does not affect processing. VTAM issues message IST793E.

Adjacent SSCP selection function (function code X'06')

VTAM invokes the adjacent SSCP selection function, during the processing of an LU-LU session request, to select the SSCP to which the request is routed. This function enables you to shorten or reorder the list of SSCPs from which the next SSCP used in session setup is chosen during CDINIT or DSRLST routing. This function is invoked for DSRLST routing only if DSRLST is specified in the exit options and enabled during the begin function. If no adjacent SSCPs exist, this function is not invoked.

The session management exit routine receives as input a list of adjacent SSCPs, and any additional SSCPs that VTAM considers appropriate for this LU-LU session. These additions include the owning SSCP and prior successful adjacent SSCPs. The order in the list of adjacent SSCPs received by the exit is based on the customer start parameters for SSCP tables. If PRIORITY is specified for the SSCPORD start option, the list is passed to the exit in VTAM-specified order. If DEFINED is specified for SSCPORD, the list is passed to the exit in user-defined order.

By using the dynamic adjacent SSCP table function with the adjacent SSCP selection function, you can maintain control of session routing without having to define adjacent SSCP tables. The adjacent SSCP selection function is provided with a list of all active adjacent SSCPs, which it alters as necessary, to control session routing.

APPN considerations for adjacent SSCP selection

BSC 3270 devices must use subarea only for their sessions. When the SME adjacent SSCP selection function is called for a BSC session partner, the list of SSCPs passed to the exit will not contain ISTAPNCP, even though it was coded in the adjacent SSCP table.

Table 12 on page 22 shows the adjacent SSCP selection function parameter list pointed to by register 1. The parameters are described in [“Parameter descriptions”](#) on page 34.

Table 12. Adjacent SSCP selection function parameter list

Dec (Hex) Offset	Size (Bytes)	Description
0 (0)	4	Address of environment vectors
4 (4)	4	Address of function code and related session information
8 (8)	4	Address of user data field
12 (C)	4	Address of PLU resource identifier control vector
16 (10)	4	Address of SLU resource identifier control vector
20 (14)	4	Address of session ID
24 (18)	4	Reserved
28 (1C)	4	Address of SSCP name list
68 (44)	4	Address of session authorization data vector
72 (48)	4	Address of VTAM exit services parameter list
76 (4C)	4	Address of session management data area

Final register contents

The routine must leave the register status as follows:

Registers 0–14:

Restored to entry contents.

Register 15:

Return Code:

X'0'

Processing successfully completed, proceed with session setup; use SSCP name list returned by this function. Changing the SSCP name list is optional.

X'4'

Proceed with session setup; use the original adjacent SSCP list received by the exit routine.

X'8'

Session setup not authorized and session setup fails; do not reroute.

X'C'

Session setup not authorized; allow sending SSCP to reroute.

X'01'–X'7F'

For other return code values, exit processing did not work. The session management exit remains active.

In addition to these specific return codes, there are other codes that control the exit's disablement. See [“Final register contents” on page 3](#) for more information about those codes.

Note: If the exit returns a return code that is not valid (that is, a return code not defined here) or if the exit routine abends, the session initiation request continues to be processed using the original list of adjacent SSCPs received by the exit routine. VTAM also uses the original adjacent SSCP list if the returned list is not valid, empty, or larger than the list passed to the routine. VTAM issues message IST793E.

Alias selection function (function code X'07')

This function is processed along the session setup path during session initiation. The alias selection function can be called during:

- INIT OTHER CD processing to translate the PLU and the SLU names, SLU logon mode name, PLU- and SLU-owning SSCP names, and SLU class-of-service (CoS) name.
- CDINIT request routing processing to translate the DLU real names, OLU alias name, DLU logon mode and CoS names, and associated LU alias names, and to determine the DLU-owning SSCP name.
- CDINIT response time to translate the CoS names, associated LU alias names, and USERVAR names, and when requested by the setting of start option TRANSLAT=USERVAR. TRANSLAT=USERVAR allows the exit to be invoked even when translation is not required, so that USERVAR information can be passed to the exit.
- DSRLST routing time to translate the DLU real names and the USERVAR names, and to determine the DLU-owning SSCP names.

If the exit returns a translated name, subsequent hosts do not request translation for that name. The translated names that the exit returns vary, depending on the network configuration and the prior session establishment.

When the first logon to a DLU is attempted, VTAM invokes the alias selection function to translate the DLU name. After the session is established, logon attempts for subsequent sessions continue to cause VTAM to invoke the alias function, but VTAM does not pass the DLU name to the exit for translation. This is because the DLU real network ID and real name are already known, either through prior translation or a predefined real CDRSC.

Examples of session flows for the alias selection function are shown in [“Session flows for alias selection function” on page 62](#).

Table 13 on page 23 shows the alias selection function parameter list pointed to by register 1. The parameters are described in [“Parameter descriptions” on page 34](#).

Table 13. Alias selection function parameter list

Dec (Hex) Offset	Size (Bytes)	Description
0 (0)	4	Address of environment vectors

Table 13. Alias selection function parameter list (continued)

Dec (Hex) Offset	Size (Bytes)	Description
4 (4)	4	Address of function code and related session information
8 (8)	4	Address of user data field
12 (C)	4	Address of PLU resource identifier control vector
16 (10)	4	Address of SLU resource identifier control vector
20 (14)	4	Address of session ID
24 (18)	4	Reserved
28 (1C)	4	Address of OLU gateway information vector
32 (20)	4	Address of DLU gateway information vector
36 (24)	4	Address of OLU adjacent SSCP vector
40 (28)	4	Address of DLU adjacent SSCP vector
44 (2C)	4	Address of alias selection input parameter list
48 (30)	4	Address of alias selection output parameter list
52 (34)	4	Reserved
56 (38)	4	Address of USERVAR resource identifier control vector (can be 0)
68 (44)	4	Address of session authorization data vector
72 (48)	4	Address of VTAM exit services parameter list
76 (4C)	4	Address of session management data area

Final register contents

The routine must leave the register status as follows:

Registers 1–14:

Restored to entry contents.

Register 15:

Return Code:

X'0'

Processing successfully completed, no information has been translated; do not invoke the NetView alias application program.

X'4'

No information has been translated; invoke the NetView alias application program to translate all data.

X'8'

Some information has been translated; do not invoke the NetView alias application program.

X'C'

Some information has been translated; invoke the NetView alias application program to translate the remaining information.

X'10'

Session initiation cannot continue through this SSCP; allow the sending SSCP to reroute the request.

X'14'

Session initiation cannot continue through this SSCP; do not allow the sending SSCP to reroute the request. The session will fail to set up.

Note: Only CDINIT and DSRLST requests can be rerouted. All other RU types fail.

X'01'–X'7F'

For other return code values, exit processing did not work. The session management exit remains active.

In addition to these specific return codes, there are other codes that control the exit's disablement. See [“Final register contents”](#) on page 3 for more information about those codes.

If the exit routine returns a return code that is not valid (that is, a return code not defined here), VTAM issues message IST793E and continues as though no exit routine exists. The NetView alias application program, if active, is invoked to perform translation.

Adjacent link station selection function (function code X'08')

With the adjacent link station (ALS) selection function, VTAM can dynamically determine or select the ALS to use for an independent LU that serves as the destination logical unit (DLU) in a session. ALS selection is called only if the DLU is the SLU. The DLU cannot be a local dependent LU or an application program residing in this VTAM.

If you code your session management exit routine to support the ALS selection function, VTAM invokes the function when it receives an INIT request or a DSRLST request. This function is invoked for DSRLST routing only if DSRLST is specified in the exit options and enabled during the begin function. In the origin logical unit (OLU) domain, the ALS selection is the first function invoked for the session. In other domains, VTAM invokes the function following the session management alias selection function that determines the DLU real name and network ID. VTAM invokes the function for INIT OTHER CD if the new PLU is not located in this domain and the SLU can potentially be an independent LU.

The session management exit provides a list of potential adjacent link stations an independent LU can use for this session. The ALS list is passed to the exit in the ALS list information vector. An ALS determines the boundary function link an independent LU uses to connect to the subarea network. By selecting an ALS, the exit can dynamically determine whether the session is to use a boundary function connection from this SSCP, or route to another SSCP to complete the connection.

The ALS selection function also allows you to specify an ALS for a session that is not in the current ALS list. The ALS name information vector returns a PU name to use for the connection. The ALS (the PU name) specified in the vector does not have to be in the list passed to the exit. In fact, you can code the ALS selection function such that the ALS specified in the name vector overrides the current ALS list. If the specified PU name is not in the current ALS list associated with the LU, you can set an indicator in the ALS name information vector to add the PU name to the current list.

You can use the ALS selection function to select a different ALS for each session where the independent LU is the DLU. This enables a DLU to have multiple concurrent session paths from the network. By default, VTAM invokes the ALS selection function whenever the SLU resource is a cross-domain resource (CDRSC) that is considered to be an independent LU served by this SSCP. If you want to perform ALS specification and selection without any prior indication that the CDRSC represents an independent LU, you can specify that ALS selection is called for all CDRSCs that could be independent LUs. This eliminates the requirement to predefine an ALS for the resource. If your installation does not need this feature, but instead defines an ALS for each independent LU (or otherwise indicates that this CDRSC is for an independent LU), you can specify that ALS selection is allowed only when the DLU is considered to be an independent LU. This can reduce the number of invocations of the exit. Whether the ALS selection function is invoked for all CDRSCs that could be independent LUs, or only for DLUs considered to be independent LUs, is specified in the exit options and enabled during the begin function.

After the ALS is selected, through either the current ALS list or the session management exit, session setup continues. If the chosen PU becomes unavailable after setup begins, VTAM does not attempt to select another ALS for the session. The session must be restarted.

Table 14 on page 26 shows the ALS selection function parameter list pointed to by register 1. The parameters are described in “Parameter descriptions” on page 34.

Table 14. ALS selection function parameter list

Dec (Hex) Offset	Size (Bytes)	Description
0 (0)	4	Address of environment vectors
4 (4)	4	Address of function code and related session information
8 (8)	4	Address of user data field
12 (C)	4	Address of PLU resource identifier control vector
16 (10)	4	Address of SLU resource identifier control vector
20 (14)	4	Address of session ID
24 (18)	4	Reserved
28 (1C)	4	Address of OLU gateway information vector
32 (20)	4	Reserved
36 (24)	4	Address of OLU adjacent SSCP vector
40 (28)	4	Address of DLU adjacent SSCP vector
44 (2C)	4	Reserved
48 (30)	4	Reserved
52 (34)	4	Address of session initiation user data field
56 (38)	4	Address of ILU resource identifier control vector
60 (3C)	4	Address of ALS list information vector
64 (40)	4	Address of ALS name information vector
68 (44)	4	Address of session authorization data vector
72 (48)	4	Address of VTAM exit services parameter list
76 (4C)	4	Address of session management data area

Final register contents

The routine must leave the register status as follows:

Registers 1–14:

Restored to entry contents

Register 15:

Return Code:

X'0'

Processing successfully completed. The exit did not specify an ALS. Choose the ALS from the current list. If no entry in the current list is available for use, the session request should be routed to a cross-domain link.

X'4'

The exit specified an ALS. However, if the specified ALS is not available for use, choose the ALS from the current list. If no entry in the current list is available for use, route the session request to a cross-domain link.

X'8'

The exit did not specify an ALS. Choose the ALS from the current list. If no entry in the current list is available for use, do not route the session request to a subarea cross-domain link.

X'C'

The exit specified an ALS. However, if the ALS is not available for use, choose the ALS from the current list. If no entry in the current list is available for use, do not route the session request to a subarea cross-domain link.

X'10'

The exit specified an ALS. If the ALS is not available for use, do not choose the ALS from the current list and do not route the session request to a subarea cross-domain link.

X'14'

The exit did not specify an ALS. Route the session request to a cross-domain link.

X'01'–X'7F'

For other return code values, exit processing did not work. The session management exit remains active.

In addition to these specific return codes, there are other codes that control the exit's disablement. See [“Final register contents” on page 3](#) for more information about those codes.

Note: If the exit returns a return code that is not valid (that is, a return code not defined here), the session initiation request continues to be processed using the current ALS list specified to the exit routine. VTAM issues message IST793E.

You can use the return codes to manipulate the processing of the ALS selection function. You can set a return code that causes VTAM to choose an ALS from the default list, choose the ALS returned in the ALS name information vector, or route the session cross-domain using a virtual route path. There are also return codes to handle cases where the selected ALS is unavailable.

VTAM does not perform ALS selection for an independent LU that is both the DLU and the PLU for the session. In that case, VTAM forwards the session request to an adjacent SSCP to locate the destination resource. Because VTAM does not drive ALS selection for such session requests, you cannot use the ALS return codes to affect routing for all resources. Instead, if you do not want the session request to be routed to a cross-domain link, you can use the initial authorization function described in [“Initial authorization function \(function code X'00'\)” on page 14](#).

APPN considerations for adjacent link station selection

In an APPN environment, the ALS selection function helps determine whether VTAM should attempt to locate a resource in the APPN network. There are other significant considerations as well; for example, see [z/OS Communications Server: SNA Resource Definition Reference](#) for information about the SORDER start option and about adjacent SSCP tables.

The following two topics describe the action VTAM takes based on final register contents from the ALS selection function (see [“Final register contents” on page 26](#)).

Function specifies no alternate ALS

When an ALS selection function specifies that no alternate ALS is to be selected, one of the following courses of action applies:

- If an ALS is returned by the exit routine, VTAM determines whether the ALS is an APPN connection or a LEN connection. An ALS must be active in order to be identified as an APPN connection.
 - If the returned ALS is an APPN connection, VTAM's topology and routing services attempts to find a route through the APPN network to the logical unit. A route computed by topology and routing services might use the ALS returned by the exit routine. For information about topology and routing services, see [z/OS Communications Server: SNA Network Implementation Guide](#).
 - If VTAM is unable to find a route through APPN, or if the returned ALS is a LEN connection, and if the returned ALS (either APPN or LEN) fits one of the following descriptions, session setup, using LEN protocols, continues:

- Active
- Leased, and in the process of activation
- Switched, and pending dial
- Connectable, switched
- If the ALS is not in a usable condition, the session fails with a sense code indicating that no PU is available for the session tail.
- If no ALS is returned by the exit routine, VTAM attempts cross-domain routing.

Exit not enabled, or exit authorizes alternate ALS

When the exit routine is not enabled, or if it is enabled and indicates that an alternate ALS can be selected, all entries in the ALS list, as well as any ALS returned by the exit routine, are evaluated by VTAM to determine whether any of the ALS entries are APPN connections, and to determine which ALS is the *first best*. An ALS that is returned by the exit routine is considered for selection before entries in the ALS list are considered.

- If an APPN ALS is found, VTAM attempts to find a route through the APPN network to the logical unit. A route computed by topology and routing services might use the ALS returned by the exit routine.
- If transaction routing services is unable to find a route through APPN, or if no APPN ALS is found, the first ALS that is found that fits one of the following descriptions is selected and session setup, using LEN protocols, continues:
 - Active
 - Leased, and in the process of activation
 - Switched, and pending dial
 - Connectable, switched
- Otherwise, VTAM attempts cross-domain routing, which might be limited to APPN if the exit routine prohibits subarea cross-domain routing, based on return codes (see [“Final register contents” on page 26](#)).

Exit replacement function (function code X'09')

This function indicates that the current session management exit routine is going to be replaced with a new copy of the session management exit routine through a MODIFY EXIT,OPTION=REPL command. This function is processed immediately before the exit is replaced. The replacement will not interrupt VTAM processing. If a replace command fails, the exit for which the command was issued is inactive. For more information about replacing exits, see [“Replacing VTAM exit routines” on page 175](#). For more information about the command, see [z/OS Communications Server: SNA Operation](#).

Table 15 on page 28 shows the exit replacement function parameter list pointed to by register 1. The parameters are described in [“Parameter descriptions” on page 34](#).

Table 15. Exit replacement function parameter list

Dec (Hex) Offset	Size (Bytes)	Description
0 (0)	4	Address of environment vectors
4 (4)	4	Address of function code and related session information code
8 (8)	4	Address of user data field
16 (10)	4	Address of a parameter string if you specify a value for the PARMS operand on the MODIFY EXIT command. The parameter string is in the form of a 2-byte length field followed by the actual character data. If you do not specify a value, this pointer will be zero.

Table 15. Exit replacement function parameter list (continued)

Dec (Hex) Offset	Size (Bytes)	Description
72 (48)	4	Address of VTAM exit services parameter list

Final register contents

The routine must leave the register status as follows:

Registers 0–14:

Restored to entry contents.

Register 15:

Return Code:

0

Processing successfully completed; continue termination.

X'01'–X'7F'

For return codes greater than zero, exit processing did not work. The session management exit remains active.

In addition to these specific return codes, there are other codes that control the exit's disablement. See “Final register contents” on page 3 for more information about those codes.

If VTAM cannot find the new session management exit when it tries to load the exit, VTAM issues message IST985I.

Exit replaced function (function code X'0A')

This function indicates that the previous session management exit routine was replaced by the current session management exit routine through a MODIFY EXIT,OPTION=REPL command. This function is processed as the first invocation after the replacement. For more information about replacing exits, see “Replacing VTAM exit routines” on page 175. For more information about the command, see [z/OS Communications Server: SNA Operation](#).

Table 16 on page 29 shows the parameter list pointed to by register 1. The parameters are described in “Parameter descriptions” on page 34.

Table 16. Exit replaced function parameter list

Dec (Hex) Offset	Size (Bytes)	Description
0 (0)	4	Address of environment vectors
4 (4)	4	Address of function code and related session information code
8 (8)	4	Address of user data field
16 (10)	4	Address of a parameter string if you specify a value for the PARMS operand on the MODIFY EXIT command. The parameter string is in the form of a 2-byte length field followed by the actual character data. If you do not specify a value, this pointer will be zero.
72 (48)	4	Address of VTAM exit services parameter list

Final register contents

The routine must leave the register status as follows:

Registers 0–14:

Restored to entry contents.

Register 15:

Return Code:

0

Processing successfully completed; continue termination.

X'01'–X'7F'

For return codes greater than zero, exit processing did not work. The session management exit remains active.

In addition to these specific return codes, there are other codes that control the exit's disablement. See [“Final register contents” on page 3](#) for more information about those codes.

Virtual route selection function (function code X'0B')

This function enables you to use a session management exit routine to modify the virtual routes (VRs) and the associated transmission priorities (TPs) that are to be used in session establishment. A valid change can be made at the interchange node, gateway SSCP, or the PLU's SSCP. Possible valid changes are reordering the VR/TP list, adding VR/TP pairs to the list, or deleting VR/TP pairs from the list. When adding pairs to the list, you cannot exceed the SNA-defined maximum number of 24.

When the exit returns the modified VR/TP list, VTAM verifies the list. However, VTAM does not remove duplicate entries from the list. VTAM verifies that all VR and TP numbers in the list fall within the valid ranges of SNA values for VRs (0 – 7) and TPs (0 – 2). If VTAM encounters a value that is not valid, or if the list exceeds the allowable maximum, VTAM restores the list to its original state and uses the original list to determine routing. If the values in the modified list are valid and there are no errors in the list, VTAM uses the modified list to determine routing. If an entry for a VR/TP pair is valid but not active, VTAM selects the next pair in the list to use when attempting session establishment.

The session management exit is called for virtual route selection (function code X'0B') in two cases: in the gateway SSCP and in the PLU's SSCP.

Gateway SSCP processing of the exit occurs when a gateway SSCP is preparing to send the VR list for the session to a gateway NCP. This call to the VR selection function allows you to change VRs and TPs from the gateway NCP to the next node on the route to the SLU. When control of a gateway NCP is shared between two gateway SSCPs, only the gateway SSCP on the destination LU side of the gateway calls the function. (Shared gateway NCPs have GWCTL=SHR coded on the PCCU definition statement for both gateway SSCPs.)

PLU host processing of the exit occurs when the PLU's SSCP is preparing to send the VR list to the PLU. This is done when the PLU is an application residing in this SSCP, a CDRSC representing an independent LU, or a CDRSC representing an APPN LU. If the PLU is either an APPN LU or an independent LU, VR selection is done only when the path between the PLU boundary function and the SLU is on a virtual route. (To differentiate between a CDRSC representing a cross-domain resource and an APPN LU or an independent LU, look at the hierarchy control vector. An APPN or an independent LU has an entry for the PU representing the adjacent link station. Cross-domain resources do not have a PU.) This call to the VR selection function allows you to change VRs and TPs from the subarea of the PLU to the subarea of the next gateway NCP or boundary function on the route to the SLU.

The VR selection function might be called multiple times in the same host because the function can modify the routes between a gateway NCP and the SLU host, a gateway NCP and the PLU host, and between two gateway NCPs.

The session flow shown in [Figure 5 on page 223](#) indicates when the VR selection function is invoked and for what reason.

You can use the virtual route selection exit, ISTEVCVR, for same-network sessions. ISTEVCVR has access to session count data which the VR selection function does not, so you might have a need to use both exits. If you use both the VR selection function and ISTEVCVR, consider the following:

- If both the VR selection function and ISTEVCVR are installed, both will be called. The session management exit executes first. This means that the VR/TP list passed to ISTEVCVR might be modified by the VR selection function.
- If you plan to use ISTEVCVR for VR selection in the domain of the PLU, you should code your session management exit to return to VTAM immediately when the following two conditions are met:
 - The PLU hierarchy control vector identifies the PLU as an application program.
 - The adjacent network identifier in the virtual route/transmission priority list information vector is equal to the host network identifier in the environment vector. This prevents your VR selection function from being called in the domain of the PLU. Otherwise, the VR selection function would alter the VR/TP list before it is passed to ISTEVCVR. The changes would be cumulative.

Table 17 on page 31 shows the parameter list pointed to by register 1.

Table 17. Virtual route selection function parameter list

Dec (Hex) Offset	Size (Bytes)	Description
0 (0)	4	Address of environment vectors
4 (4)	4	Address of function code and related session information
8 (8)	4	Address of user data field
12 (C)	4	Address of PLU resource identifier control vector
16 (10)	4	Address of SLU resource identifier control vector
20 (14)	4	Address of session ID
24 (18)	4	Reserved
28 (1C)	4	Address of OLU gateway information vector
32 (20)	4	Address of DLU gateway information vector
36 (24)	4	Address of OLU adjacent SSCP vector
40 (28)	4	Address of DLU adjacent SSCP vector
44 (2C)	4	Reserved
48 (30)	4	Reserved
52 (34)	4	Reserved
56 (38)	4	Reserved
60 (3C)	4	Address of VR/TP list information vector
68 (44)	4	Address of session authorization data vector
72 (48)	4	Address of VTAM exit services parameter list
76 (4C)	4	Address of session management data area

Final register contents

The routine must leave the register status as follows:

Registers 1–14:

Restored to entry contents

Register 15:

Return Code:

X'0'

Processing successfully completed. VTAM attempts to establish the session with the modified VR/TP list.

X'4'

VTAM attempts to establish the session with the original VR/TP list defined in the CoS table. No modifications from the exit are used.

X'01'–X'7F'

For other return code values, exit processing did not work. The session management exit remains active.

In addition to these specific return codes, there are other codes that control the exit's disablement. See [“Final register contents”](#) on page 3 for more information about those codes.

Note: A return code that is not valid (that is, a return code not defined here) means that session establishment will be attempted with the original VR/TP list defined in the CoS table. VTAM issues message IST793E.

HPR virtual route selection function (function code X'0C')

This function enables you to use a session management exit routine to modify the virtual routes (VRs) and the associated transmission priorities (TPs) that are to be used in session establishment. A valid change can be made at an interchange node or the PLU's SSCP. Possible valid changes are reordering the VR/TP list, adding VR/TP pairs to the list, or deleting VR/TP pairs from the list. When adding pairs to the list, you cannot exceed the SNA-defined maximum number of 24.

When the exit returns the modified VR/TP list, VTAM verifies the list. However, VTAM does not remove duplicate entries from the list. VTAM verifies that all VR and TP numbers in the list fall within the valid ranges of SNA values for VRs (0 – 7) and TPs (0 – 2). If VTAM encounters a value that is not valid, or if the list exceeds the allowable maximum, VTAM restores the list to its original state and uses the original list to determine routing. If the values in the modified list are valid and there are no errors in the list, VTAM uses the modified list to determine routing. If an entry for a VR/TP pair is valid but not active, VTAM selects the next pair in the list to use when attempting session establishment.

During establishment of an RTP connection, the session management exit is called for HPR virtual route selection (function code X'0C') whenever the RTP connection uses a virtual route.

The session flow shown in [Figure 8 on page 227](#) indicates when the HPR VR selection function is invoked.

[Table 18 on page 32](#) shows the HPR VR selection function parameter list pointed to by register 1.

Table 18. HPR virtual route selection function parameter list

Dec (Hex) Offset	Size (Bytes)	Description
0 (0)	4	Address of environment vectors
4 (4)	4	Address of function code and related session information
8 (8)	4	Address of user data field
12 (C)	4	Reserved
16 (10)	4	Reserved
20 (14)	4	Address of session ID
24 (18)	4	Reserved
28 (1C)	4	Reserved
32 (20)	4	Reserved
36 (24)	4	Reserved

Table 18. HPR virtual route selection function parameter list (continued)

Dec (Hex) Offset	Size (Bytes)	Description
40 (28)	4	Reserved
44 (2C)	4	Reserved
48 (30)	4	Reserved
52 (34)	4	Reserved
56 (38)	4	Reserved
60 (3C)	4	Address of VR/TP list information vector
68 (44)	4	Address of session authorization data vector
72 (48)	4	Address of VTAM exit services parameter list
76 (4C)	4	Address of session management data area

Final register contents

The routine must leave the register status as follows:

Registers 1–14:

Restored to entry contents

Register 15:

Return Code:

X'0'

Processing successfully completed. VTAM attempts to establish the session with the modified VR/TP list.

X'4'

VTAM attempts to establish the session with the original VR/TP list defined in the CoS table. No modifications from the exit are used.

X'01'–X'7F'

For other return code values, exit processing did not work. The session management exit remains active.

In addition to these specific return codes, there are other codes that control the exit's disablement. See [“Final register contents” on page 3](#) for more information about those codes.

Note: A return code that is not valid (that is, a return code not defined here) means that RTP establishment will be attempted with the original VR/TP list defined in the CoS table. VTAM issues message IST793E.

End function (function code X'FF')

This function of the session management exit routine is processed after entering a HALT, a HALT QUICK, or a MODIFY EXIT,OPTION=INACT command. This function should perform any cleanup required and return processing to VTAM. HALT CANCEL causes VTAM to abend. In this case, the end function is not processed. See [z/OS Communications Server: SNA Operation](#) for more information about these commands.

Table 19 on page 34 shows the parameter list pointed to by register 1. The parameters are described in [“Parameter descriptions” on page 34](#).

Table 19. End function parameter list

Dec (Hex) Offset	Size (Bytes)	Description
0 (0)	4	Address of environment vectors
4 (4)	4	Address of exit routine function code
8 (8)	4	Address of user data field
16 (10)	4	Address of a parameter string if you specify a value for the PARMS operand on the MODIFY EXIT command. The parameter string is in the form of a 2-byte length field followed by the actual character data. If you do not specify a value, this pointer will be zero.
72 (48)	4	Address of VTAM exit services parameter list

Final register contents

The routine must leave the register status as follows:

Registers 0–14:

Restored to entry contents

Register 15:

Return Code:

X'0'

Processing successfully completed, continue termination.

X'01'–X'7F'

Exit processing did not work. The session management exit remains active.

In addition to these specific return codes, there are other codes that control the exit's disablement. See [“Final register contents” on page 3](#) for more information about those codes.

Note: A return code that is not valid (that is, a nonzero return code) causes VTAM to continue termination. VTAM issues message IST793E.

VTAM termination does not complete if the exit routine does not complete. In that case, the operator must issue a HALT CANCEL command to terminate VTAM.

HALT CANCEL causes VTAM to abend. In this case, the end function is not processed. See [z/OS Communications Server: SNA Operation](#) for more information about these commands.

Parameter descriptions

The parameter lists for each function are described in [Table 6 on page 13](#) through [Table 19 on page 34](#). All of the parameter lists are summarized in [Table 2 on page 6](#), [Table 3 on page 8](#), [Table 4 on page 10](#), and [Table 5 on page 12](#). The parameters in each list are described in the topics that follow. The parameters are described in the order they appear in the parameter lists.

Environment vector list

The environment vector list provides information specific to the host in which the exit is operating. The information includes the network ID, the SSCP name, and other information that identifies the environment in which the exit routine is operating. The vector list is preceded by a 2-byte length field.

Table 20. Environment vectors

	Dec (Hex) Offset	Size (Bytes)	Description
A header precedes the list of vectors:			
	0 (0)	2	Total length of the parameter list including this length field (m)
This header is followed by several vectors, which can be in any order.			
The network identification vector:			
	0 (0)	1	Length of the vector including this field (n)
	1 (1)	1	ID of the vector (X'06')
	2 (2)	n-2	ID of the network in which this exit routine is operating (from the NETID start option)
The SSCP name vector:			
	0 (0)	1	Length of the vector including this field (n)
	1 (1)	1	ID of the vector (X'07')
	2 (2)	n-2	SSCP name of the host in which this exit is operating. The start option SSCPNAME is required.
The host PU name vector:			
	0 (0)	1	Length of the vector including this field (n)
	1 (1)	1	ID of the vector (X'08')
	2 (2)	n-2	Name of the host in which this exit is operating (from the HOSTPU start option)
The host PU network address vector:			
	0 (0)	1	Length of the vector including this field
	1 (1)	1	ID of the vector (X'09')
	2 (2)	6	Network address of the host PU
The OLU adjacent network vector is used for the following functions:			
• Initial authorization	0 (0)	1	Length of the vector including this field (n) If this information is not applicable, the length field contains X'00' and the actual length of the vector is X'01'.
• Secondary authorization			
• Initial accounting			
• Final accounting			
• XRF session switch			
• ALS selection			
	1 (1)	1	ID of the vector (X'0A')

Table 20. Environment vectors (continued)

	Dec (Hex) Offset	Size (Bytes)	Description
	2 (2)	n-2	Network name of the adjacent network in the OLU direction ¹
The DLU adjacent network vector is used for the following functions:			
• Initial authorization (for INIT OTHER CD)	0 (0)	1	Length of the vector including this field (n)
• Secondary authorization			If this information is not applicable, the length field contains X'00' and the actual length of the vector is X'01'.
• Initial accounting			
• Final accounting			
• XRF session switch			
• Alias selection			
• ALS selection			
	1 (1)	1	ID of the vector (X'0B')
	2 (2)	n-2	Network name of the adjacent network in the DLU direction ¹
The VTAM release vector:			
	0 (0)	1	Length of the vector including this field (n)
	1 (1)	1	ID of the vector (X'0D')
	2 (2)	1	VTAM product code
	3 (3)	1	VTAM version code
	4 (4)	1	VTAM release code
	5 (5)	1	VTAM modification code

Notes:

- The network ID of this host is used if any of the following is true:
 - The resource is in this domain.
 - The adjacent SSCP in the direction of the resource is in this network.
 - This host is not a gateway SSCP.
- The network ID adjacent to this network in the direction of the resource is used if both of the following are true:
 - The adjacent SSCP in the direction of the resource is cross-network.
 - This is a gateway SSCP.

Function code and related session information

Byte 0 of this parameter contains the function code that determines which parameters VTAM passes to the exit routine.

The remaining bytes contain information about the initiator of the session, the type of request (CDINIT or DSRLST) being processed, whether the session management exit routine is being driven for takeover,

and whether a session is a backup XRF session. The bytes also contain information about the SSCP that invoked the session management exit routine, indicate whether the RU was a request or a response, and indicate whether an INIT OTHER CD or other RU was being processed. The other possible RUs are CDINIT or DSRLST. You can determine the RU by referring to the related session information field.

Table 21. Function code and related session information

Byte	Description
0	<p>Function Code:</p> <p>X'00' The exit routine has been called for initial session authorization; only partial information might be available. This function is processed for initiation or LU takeover if the sessions being taken over were established using extended BIND protocols.</p> <p>X'01' The exit routine has been called for secondary session authorization; complete session initiation information is available. This function is processed for initiation or LU takeover if the sessions being taken over were established using extended BIND protocols. An indicator in the related session information indicates whether the invocation is due to session setup or failure.</p> <p>X'02' The exit routine has been called for session accounting at session setup completion. The session is in an active state.</p> <p>X'03' The exit routine has been called for session accounting at the completion of session termination.</p> <p>X'04' The exit routine has been called for gateway path selection. The related session information defines whether the function is invoked for the DLU or OLU direction.</p> <p>X'05' The exit routine has been called for XRF session switch processing.</p> <p>X'06' The exit routine has been called for SSCP selection.</p> <p>X'07' The exit routine has been called for the alias selection function. CDINIT, DSRLST, or INIT OTHER CD is being processed by this SSCP.</p> <p>X'08' The exit routine has been called for the ALS selection function.</p> <p>X'09' The exit routine is going to be replaced with a new copy of the session management exit routine through a MODIFY EXIT,OPTION=REPL command.</p>

Table 21. Function code and related session information (continued)

Byte	Description
	Function Code:
X'0A'	The previous exit routine was replaced by the current session management exit routine through a MODIFY EXIT,OPTION=REPL command.
X'0B'	The exit routine has been called for virtual route selection. The related session information defines whether the function is invoked for the DLU direction or the OLU direction.
X'0C'	The exit routine has been called for HPR virtual route selection. The related session information defines whether the function is invoked for the DLU direction or the OLU direction.
X'FE'	The exit routine has been called to select the functions it will process. This begin function is processed only once, either during VTAM initialization or after issuing a MODIFY EXIT,ID=exitname,OPTION=ACT command.
X'FF'	The exit routine has been called to perform required cleanup during VTAM termination. This end function is processed only once, either during VTAM termination or after issuing a MODIFY EXIT,ID=exitname,OPTION=INACT command.
1	Related Session Information:
	For all functions except begin, exit replacement, exit replaced, and end (function codes X'00'–X'08'), the following flags indicate whether a session is a backup XRF session:
B'0...'	The session is not a backup XRF session. If the function code is X'05', the session is now a primary session but was previously a backup session. For all other valid function codes, the session is either a primary session or not enabled for XRF.
B'1...'	The session is a backup XRF session. If the function code is X'05', the session was previously a primary XRF session.
	For the session authorization, alias selection, or ALS selection functions (function codes X'00', X'01', X'06', X'07', or X'08'), the following flags indicate the session initiator if Byte 1 Bit 5 indicates CDINIT (not DSRLST):
B'..00'	Autologon session (VARY LOGON, LOGAPPL).
B'..01'	PLU requested the session (SIMLOGON, OPNDST ACQUIRE). (PLU is the OLU, SLU is the DLU.)
B'..10'	SLU requested the session (USS logon, INIT SELF). (SLU is the OLU, PLU is the DLU.)
B'..11'	Some other LU requested the session: third-party initiation (CLSDST PASS).

Table 21. Function code and related session information (continued)

Byte	Description
	For the session authorization or alias selection functions (function codes X'00', X'01', or X'07'), the following flag indicates whether the DLU real network ID is assumed:
B'.... 0...'	DLU real network ID is not assumed.
B'.... 1...'	DLU real network ID is assumed and is being processed.
	For the SSCP selection, alias selection, or ALS selection function (function codes X'06', X'07', or X'08'), the following flag indicates whether the function is invoked for CDINIT or DSRLST routing:
B'.... .0..'	Function is invoked for CDINIT routing.
B'.... .1..'	Function is invoked for DSRLST routing.
	For the gateway path selection function (function code X'04') or the virtual route selection function (function code X'0B'), the following flag indicates whether the function is invoked for the DLU or OLU direction:
B'.... ..0.'	Selection is invoked for OLU direction.
B'.... ..1.'	Selection is invoked for DLU direction.
	For the session authorization, initial accounting, or ALS selection functions (function codes X'00'–X'02', or X'08'), the following flag indicates whether session management exit routine functions are being driven during SSCP takeover:
B'.... ...0'	Session management exit functions are not driven for takeover.
B'.... ...1'	Session management exit functions are driven for takeover.
	Bits not shown are reserved.

Table 21. Function code and related session information (continued)

Byte	Description
2	<p>Related Session Information:</p> <p>For the initial authorization, alias selection, or ALS selection functions (function codes X'00', X'07', or X'08'), the following flag indicates whether the function is invoked for INIT OTHER CD routing, or for some other type of RU. The initial authorization function is invoked only for INIT OTHER CD and CDINIT routing.</p> <p>B'0...' Alias selection or initial authorization is invoked for other routing (CDINIT or DSRLST routing, see related description under byte 1 to determine the type of routing).</p> <p>B'1...' Alias selection or initial authorization is invoked for INIT OTHER CD routing.</p> <p>For the alias selection function (function code X'07'), the following flag indicates which SSCP requested translation:</p> <p>B'.00.' The SSCP of the ILU requested translation.</p> <p>B'.01.' The SSCP of the OLU requested translation.</p> <p>B'.10.' The SSCP of the DLU requested translation.</p> <p>B'.11.' An intermediate SSCP (neither the OLU nor the DLU) requested translation.</p>

Table 21. Function code and related session information (continued)

Byte	Description
	For the alias selection function (function code X'07'), the following flag indicates whether the function is invoked because of a request or response:
B'...0'	Alias selection is invoked because of a request.
B'...1'	Alias selection is invoked because of a response.
	For the secondary authorization function (function code X'01'), the following flag indicates whether session initiation has succeeded:
B'.... 0...'	Session setup has succeeded; a positive initiate response is being processed.
B'.... 1...'	Session setup has failed; a negative response to session initiation or a session setup failure request is being processed.
	For all functions except begin, exit replacement, exit replaced, and end, the following flag indicates whether the DSRLST that is being processed is a resource discovery search.
B'.... .0..'	DSRLST being processed is not a resource discovery search.
B'.... .1..'	DSRLST being processed is a resource discovery search.
	For all functions except begin, exit replacement, exit replaced, and end (function codes X'FE', X'09', X'0A', and X'FF'), the following flag indicates whether the session causing this exit to be driven is a CP-CP session.
B'.... ..0.'	Session driving this exit is not a CP-CP session.
B'.... ..1.'	Session driving this exit is a CP-CP session.
	For the secondary authorization function (function code X'01'), the following flag indicates whether CoS substitution was done:
B'....0'	OLU CoS substitute not used.
B'....1'	OLU CoS substitute used.
3	Related Session Information:
	For the secondary authorization function (function code X'01'), the following flag indicates whether CoS substitution was done:
B'1...'	DLU CoS substitute used.
B'0...'	DLU CoS substitute not used.

Table 21. Function code and related session information (continued)

Byte	Description
	For the session authorization function (initial authorization X'00' or secondary authorization X'01'), the following flag indicates whether session management exit routine functions are being driven during SESSST processing:
B'0..'	An LU-LU session has been established to a DLUR LU. VTAM was not aware of the session until a SESSST was received from the LU.
B'1..'	Session management exit functions are driven for SESSST.
	For the session accounting function, the following flag indicates whether session management exit routine functions are being driven for multinode persistent session recovery:
B'..0.'	Session exit management exit functions are not being driven for multinode persistent session recovery.
B'..1.'	Session exit management exit functions are being driven for multiple node persistent session recovery.
	For all functions, the following flag indicates whether VTAM Exit Services is available:
B'....0'	VTAM does not support the session management exit calling the VTAM Exit Services routine.
B'....1'	VTAM supports the session management exit calling the VTAM Exit Services routine. Exit Services is enabled and available for use.
4	Related Session Information:
	For the initial authorization function (function code X'00'), the following indicates the type of RU that initiated the session in this host.
B'1...'	Initiation came from a local resource.
B'.1..'	Initiation came in from APPN.
B'..1.'	Initiated by a CDINIT (other than a CDINT5).
B'...1'	Initiated by a CDINIT type 5.
B'.... 1...'	Initiated by a BFINIT type X'0F' from an RTP PU.
B'.... .1...'	Initiated by a BFINIT type X'00' from an APPN TG.
B'.... ..1.'	Initiated by a BFINIT type X'00' not from an APPN TG.
	The remaining bits are reserved.

User data field

The exit routine can use this 8-byte user data field, originally initialized to 0, for any purpose. For example, the exit can use this field to store the address of a storage area obtained dynamically.

VTAM saves the contents of this user field after every successful invocation of the session management exit routine. In this way, the contents are available to the exit routine the next time the routine is called.

The last 4 bytes can be used to uniquely identify an exit's level or version. These last 4 bytes are displayed following subsequent use of the DISPLAY EXIT command. See [z/OS Communications Server: SNA Operation](#) for information about this command.

Notes:

- Do not confuse this field with the session initiation user data field described in [“Session initiation user data field”](#) on page 62.
- VTAM returns the address of this field on subsequent calls, not the field itself.

Exit options

The exit options field is a 4-byte field that indicates the functions for which VTAM will call the exit routine. This field is modified by the exit routine during begin function processing. The bit definitions are described in [Table 22 on page 43](#).

Table 22. Session management exit options

Byte	Bit	Description
0	B'1...'	The exit routine will process the initial authorization function (function code X'00').
	B'0...'	The exit routine will not process the initial authorization function (function code X'00'). The default is 0.
	B'.1..'	The exit routine will process the secondary authorization function (function code X'01').
	B'.0..'	The exit routine will not process the secondary authorization function (function code X'01'). The default is 0.
	B'..1.'	The exit routine will process the initial and final accounting functions (function codes X'02' and X'03').
	B'..0.'	The exit routine will not process the initial and final accounting functions (function codes X'02' and X'03'). The default is 0.
	B'...1'	The exit routine will process the gateway path selection function (function code X'04').
	B'...0'	The exit routine will not process the gateway path selection function (function code X'04'). The default is 0.
	B'.... 1...'	The exit routine will process the end function (function code X'FF').
	B'.... 0...'	The exit routine will not process the end function (function code X'FF'). The default is 0.
	B'.... .1..'	The exit routine will process the XRF session switch function (function code X'05').
	B'.... .0..'	The exit routine will not process the XRF session switch function (function code X'05'). The default is 0.
	B'.... ..1.'	The exit routine will process the adjacent SSCP selection function (function code X'06').

Table 22. Session management exit options (continued)

Byte	Bit	Description
1	B'.... ..0.'	The exit routine will not process the adjacent SSCP selection function (function code X'06'). The default is 0.
	B'....1'	Adjacent SSCP selection is allowed for DSRLST.
	B'....0'	Adjacent SSCP selection is not allowed for DSRLST. The default is 0.
	B'1...'	Session management exit functions are allowed for takeover of LU sessions.
	B'0...'	Session management exit functions are not allowed for takeover of LU sessions. The default is 0.
	B'.1..'	Initial authorization is allowed for INIT OTHER CD.
	B'.0..'	Initial authorization is not allowed for INIT OTHER CD. The default is 0.
	B'..1.'	The exit routine will process the alias selection function (function code X'07').
	B'..0.'	The exit routine will not process the alias selection function (function code X'07'). The default is 0.
	B'...1'	The exit routine will process the ALS selection function (function code X'08').
	B'...0'	The exit routine will not process the ALS selection function (function code X'08'). The default is 0.
	B'.... 1...'	ALS selection is allowed for DSRLST. For this indicator to be valid, also indicate that the exit routine will process the ALS selection function.
	B'.... 0...'	ALS selection is not allowed for DSRLST. The default is 0. For this indicator to be valid, also indicate that the exit routine will process the ALS selection function.
	B'.... .1..'	ALS selection is allowed for all CDRSCs that could possibly be an independent LU served by this SSCP. For this indicator to be valid, also indicate that the exit routine will process the ALS selection function.
	B'.... .0..'	ALS selection is allowed only when the DLU is considered to be an independent LU. The default is 0. For this indicator to be valid, also indicate that the exit routine will process the ALS selection function.
	B'.... ..1.'	The exit routine will be driven for MODIFY EXIT processing. The default is 0.
	B'.... ..0.'	The exit routine will not be driven for MODIFY EXIT processing. The default is 0.
	B'....1'	The exit routine will process the virtual route selection function (function code X'0B'). The default is 0.

Table 22. Session management exit options (continued)

Byte	Bit	Description
	B'.... ...0'	The exit routine will not process the virtual route selection function (function code X'0B'). The default is 0.
2	B'1...'	Session accounting exit functions are allowed for multinode persistent session recovery.
	B'0...'	Session accounting exit functions are not allowed for multinode persistent session recovery.
	B'.1..'	The exit routine will process the HPR virtual route selection function (function code X'0C'). The default is 0.
	B'.0..'	The exit routine will not process the HPR virtual route selection function (function code X'0C'). The default is 0.

PLU, SLU, ILU, and USERVAR resource identifier control vectors

Additional vectors following the RIC might not exist. Your SME routine should parse for a vector key of zero, which indicates there are no more vectors.

A PLU and SLU hierarchy control vector (see Table 24 on page 49) might follow a PLU or SLU resource identifier control (RIC) vector. A RIC vector might be followed by other vectors (X'FF', X'FE', ...). However, the length of these other vectors is not included in the length of the RIC.

The addresses of the PLU and the SLU RIC vectors are passed to the exit routine for the following functions:

Code	Description
X'00'	Initial authorization
X'01'	Secondary authorization
X'02' and X'03'	Initial and final accounting
X'04'	Gateway path selection
X'05'	XRF session switch
X'06'	Adjacent SSCP selection
X'07'	Alias selection
X'08'	ALS selection

The ILU RIC vector provides information about the initiating logical unit name in CLSDST PASS processing. The ILU name is the real name of the LU; no translations are performed on the name.

The address of the ILU RIC vector is passed to the exit routine for the following functions:

Code	Description
------	-------------

X'00'

Initial authorization

X'01'

Secondary authorization

X'08'

ALS selection

The USERVAR RIC vector provides information about the USERVAR name of the DLU for this session setup request. VTAM passes the USERVAR name to the exit in the alias name field and the USERVAR type in the usage indicator field. Other fields in the USERVAR RIC vector might not be set when VTAM passes the vector to the exit.

If present, the address of the USERVAR RIC vector is passed to the exit routine for function code X'07'. The formats of the PLU, SLU, ILU, and USERVAR RIC vectors are the same.

All the information in the USERVAR RIC vector is available for the following functions:

Code**Description****X'01'**

Secondary authorization

X'02' and X'03'

Initial and final accounting

X'05'

XRF session switch

Only a portion of the information in the USERVAR RIC vector might be available for the following functions:

Code**Description****X'00'**

Initial authorization

X'04'

Gateway path selection

X'06'

Adjacent SSCP selection

X'07'

Alias selection

X'08'

ALS selection

For these functions, the real name of the destination logical unit (DLU), the name of the DLU's owning SSCP, and the alias name of the origin LU (OLU) might be unknown. If a name is unknown, its length field is set to zeros and the length field of the next vector immediately follows.

The format of the PLU, SLU, ILU, and USERVAR resource identifier control vectors is described in [Table 23](#) on page 46.

Table 23. PLU, SLU, ILU, and USERVAR resource identifier control vectors

Dec (Hex) Offset	Size (Bytes)	Description
0 (0)	1	Vector key (X'19')

Table 23. PLU, SLU, ILU, and USERVAR resource identifier control vectors (continued)

Dec (Hex) Offset	Size (Bytes)	Description
1 (1)	1	<p>Vector length (number of bytes of vector, not including the vector key and length of this length field)</p> <p>Note: If any of the name fields do not exist (cannot be determined at the time the exit is called), the name's length field is set to zeros and the length field of the next name in the RIC vector immediately follows.</p> <p>The vector data length might include the length of the LU-address vectors when the LU-address vectors are present.</p>
2 (2)	1	SSCP rerouting count (number of SSCPs remaining in the count for this session setup request). This byte is reserved for the final accounting function (X'03').
3 (3)	1	<p>Usage indicators. The flags are defined as follows:</p> <p>B'1...' This name has been translated.</p> <p>B'0...' This name has not been translated.</p> <p>B'.1..' The resource is the target resource.</p> <p>Note: The OLU or ILU is never the target resource.</p> <p>B'.0..' The resource is not the target resource.</p> <p>Note: The DLU is always the target resource.</p> <p>B'.... ..01' This USERVAR name is static.</p> <p>B'.... ..10' This USERVAR name is dynamic.</p> <p>B'.... ..11' This USERVAR name is volatile.</p> <p>Note: The USERVAR name flags are meaningful only if this is a USERVAR RIC vector.</p>
This is followed by the SSCP name:		
0 (0)	1	Length of SSCP name (m)
1 (1)	m	Symbolic name of SSCP that controls the LU
This is followed by the real network ID:		
0 (0)	1	Length of network ID (n)
1 (1)	n	Network ID of the network containing the LU

Table 23. PLU, SLU, ILU, and USERVAR resource identifier control vectors (continued)

Dec (Hex) Offset	Size (Bytes)	Description
This is followed by the real LU name:		
0 (0)	1	Length of the LU name (p)
1 (1)	p	Network name of the LU (real name)
This is followed by the alias network ID:		
0 (0)	1	Length of network ID (q)
1 (1)	q	Network ID of the network in which the following alias LU name is known
This is followed by the alias LU name:		
0 (0)	1	Length of alias LU name (r)
1 (1)	r	Alias LU name
For PLU, SLU, and USERVAR resource identifier control vectors, this can be followed by up to three LU-address vectors. See “Design requirements” on page 4 for restrictions regarding format of network addresses:		
0 (0)	1	ID of the vector (X'1A')
1 (1)	1	Vector length (number of bytes of vector, not including the vector key and length of this length field)
2 (2)	6	PLU or SLU address in this network
8 (8)	1	Reserved
9 (9)	1	Reserved
0 (0)	1	ID of the vector (X'1A')
1 (1)	1	Vector length (number of bytes of vector, not including the vector key and length of this length field)
2 (2)	6	PLU or SLU address in the adjacent network in the OLU direction
8 (8)	1	Reserved
9 (9)	1	Reserved
0 (0)	1	ID of the vector (X'1A')
1 (1)	1	Vector length (number of bytes of vector, not including the vector key and length of this length field)
2 (2)	6	PLU or SLU address in the adjacent network in the DLU direction
8 (8)	1	Reserved
9 (9)	1	Reserved

A PLU and SLU hierarchy control vector follows a PLU or SLU resource identifier control vector. The PLU and SLU hierarchy control vector is not part of the RIC vector.

Table 24. PLU and SLU hierarchy control vector

Dec (Hex) Offset	Size (Bytes)	Description
PLU and SLU hierarchy control vector header:		
0 (0)	1	ID of the vector (X'FF')
1 (1)	1	Vector length (number of bytes of vector, not including the vector key and length of this length field)
2 (2)	1	Number of hierarchy resource entries (n)
This header is followed by a variable number (n) of hierarchy resource entries:		
0 (0)	8	Resource name
8 (8)	1	Resource type
		Dec (Hex)
		1 (1) Communication controller
		2 (2) APPL major node
		3 (3) Local non-SNA major node
		4 (4) Switched major node
		5 (5) Local SNA major node
		6 (6) CDRM major node
		7 (7) CDRSC major node
		8 (8) CA major node
		9 (9) Reserved
		10 (A) CDRM
		11 (B) Reserved
		12 (C) GROUP
		31 (1F) Reserved
		32 (20) RTP major node

Table 24. PLU and SLU hierarchy control vector (continued)

Dec (Hex) Offset	Size (Bytes)	Description
8 (8)	1	13 (D) Reserved 14 (E) LINE 15 (F) Direct attachment node 16 (10) APPL 17 (11) Reserved 18 (12) PU 19 (13) Reserved 20 (14) Reserved 21 (15) Reserved 22 (16) LU 23 (17) Link station 24 (18) CDRSC 25 (19) Reserved 26 (1A) Reserved 27 (1B) Reserved 28 (1C) LAN major node 29 (1D) Packet major node 30 (1E) XCA major node
9 (9)	3	Reserved

SLU table name vector

A SLU table name vector follows a SLU hierarchy control vector. Table name vectors contain the USS table, logon mode table, and interpret table names associated with the SLU.

Table 25. SLU table name vector

Dec (Hex) Offset	Size (Bytes)	Description
Table name vector header:		
0 (0)	1	ID of the vector (X'FE')
1 (1)	1	Vector length (number of bytes of vector, not including bytes 0 and 1 of this vector header)
2 (2)	1	Number of vector entries (n)
This header is followed by a variable number (n) of table name entries:		
0 (0)	8	Table name
8 (8)	1	Table type
		Dec (Hex)
		1 (1)
		Interpret table
		2 (2)
		Logon mode table
		3 (3)
		USS table
9 (9)	3	Reserved

Note: SLU table name vectors are not accessible directly. Rather, they are accessed indirectly through the SLU RIC vector by chaining through the vector keys and length fields. Table name vectors are present only for terminal-type dependent LUs in the SLU-owning host.

PLU and SLU fully qualified associated resource name control vector

A PLU and SLU fully qualified associated resource name control vector follows the SLU table name vector in only the PLU and SLU RICs. This vector contains network-qualified names of associated resources and can contain multiple entries.

The format of this vector follows:

Table 26. PLU and SLU fully qualified associated resource name control vector

Dec (Hex) Offset	Size (Bytes)	Description
PLU and SLU associated resource vector header		
0 (0)	1	ID of the Vector (X'FD')
1 (1)	1	Length of the vector (not including this field and not including the vector ID byte)
2 (2)	1	The number of entries (n)
This header is followed by a variable number (n) of associated resource entries		
0 (0)	8	Associated resource network ID
8 (8)	8	Associated resource name

Table 26. PLU and SLU fully qualified associated resource name control vector (continued)

Dec (Hex) Offset	Size (Bytes)	Description
16 (10)	1	Resource type Dec (Hex) 1 (1) Owning CP 2 (2) Adjacent CP
17 (11)	3	Reserved

PLU and SLU adjacent link control vector

A PLU and SLU adjacent link control vector follows the fully qualified associated resource name control vector in only the PLU and SLU RICs. This vector indicates whether the adjacent link is an intersubnetwork link.

The format for this vector follows:

Table 27. PLU and SLU adjacent link control vector

Dec (Hex) Offset	Size (Bytes)	Description
1 (1)	1	ID of the vector (X'FC')
2 (2)	1	Connection Flags Bit Value B'1...' The adjacent link is defined to be an intersubnetwork link.

Session identifier

The session identifier is a unique 8-byte identifier for this session. The identifier is followed by a qualifier, which is the name and network identifier of the SSCP that generated the session identifier. The form of the qualifier is 'netid.sscpname'. The qualifier is left-adjusted and padded on the right with blanks.

Time-of-day field

This 8-byte field contains the time of day the session started (for function code X'02'), ended (for function code X'03'), or was taken over (for function code X'05'). The time-of-day field is in system format time.

OLU gateway information vector

The OLU gateway information vector (GIV) provides the gateway NCP name and CoS names for the gateway in the direction of the origin LU. This word of the parameter list is set to 0 if the session management exit routine is scheduled in the origin LU's network, because there is no such gateway NCP.

The format of the OLU gateway information vector is shown in [Table 28 on page 53](#). Its format is the same as that of the DLU gateway information vector.

Table 28. Format of OLU gateway information vector

Dec (Hex) Offset	Size (Bytes)	Description
0 (0)	1	Length of vector data (excluding this field) Note: If one of the name fields does not exist (cannot be determined at the time the exit is called), the name's length field is set to zeros and the length field of the next name in the GIV vector immediately follows.

This is followed by the gateway node name:

0 (0)	1	Length of gateway NCP name (m)
1 (1)	m	Name of the gateway NCP in the direction of the origin LU

This is followed by a CoS name:

0 (0)	1	Length of CoS name (n)
1 (1)	n	CoS name for the network on the origin LU side of this gateway NCP

This is followed by another CoS name:

0 (0)	1	Length of CoS name (p)
1 (1)	p	CoS name for the network on the destination LU side of this gateway NCP

Gateway path selection list

This word of the parameter list points to the list of gateway paths available for this LU-LU session. [Table 29 on page 53](#) shows the format of this list.

The list has entries for all the gateway paths defined to VTAM (by the GWPATH definition statements) for each gateway NCP for which the gateway SSCP is performing gateway path selection. The gateway path entries are passed to the session management exit routine in the order they are defined to VTAM. However, the gateway NCP used for an SSCP-SSCP session is always placed first in the list if this SSCP is in session with the gateway NCP. If the SSCP is not in session with the gateway NCP being used for the SSCP-SSCP session, the first entry in the selection list is the entry that matches the subarea address and the adjacent network that is being used for the SSCP-SSCP session (the adjacent network as specified on the ADJNET operand on the GWPATH statement).

Note that the gateway path selection function might be processed in one SSCP to select a gateway for an adjacent SSCP.

Table 29. Format of gateway path selection list

Dec (Hex) Offset	Size (Bytes)	Description
0 (0)	2	Total length of the gateway NCP path list (excluding this field). Do not use this field to determine the last gateway path entry.
2 (2)	8	Network ID of the adjacent SSCP

Table 29. Format of gateway path selection list (continued)

Dec (Hex) Offset	Size (Bytes)	Description
10 (A)	2	Number of gateway path selection entries (n). Use this number to determine the last gateway path entry.
This header is followed by a variable number (n) of gateway path selection entries:		
0 (0)	1	Length of the following gateway path information (excluding this field). Use this field to determine the end of this path specification and the beginning of the next one.
1 (1)	8	Name of the gateway NCP connecting this network to the adjacent network (obtained from the GWN operand on the GWPATH definition statement or determined from the SUBAREA operand on the GWPATH statement). This field is set to zeros if the GWN operand was not coded on GWPATH and this SSCP is not in session with the gateway NCP.
9 (9)	4	Subarea address of the gateway NCP in this network (obtained from the SUBAREA operand on the GWPATH definition statement or determined from the GWN operand on the GWPATH statement). This field is set to zeros if this SSCP is not in session with the gateway node and the SUBAREA operand was not coded on GWPATH. Note: If both the GWN and SUBAREA operands are coded on the GWPATH definition statement, you cannot determine whether this SSCP is in session with the gateway NCP.
13 (D)	8	Network ID of the adjacent network accessed by this gateway NCP (from the ADJNET operand on the GWPATH statement).

You can modify the gateway path selection list in the following ways:

- You can reduce the number of gateway path entries in the list by decreasing the number of entries specified in the list header. If the exit routine returns fewer alternate gateway nodes than presented to it at entry (in the gateway path selection list), VTAM uses the smaller list of entries. In its attempt to find a gateway node to use for the LU-LU session, VTAM searches that list of entries in the order the exit routine returns the entries to VTAM.
- You can change the order of the gateway path entries in the list. The exit routine returns the entries to VTAM in the same order the gateway path entries are used to attempt LU-LU session setup.

For a list to be correct, the following requirements must be met:

- Each returned gateway path entry must match one of the entries originally received. You cannot add new entries to the list.
- The number of entries returned cannot be greater than the number of entries passed to the exit.
- The adjacent NETID field of the adjacent SSCP must remain unchanged.
- The returned gateway path list must contain at least one valid entry.

If the list returned by the exit routine is not valid for any of these reasons, the original list that was sent to the exit routine is used.

The address of the gateway path selection list should remain unchanged. The modified gateway path list should use the same storage that VTAM passed to the exit routine.

SSCP name list

This parameter contains the list of names of adjacent SSCPs available for this LU-LU session.

The format of the SSCP name list is:

Table 30. Format of SSCP name list

Dec (Hex) Offset	Size (Bytes)	Description
0 (0)	2	Total length of the SSCP name list (excluding this field)
2 (2)	2	Number of SSCP names in this list (n)
This header is followed by a variable number (n) of SSCP names:		
0 (0)	8	Adjacent SSCP name

You can modify the SSCP name list in the following ways:

- You can reduce the number of SSCP name entries in the list by decreasing the number of entries specified in the list header. If the exit routine returns fewer SSCP names than presented to it at entry (in the SSCP name list), VTAM uses the smaller list of entries. In its attempt to find an SSCP name to use, VTAM searches that list of entries in the order the exit returned the entries to VTAM.
- You can change the order of the SSCP name entries in the list. The exit routine returns the entries to VTAM in the same order the SSCP name entries are used.

For a list to be correct, the following requirements must be met:

- Each SSCP name entry returned must match one of the entries originally received. You cannot add new entries to the list.
- The number of entries returned cannot be greater than the number of entries passed to the exit.
- The returned SSCP name list must contain at least one valid entry.

If the list returned by the exit routine is not valid for any of these reasons, the original list that was sent to the exit routine is used.

The address of the SSCP name list should remain unchanged. The modified SSCP name list should use the same storage that VTAM passed to the exit routine.

DLU gateway information vector

The DLU gateway information vector provides the gateway NCP name and CoS names for the gateway NCP in the direction of the destination LU. This word of the parameter list is set to 0 if the session management exit routine is scheduled in the destination LU's network, because there is no such gateway NCP.

The format of the DLU gateway information vector is the same as that of the OLU.

Table 31. Format of DLU gateway information vector

Dec (Hex) Offset	Size (Bytes)	Description
0 (0)	1	Length of vector data (excluding this field)

Table 31. Format of DLU gateway information vector (continued)

Dec (Hex) Offset	Size (Bytes)	Description
This is followed by the gateway node name:		
0 (0)	1	Length of gateway NCP name (m)
1 (1)	m	Name of the gateway NCP in the direction of the DLU
This is followed by a CoS name:		
0 (0)	1	Length of CoS name (n)
1 (1)	n	CoS name for the network on the OLU side of this gateway NCP
This is followed by another CoS name:		
0 (0)	1	Length of the following CoS name (p)
1 (1)	p	CoS name for the network on the DLU side of this gateway NCP

OLU adjacent SSCP vector

The OLU adjacent SSCP vector provides the name of the adjacent SSCP in the direction of the OLU. For INIT OTHER CD processing, this vector provides the name of the adjacent SSCP in the direction of the ILU.

This word of the parameter list is set to 0 if the session management exit routine is scheduled in the domain of the OLU, because there is no OLU adjacent SSCP.

Table 32. Format of OLU adjacent SSCP vector

Dec (Hex) Offset	Size (Bytes)	Description
0 (0)	1	Length of vector data (excluding this field)
This is followed by the adjacent SSCP name:		
0 (0)	1	Length of the adjacent name (q)
1 (1)	q	Name of the adjacent SSCP in the direction of the OLU

DLU adjacent SSCP vector

The DLU adjacent SSCP vector provides the name of the adjacent SSCP in the direction of the DLU. For INIT OTHER CD processing, this vector provides the name of the adjacent SSCP in the direction of the SLU.

This word of the parameter list is set to 0 if the session management exit routine is scheduled in the domain of the DLU, because there is no DLU adjacent SSCP.

Table 33. Format of DLU adjacent SSCP vector

Dec (Hex) Offset	Size (Bytes)	Description
0 (0)	1	Length of vector data (excluding this field)

Table 33. Format of DLU adjacent SSCP vector (continued)

Dec (Hex) Offset	Size (Bytes)	Description
This is followed by the adjacent SSCP name:		
0 (0)	1	Length of the adjacent name (q)
1 (1)	q	Name of the adjacent SSCP in the direction of the DLU

Alias selection input parameter list

The alias selection input parameter list for LU-LU sessions contains information necessary for translation and is a fixed length. Its storage is obtained during begin function processing. If storage cannot be obtained for the alias parameter list during the begin function, message IST793E is issued indicating the error and the alias selection function is not invoked. Depending on which information needs translation, different fields are set.

The format of the alias selection input parameter list is described in [Table 34 on page 57](#). The input constants in the parameter list are described in [Table 35 on page 60](#).

Table 34. Alias selection input parameter list

Dec (Hex) Offset	Size (Bytes)	Description
0 (0)	1	Flags for NetView alias application program:
		B'1...'
		NetView alias application program is currently active.
		B'0...'
		NetView alias application program is not active.
		B'.1..'
		NetView alias application program used to be active.
		B'.0..'
		NetView alias application program was never active. This bit is meaningful only if the previous bit is off.
		..xx xxxx
		Reserved
1 (1)	7	Reserved
		Start of NAME1 information
8 (8)	1	Indicates the role of the LU in the session (see Table 35 on page 60)
9 (9)	1	Indicates the primary and secondary roles of the OLU and DLU (see Table 35 on page 60)
		xxxx
		Indicates whether the LU is the OLU, DLU, or unknown
	 xxxx
		Indicates whether the LU is the PLU, SLU, or unknown
10 (A)	1	Information pertaining to the LU name relevant for translation

Table 34. Alias selection input parameter list (continued)

Dec (Hex) Offset	Size (Bytes)		Description
		X...	1 = NAME1 is the real name. 0 = NAME1 is the alias name.
		.xxx xxxx	Reserved
11 (B)	5		Reserved
16 (10)	8	NAME1	LU name to be translated
24 (18)	8	NETFROM1	Network in which NAME1 is known
32 (20)	8	NETTO1	Network to which NAME1 is to be translated. Blank or zero if exit needs to determine the network.
Start of NAME2 information			
40 (28)	1		Indicates the role of the LU in the session (see Table 35 on page 60)
41 (29)	1		Indicates the primary and secondary roles of the OLU and DLU (see Table 35 on page 60)
		xxxx	Indicates whether the LU is the OLU, DLU, or unknown
	 xxxx	Indicates whether the LU is the PLU, SLU, or unknown
42 (2A)	1		Information pertaining to the LU name relevant for translation
		X...	1 = NAME2 is the real name 0 = NAME2 is the alias name
		.xxx xxxx	Reserved
43 (2B)	5		Reserved
48 (30)	8	NAME2	LU name to be translated
56 (38)	8	NETFROM2	Network in which NAME2 is known
64 (40)	8	NETTO2	Network to which NAME2 is to be translated. Blank or zero if exit needs to determine the network.
Start of CoS/LOG information			
72 (48)	8	CoSNAME	CoS name
80 (50)	8	LOGNAME	Logon mode name
88 (58)	8	NETFROM	Network ID in which the name or names are known
96 (60)	8	NETTO	Network ID to which the name or names are to be translated
Start of NAME3 information			

Table 34. Alias selection input parameter list (continued)

Dec (Hex) Offset	Size (Bytes)		Description
104 (68)	1		Indicates the role of the LU in the session (see Table 35 on page 60)
105 (69)	1		Indicates the primary and secondary roles of the OLU and DLU (see Table 35 on page 60)
		xxxx	Indicates whether the LU is the OLU, DLU, or unknown
	 xxxx	Indicates whether the LU is the PLU, SLU, or unknown
106 (6A)	1		Information pertaining to the LU name relevant for translation
		X...	1 = NAME3 is the real name 0 = NAME3 is the alias name
		.xxx xxxx	Reserved
107 (6B)	5		Reserved
112 (70)	8	NAME3	LU name to be translated
120 (78)	8	NETFROM3	Network in which NAME3 is known
128 (80)	8	NETTO3	Network to which NAME3 is to be translated. Blank or zero if exit needs to determine the network.
Start of NAME4 information			
136 (88)	1		Indicates the role of the LU in the session (see Table 35 on page 60)
137 (89)	1		Indicates the primary and secondary roles of the OLU and DLU (see Table 35 on page 60)
		xxxx	Indicates whether the LU is the OLU, DLU, or unknown
	 xxxx	Indicates whether the LU is the PLU, SLU, or unknown
138 (8A)	1		Information pertaining to the LU name relevant for translation
		X...	1 = NAME4 is the real name 0 = NAME4 is the alias name
		.xxx xxxx	Reserved
139 (8B)	5		Reserved
144 (90)	8	NAME4	LU name to be translated
152 (98)	8	NETFROM4	Network in which NAME4 is known

Table 34. Alias selection input parameter list (continued)

Dec (Hex) Offset	Size (Bytes)		Description
160 (A0)	8	NETTO4	Network to which NAME4 is to be translated. Blank or zero if exit needs to determine the network.

The input constants in [Table 35 on page 60](#) are used in the alias selection input parameter list. These constants indicate:

- Whether the name to be translated is a real name or an alias name
- Whether the LU is the OLU or DLU
- Whether the LU is the PLU or SLU
- The role of the LU

Table 35. Constants for the alias selection input parameter list

Type	Value	Description
Settings for real name or alias name		
Bit	0	The name sent for translation is the alias name. The real name needs to be determined.
Bit	1	The name sent for translation is the real name. The alias name needs to be determined.
Settings for OLU or DLU		
Bit	0000	It cannot be determined whether the LU is the OLU or DLU.
Bit	0001	The LU is the origin logical unit for this session.
Bit	0010	The LU is the destination logical unit for this session.
Settings for PLU or SLU		
Bit	0000	It cannot be determined whether the LU is the PLU or SLU.
Bit	0001	The LU is the primary logical unit for this session.
Bit	0010	The LU is the secondary logical unit for this session.
Settings for LU role		
Hex	00	The LU is a session partner.
Hex	01	The LU is the primary printer.
Hex	02	The LU is the alternate printer.

The flags explained at offset 0 (see [Table 34 on page 57](#)) contain information that is not included in the translate inquiry. The first flag indicates whether the NetView alias application program is currently active. The second flag indicates whether the NetView alias application program was ever active. Use these bits to maintain system integrity.

Alias selection output parameter list

Translation data is returned to VTAM in the alias selection output parameter list. If a name translation is requested, but the network ID into which the name needs to be translated is not known, the network

ID must first be determined. The network ID is determined by the alias selection function and returned in the alias selection output parameter list. For CDINIT, DSRLST, and INIT OTHER CD processing for SLU information, the network ID is returned in RTNNET1. For INIT OTHER CD processing for PLU information, the network ID is returned in RTNNET2. If any names were translated, but the network ID was not known, the session fails to set up.

Table 36. Alias selection output parameter list

Dec (Hex) Offset	Size (Bytes)		Description
0 (0)	8	RTNNAME1	The name returned by the user corresponding to NAME1
8 (8)	8	RTNNAME2	The name returned by the user corresponding to NAME2
16 (10)	8	RTNCOS	The translated CoS name
24 (18)	8	RTNLOG	The translated logon mode name
32 (20)	8	RTNSSCP1	The SSCP name determined by the exit. For CDINIT and DSRLST processing, this field contains the name of the SSCP that owns the DLU. For INIT OTHER CD processing, this field contains the name of the SSCP that owns the SLU.
40 (28)	8	RTNSSCP2	The SSCP name determined by the exit. For CDINIT and DSRLST processing, this field is not set. For INIT OTHER CD processing, this field contains the name of the SSCP that owns the PLU.
48 (30)	8	RTNNET1	The network ID if the exit was to determine it.
56 (38)	8	RTNNET2	The network ID if the exit was to determine it.
64 (40)	8	RTNNAME3	The name returned by the user corresponding to NAME3
72 (48)	8	RTNNAME4	The name returned by the user corresponding to NAME4

The returned information must meet the following requirements:

- The real network ID (the network to translate the information into) must be returned if it was not sent.
- If a real network ID is returned, and the network to translate the names into was sent, the networks must be the same. The network ID cannot be altered.
- If you indicate that information was translated, but no information was returned, processing continues without updating any information. The return code determines whether the NetView alias application program will be invoked.
- The input parameter list cannot be altered.
- Blank values are valid only for the CoS and the logon mode names.
- LU, NETID, and the owning SSCP names must be padded on the right with blanks.
- If information was needed, and a network ID was returned, this network ID is used for the real name (if it was not known). This network ID is sent to the NetView alias application program to determine the rest of the unknown information, if wanted.
- All names must contain 1–8 characters in the following format:
 - First character: Alphabetic (A–Z) or the national characters @, #, or \$
 - Second through eighth characters: Alphanumeric (A–Z or 0–9) or the national characters @, #, or \$

Note: Alphabetic characters must be in uppercase.

If the list returned by the exit routine fails to meet any of these requirements, the request fails, unless otherwise indicated.

The address of the alias parameter list should remain unchanged.

The parameter list contains sections labeled NAME1, NAME2, NAME3 and NAME4. Each of these sections include information relating to the LU names, for example:

- Which LU name is requesting translation
- Role of the LU
- Role of the PLU or SLU and OLU or DLU
- Whether a real name or alias name is being sent for translation
- Whether the network ID to which the name is to be translated is known

In general, if a name is included in the input list, the exit can translate the name. This rule holds for all names except the DLU name, where the following additional guidelines apply:

- If the DLU name is an alias name, the exit can translate the DLU name to its real name and provide the owning SSCP name.
- If the DLU name is a real name and VTAM invokes the alias selection function for a response, the exit can translate the LU name to the alias name, but should not return an owning SSCP name.

NAME1 is the first name to be translated. NETFROM1 is the network in which the name to be translated (NAME1) is known. NETTO1 is the network to which NAME1 is to be translated. For example:

- NAME1 is APPLB5
- NETFROM1 is NETB
- NETTO1 is NETA

These values indicate that APPLB5 is the name known in NETB. APPLB5 in NETB must be translated to the LU name in NETA. If these values are set for the first name, the alias selection function can evaluate the values, and return the translated name in the output parameter list in field RTNNAME1.

COSNAME is the class-of-service name to be translated. LOGNAME is the logon mode name to be translated. NETFROM is the network ID in which either the logon mode name, the CoS name, or both names are known. NETTO is the network ID to which either the logon mode name, the CoS name, or both names are to be translated. For example:

- LOGNAME is INTERACT
- NETFROM is NETA
- NETTO is NETB

INTERACT in NETA is to be translated into the logon mode name as known in NETB. The translated logon mode name is returned in the output parameter list in RTNLOG.

Note: If any NETTO is unknown, VTAM requests that the network ID be provided. If the network ID is not returned, the translated information is ignored.

Session flows for alias selection function

VTAM invokes the alias selection function from a variety of places throughout session initiation and DSRLST processing. This is illustrated in [“Session flows for alias selection” on page 223](#).

Session initiation user data field

The address of this user data field is passed to the exit routine in the initial authorization function (function code X'00') parameter list and the ALS selection function (function code X'08') parameter list. This word points to a data field that contains user data specified by the originating resource and carried by the session initiation request.

The format of the session initiation user data field is shown in [Table 37 on page 63](#).

Note: Do not confuse this field with the user data field described in [“User data field” on page 43](#).

Table 37. Format of session initiation user data field

Dec (Hex) Offset	Size (Bytes)	Description
0 (0)	1	Total length of the session initiation user data
1 (1)	Up to 255	Session initiation user data

Adjacent link station list information vector

This vector contains the list of known potential adjacent link stations (ALS) for this LU-LU session. A different PU represents each ALS in the list. VTAM selects a PU name from the list. The PU determines the session path that will connect the independent LU with the subarea network. If you want the independent LU to use an ALS that is not in the current ALS list, you can code the ALS selection function to select another ALS and override the list by passing the new ALS in the ALS name information vector.

If the session management exit does not specify (through the ALS name information vector) an ALS to use, or if the specified ALS is unavailable and the return code indicates that processing is to continue, VTAM examines the ALS list and selects the most usable entry to use as the ALS. VTAM examines the ALS entries in the order they appear in the list and searches for an ALS by using the following criteria, listed in order of preference:

1. An active ALS
2. A leased ALS that is in the process of activation and considered usable
3. A switched ALS that is pending dial
4. A connectable switched ALS

If you code your exit to support the ALS selection function, VTAM invokes the function for each session. Therefore, you can choose a different ALS for each session involving an independent LU serving as a DLU.

You can code your exit to specify that an alternate ALS is not to be selected; see [“Function specifies no alternate ALS” on page 27](#) for more information.

The format of the ALS list information vector is shown in [Table 38 on page 63](#).

Table 38. ALS list information vector

Dec (Hex) Offset	Size (Bytes)	Description
0 (0)	2	Total length of the ALS list information vector (excluding this field)
2 (2)	2	Number of PU entries in the list (n). Use this number to determine the last PU entry.
4 (4)	4	Reserved

This header is followed by a variable number (n) of PU entries:

0 (0)	2	Length of the following PU entry (excluding this field). Use this field to determine the end of this PU entry and the beginning of the next one.
-------	---	--

Table 38. ALS list information vector (continued)

Dec (Hex) Offset	Size (Bytes)	Description
2 (2)	1	PU status flags B'1...' The PU is non-switched. B'0...' The PU is switched. B'.000' The PU is inactive or unknown. B'.001' The PU is connectable (CONCT). B'.010' The PU is pending active and is usable for LU- LU sessions. B'.011' The PU is active. B'.100' The PU is pending active, or pending takeover, and is not usable for LU-LU sessions. B'.... 1...' The PU supports dynamic definitions for independent LUs. B'.... 0...' The PU does not support dynamic definitions for independent LUs. B'.... .1..' The PU was dynamically associated with this LU by an inbound session request. B'.... .0..' The PU was associated with this independent LU by a MODIFY ALSLIST command or a definition statement. The remaining bits are reserved.
3 (3)	1	Reserved
4 (4)	8	PU network ID (if known)
12 (C)	8	PU name
20 (14)	4	PU subarea address
24 (18)	2	Count of independent LU (ILU) sessions associated with this PU
26 (1A)	2	Reserved

Adjacent link station name information vector

If you use the ALS selection function to choose an ALS that is not in the ALS list, the PU name returned in this vector becomes the ALS for this session. Depending on the flag settings in this vector, the PU name might be added to the ALS list for this independent LU. If you want the PU name to be added to the list,

and the list already exists, the PU name is added to the end of the list. If there is no list, one is allocated and the PU name becomes the first entry in the list.

The purpose of this vector is to provide multiple potential concurrent session paths for an independent LU. If one of the LU's potential ALSs becomes unusable, VTAM tries another ALS for subsequent activations. There is no attempt to reroute existing or pending sessions that are using the ALS that has failed.

You can code your exit to specify that an alternate ALS is not to be selected; see [“Function specifies no alternate ALS”](#) on page 27 for more information.

The PU whose name is returned in this vector must be one of the following:

- An active ALS
- A leased ALS that is in the process of activation and considered usable
- A switched ALS that is pending dial
- A connectable switched ALS

Also, the PU must be either an APPN node or a LEN node. If no PU meets these requirements, you can use a return code from the ALS selection function to specify that VTAM can either choose an ALS from the list, or route the session cross-domain, and session setup can continue. You can also use a return code to specify that VTAM will deny the session activation attempt when the PU does not meet the requirements.

The format of the ALS name information vector is shown in [Table 39 on page 65](#).

Table 39. ALS name information vector

Dec (Hex) Offset	Size (Bytes)	Description
0 (0)	2	Total length of the ALS name information vector
2 (2)	1	Reserved
3 (3)	1	ALS list flags B'1...' Add the PU name that follows to the default ALS list. B'0...' Do not add the PU name that follows to the default ALS list. The remaining bits are reserved.
4 (4)	8	PU name If no name is returned, this byte is set to zeros.

Virtual route/transmission priority list information vector

There is an entry in the VR/TP list information vector for each VR/TP pair in the list. The format of the VR/TP list information vector is shown in [Table 40 on page 65](#).

Table 40. VR/TP list information vector

Dec (Hex) Offset	Size (Bytes)	Description
0 (0)	8	Adjacent Network identifier (NETID)
8 (8)	8	Class of service (CoS) name
16 (10)	2	Number of entries in the VR list
18 (12)	1	VR number (0–7)

Table 40. VR/TP list information vector (continued)

Dec (Hex) Offset	Size (Bytes)	Description
19 (13)	1	TP number (0–2)
20 (14)	1	VR number of the next VR/TP pair, if any. There are VR and TP numbers for each pair in the list. If there are no more VR/TP pairs in the list, the list ends with the TP number of the last pair.

Session authorization data vector

The parameter list for the SME routine includes a session authorization data vector address at offset X'44' for all functions except begin, XRF session switch, exit replacement, exit replaced, and end. Parameter descriptions are shown in [Table 41 on page 66](#).

Table 41. Session authorization data vector

Dec (Hex) Offset	Size (Bytes)	Description
0 (0)	1	Authorization data flags B'1...' To be set to 1 by the exit at Initial Accounting if the data is to be retained until Final Accounting. If not set by the exit, the data will be freed after the Initial Accounting call. The SMEAUTH start option can be used to override the SME. For information about this option, see z/OS Communications Server: SNA Operation . B'.111 1111' Reserved for future use.
1 (1)	1	Vector Key X'59'
2 (2)	1	Vector Data Length
3 (3)	253	Vector Data (253 byte maximum)

Session authorization data can be modified during any of the calls. The data passed to the exit will be either the data provided by an earlier exit invocation or the vector header with a data length of zero. The storage following the vector might contain residual data from prior exit invocations.

The single-byte data length and maximum 253-byte data field can be created or modified. Modified data is passed to the PLU on session setup as part of the CINIT. The data passed to the exit during Virtual Route selection and Initial Accounting can be modified, but the modifications are not passed to the PLU application or other SSCPs. Instead, they are passed during the Final Accounting call.

Once included by an SME, the vector cannot be removed before the Initial Accounting call. This is necessary to allow the code to detect a migration host in the session-setup path. If an exit needs to remove the data before Initial Accounting, it sets the length to one byte and the data byte to zero. Unless the exit takes explicit action, the data is freed automatically at the time of Initial Accounting so that hosts that do not want the data saved do not have to add code to their exits.

VTAM frees the data if the exit is not being called for accounting or if the X'80' bit in byte zero of the session authorization data is used by VTAM as a KEEP flag. To free the data, VTAM initializes the bit to zero. To save the data for Final Accounting, VTAM turns on the bit when the exit is called for Initial Accounting.

VTAM exit services

VTAM Exit Services can be called by the session management exit to display text on the system console. When called, VTAM Exit Services places the text into VTAM messages and displays the messages on the system console. The messages are displayed as a group with message IST1282I containing the exit routine name and load module name, as many IST1405I messages as necessary to display all of the message text, and one IST314I message to end the message group. If an error occurs while attempting to perform the requested function, or if an invalid function is requested, a message group consisting of messages IST1455I and IST1456I is issued to report the failure and reason. See [z/OS Communications Server: SNA Messages](#) for details about these messages.

Before the session management exit routine can invoke VTAM Exit Services, the exit must first check the function code and related session information to determine if VTAM Exit Services is available. The session management exit must also verify that the address of the VTAM Exit Services parameter list, contained within the session management exit parameter list at offset 72 (decimal), is not zero. See [“VTAM Exit Services parameter list” on page 67](#) for information about the VTAM Exit Services parameter list.

Initial register contents

When the session management exit calls VTAM Exit Services, register contents must be as follows:

Register 1:

Address of the VTAM Exit Services parameter list

Register 13:

Address of a standard 72-byte save area

Register 14:

Return address

Register 15:

Address of the VTAM Exit Services module (which can be obtained from the VTAM Exit Services parameter list)

Final register contents

When VTAM Exit Services returns control to the session management exit, register contents are as follows:

Registers 0–14:

Restored to entry contents.

Register 15:

Return code:

X'00'

VTAM Exit Services successfully performed the requested function.

X'04'

Input provided by the session management exit to VTAM Exit Services was not valid.

X'08'

A function which was not valid or supported was requested from VTAM Exit Services (for example, the function code in the VTAM Exit Services parameter list represents a function which is not valid, or is not supported by the current level of VTAM Exit Services).

X'0C'

VTAM Exit Services was unable to perform the requested function due to an internal (VTAM) error.

VTAM Exit Services parameter list

When VTAM Exit Services gets control, the address of the following parameter list must be in register 1.

Table 42. VTAM Exit Services parameter list

Dec (Hex) Offset	Size (Bytes)	Value when Calling VTAM Exit Services	Description
0 (0)	4	An address provided by VTAM	A pointer to a Supported Functions bitmap indicating which VTAM Exit Services functions are available.
4 (4)	1	X'01'	One-byte function code to indicate the requested function of VTAM Exit Services. (X'01' indicates the message function). Any other value will result in an error message on the system console and a non-zero return code being returned to the exit.
5 (5)	3	Reserved	Reserved
8 (8)	4	An address provided by VTAM	Address of the VTAM Exit Services module (ISTIECXS). ISTIECXS resides below the 16M line.
12 (C)	4	An address provided by the session management exit routine	List to be passed to VTAM Exit Services (pointer to the VTAM Exit Services Message parameter list)
16 (10)	4	Reserved	Reserved

The VTAM Exit Services parameter list contains a pointer to the Supported Functions bitmap which indicates which functions are supported by the current level of VTAM Exit Services. The exit should verify the requested function is supported before calling VTAM Exit Services. This bitmap allows an exit to be migrated to different levels of VTAM more easily. The structure of the Supported Functions bitmap is shown in [Table 43 on page 68](#):

Table 43. VTAM Exit Services supported functions bitmap

Byte	Description
0	<p>Bit Setting:</p> <p>B'1...' This is the last fullword of the VTAM Exit Services Supported Functions bitmap.</p> <p>B'0...' There is at least one more fullword of the VTAM Exit Services Supported Functions bitmap following this fullword. The last fullword of the bitmap will have the high-order bit on.</p> <p>B'.1...' The message function is available from VTAM Exit Services.</p> <p>B'.0..' The message function is not available from VTAM Exit Services.</p> <p>All remaining bits are reserved.</p>
1	Reserved
2	Reserved
3	Reserved

VTAM Exit Services message parameter list

The VTAM Exit Services parameter list contains a pointer to an input parameter list which is provided by the exit calling VTAM Exit Services. For the message function, the input parameter list is the VTAM Exit Services Message parameter list (EXMPL). The format of the VTAM Exit Services Message parameter list is in [Table 44 on page 69](#).

Note: Each time VTAM calls the session management exit routine, the VTAM Exit Services parameter list is initialized. This results in the pointer to the EXMPL being zeroed. The exit must always update the VTAM Exit Services Message parameter list pointer before calling VTAM Exit Services.

Table 44. VTAM Exit Services message parameter list

Dec (Hex) Offset	Size (Bytes)	Value when Calling VTAM Exit Services	Description
0 (0)	2	N/A	Reserved
2 (2)	2	Set by session management exit	A 16-bit signed number which represents the length of the message text. This number must be greater than 0 and less than or equal to 4096 (decimal). The maximum supported length is 4096 (decimal) bytes per invocation of VTAM Exit Services.
4 (4)	4	Set by session management exit	The address of the message text to be issued by VTAM Exit Services on behalf of the session management exit.

For information about design considerations and examples of invoking VTAM Exit Services, see [“VTAM exit services” on page 262](#).

Session management data area

This function allows an exit to pass a pointer of a model piece of storage (session management data area), with a descriptive header, back to VTAM on any session-related exit invocation. VTAM copies, maintains, and passes back the storage on each subsequent exit call for that session. Each session can have a session management data area.

The exit can change or update the VTAM copy of the storage when it is passed and, if it does, VTAM maintains the changed copy. The session management data area is freed automatically after initial accounting. Alternatively, if the exit indicates to VTAM that it wants the data area to remain allocated, the data area is held by VTAM until final accounting is completed.

The parameter list for the SME routine includes a session management data address at offset X'4C' for all functions except begin, exit replacement, exit replaced, and end. The parameter list is shown in [Table 45 on page 69](#).

Table 45. Session management data area parameter list

Dec (Hex) Offset	Size (Bytes)	Description
0 (0)	2	Total length of the variable data including this length field

Table 45. Session management data area parameter list (continued)

Dec (Hex) Offset	Size (Bytes)	Description
2 (2)	1	Exit data flags B'1...' To be set to 1 by the exit if the data is to be retained until Final Accounting. If it is not set by the exit, the data will be freed after the Initial Accounting call. B'.111 1111' Reserved for future use
3 (3)	*	Variable length data (2037 bytes maximum)

The size of the session-related storage is limited to 2040 (X'7F8') bytes. If this size is exceeded, the storage is not allocated, and a null (0) pointer is passed on subsequent exit invocations for this session.

A pointer to the exit's copy of storage can be passed to VTAM on return from any valid session-related invocation when the parameter list pointer at X'4C' is zero and when the exit is first invoked.

VTAM does not maintain a data area for CLSDST PASS (Initiate Other CD) or Directed Search List (DSRLST) processing.

If succeeding exit calls return different sizes for a particular session or if an exit returns a pointer to different storage, all VTAM storage is freed and a null pointer (0) is passed on subsequent exit invocations for the related session.

TCP/IP information control vector

This is defined by the control vector X'64' and contains the IP characteristics of a TN3270E LU. These characteristics include the IP address of the LU, the port name associated with the LU, and the name of the LU's domain name server, if any. When the SME receives this vector during initial or secondary authorization, or during initial or final accounting, the SME can use these characteristics of the TN3270 LU as part of its authentication logic. See *SNA Formats, GA273136* for more information about the TCP/IP information control vector.

Virtual route selection exit routine

A virtual route selection exit routine allows you to modify the ordered list of virtual routes (VRs) as specified in the class of service (CoS) entry. VTAM then uses your new list of virtual routes to select a route for the session.

You can write a virtual route selection routine to be invoked before VTAM establishes any sessions between logical units and before VTAM is terminated (to enable it to clean up any necessary resources).

VTAM calls the virtual route selection exit routine when a session between a primary logical unit in the VTAM subarea and a logical unit in another subarea is about to be established. The virtual route selection exit routine is not called if both logical units reside in the same VTAM subarea. The exit is not called for independent PLUs. VTAM to reattach the subtask, thereby regaining use of the exit.

Only the list of routes within the local network is passed to the exit routine. The exit routine is scheduled only within the subarea of the primary logical unit.

The virtual route exit routine is not scheduled when:

- An SSCP session is established.

- An LU-LU session is established that is confined to the host's subarea. For example, two application programs communicating within the same host, or an application program communicating with a channel-attached terminal.
- A session setup is attempted, and no defined routes that are operational exist between the origin and destination subareas. In this case, the request to establish a session is rejected before the exit would have been scheduled.

You can use the DISPLAY EXIT command to display information regarding a virtual route selection exit routine. You also can use a MODIFY EXIT command to activate, deactivate, or replace the virtual route selection exit routine without interrupting VTAM processing. See [“Operator commands for VTAM exit routines” on page 174](#) for more information about using the MODIFY EXIT command to modify VTAM exit routines. See [z/OS Communications Server: SNA Operation](#) for more information about these commands.

The topics that follow contain information you need to write a virtual route selection exit routine.

Initial register contents

When VTAM passes control to the virtual route selection exit routine, register contents are as follows:

Register 1:

Address of the parameter list described in [Table 46 on page 72](#)

Register 13:

Address of a standard 72-byte save area

Register 14:

Return address

Register 15:

Address of the entry point of this routine

Final register contents

The routine must leave the register status as follows:

Registers 0–14:

Restored to entry contents

Register 15:

Return Code:

X'00' (0)

Processing successfully completed.

X'01'–X'4F' (1–79)

Exit processing did not work; the VR selection exit remains active.

X'50'–X'7F' (80–127)

Reserved

Greater than X'7F' (128)

Invocation failed; VR selection exit deleted and inactive

X'F0'–X'FF' (240–255)

Reserved

Design requirements

Follow these procedures when writing this routine:

- Use standard linkage.
- Save registers 0–14.

Consider the following restrictions when writing this exit routine:

- The name of the virtual route selection exit routine module should be ISTEXCVR; however, if you have written an alternate load module, use the load module name you assigned to your replacement module.

- Do not issue any SVCs if this exit routine is running in SRB mode.
- This exit routine should use only conditional storage invocations. You can reduce the possibility of a VTAM abend during a storage shortage by coding conditional storage invocations.
- Data is addressable in 24- or 31-bit mode. It is advisable that you use 31-bit addressing.
- This exit routine can be above or below the 16M line. Data is always presented below the line.
- This exit routine must be reentrant.
- This exit routine must be link-edited into the appropriate library. See [“Installing VTAM exit routines”](#) on page 173 for more information.
- This exit routine runs under a VTAM subtask that permits the exit routine to perform any necessary I/O or other processing without affecting the VTAM main task.
- This exit routine executes in problem state with a VTAM storage key.

The exit routine can set up its own ESTAE (extended specify task abnormal exit) environment for recovery. See z/OS application development publications for more information. A SYNC interface will be used between VTAM and the exit routine, so that the VTAM module invoking the exit routine will be isolated from any ESTAE environment established by the exit routine.

Virtual route selection exit routine parameter list

When VTAM passes control to the virtual route selection routine, the address of the following parameter list is in register 1.

Table 46. Virtual route selection exit routine parameter list

Dec (Hex) Offset	Size (Bytes)	Description
0 (0)	4	Address of a 1-byte reason code indicating why the exit routine was scheduled: Reason Code X'00' To establish an LU-LU session (the first time) X'01' To establish an LU-LU session (not the first time) X'02' To establish an LU-LU session (after an abend to the exit subtask) X'03' Because VTAM is terminating due to HALT (last time) or because an operator deactivated the exit X'04' Because VTAM is terminating X'05' Because VTAM is driving the exit the last time before the exit is replaced. VTAM also passes the address of the 8-byte user field to the exit. X'06' Because VTAM is driving the exit the first time after the exit is replaced. VTAM also passes the address of the 8-byte user field to the exit.

Table 46. Virtual route selection exit routine parameter list (continued)

Dec (Hex) Offset	Size (Bytes)	Description
4 (4)	4	<p>Address of an 8-byte user field</p> <p>This field is initially set to 0. VTAM preserves the contents of this field and resupplies it to the exit routine the next time it is driven. For example, the exit routine can use this field to store the address of a dynamically obtained storage area.</p> <p>The last 4 bytes can be used to uniquely identify an exit's level or version. The last 4 bytes will be displayed following subsequent use of the DISPLAY EXIT command (see z/OS Communications Server: SNA Operation for information about this command).</p> <p>If X'05' is the first function driven, this field points to zeros.</p>
8 (8)	4	<p>Address of a 1-byte session information field:</p> <p>Bit B'1...'</p> <p>Pending session requires a VR that maps to an explicit route with an ER number of 0 (originating in the SLU subarea and terminating in the PLU subarea)</p> <p>The remaining bits are reserved.</p>
12 (C)	4	<p>Address of an 8-byte field containing the CoS name associated with this pending session establishment request</p> <p>This field is padded on the right with blanks as necessary to make a total of 8 characters.</p>
16 (10)	4	<p>Address of a 12-byte field containing the origin subarea information and LU name</p> <p>This field contains a 4-byte binary subarea number, followed by an 8-byte name in EBCDIC.</p>
20 (14)	4	<p>Address of a 12-byte field containing the destination subarea information and LU name</p> <p>This field contains a 4-byte binary subarea number, followed by an 8-byte name in EBCDIC.</p>
24 (18)	4	<p>This word and all subsequent words will each contain the address of a virtual route descriptor block. The last word of the parameter list is indicated by the setting of the high-order bit (byte 0, bit 0) to 1.</p>

Abend processing

If scheduled as a result of an abend of the subtask under which the exit routine executes, the exit routine is responsible for any cleanup or reopening of data sets that might be required.

If a dynamic storage address was stored in the user field (second word of the parameter list described in [Table 46 on page 72](#)), the address is not valid after the abend is processed.

An abend indication can occur either for a normal route selection request or for the last time the exit routine is driven during VTAM termination. Also, if the subtask under which the exit routine runs suffers an abend during *first time* route selection (reason code X'00'), and the exit routine has not yet returned to VTAM, the next time the exit routine is driven will still be considered the first time. This is not true if

the next time the exit routine is driven is for VTAM termination after an abend (reason code X'04'). In any case, modifications made by the exit routine to the user field before an abend are preserved for use by the exit routine in any cleanup or recovery actions it might perform the next time the exit is driven.

To summarize the implications of the reason code specifications:

X'00'

Driven for route selection (first time)

X'01'

Driven for route selection (nth time, normal)

X'02'

Driven for route selection (previous time resulted in abend)

X'03'

Driven for VTAM termination (last time) or because an operator deactivated the exit

X'04'

Driven for VTAM termination (after abend)

X'05'

Driven for exit replacement (last time before replacement)

X'06'

Driven after exit replacement (first time after replacement)

If the exit routine subtask abends, VTAM reattaches the subtask.

Changing the virtual route selection list

The parameter list that VTAM sends to the virtual route selection exit routine includes pointers to a list of data blocks called virtual route descriptor blocks. Each virtual route descriptor block contains the virtual route number and transmission priority for each virtual route that is defined and operative between the origin and destination subareas and within the class of service requested for the session. The format of a block is described in [Table 47 on page 74](#).

Table 47. Virtual route descriptor block format

Dec (Hex) Offset	Size (Bytes)	Description
0 (0)	1	VR number (0–7)
1 (1)	1	Transmission priority (0–2)
2 (2)	1	VR status X'01' — VR not active X'02' — VR is active
3 (3)	1	Reserved
4 (4)	2	Current ¹ number of LU-LU sessions between the origin and destination subareas using the given VR number and transmission priority
6 (6)	2	Current ¹ number of LU-LU sessions between the origin and destination subareas using the given VR number regardless of priority
8 (8)	2	Current ¹ number of all sessions between the origin and destination subareas using the given VR number and transmission priority
10 (A)	2	Current ¹ number of all sessions between the origin and destination subareas using the given VR number regardless of priority

Table 47. Virtual route descriptor block format (continued)

Dec (Hex) Offset	Size (Bytes)	Description
---------------------	-----------------	-------------

Note:

1. Current as of the queuing of the virtual route selection request to the virtual route selection exit routine subtask.

The exit routine can modify VTAM's route selection process by changing the original list of descriptor blocks it receives in the following manner:

- Reordering the list
- Deleting entries from the list
- Using only a portion of the list
- Replacing a virtual route in the list

Examples of virtual route selection exit routine processing

You cannot return more virtual route descriptor blocks than presented in the list on input. The list can be shortened, but not lengthened. The following examples illustrate each type of processing the exit routine can perform. In the examples, (*)Address means that the high-order bit is set to indicate the last entry in the list.

Example 1: Reordering the list

1. The virtual route selection exit routine receives:
 - Address of VR1,TP1 descriptor block
 - Address of VR2,TP2 descriptor block
 - Address of VR3,TP1 descriptor block
 - (*)Address of VR4,TP0 descriptor block
2. The virtual route exit routine reorders the list and returns to VTAM:
 - Address of VR3,TP1 descriptor block
 - Address of VR4,TP0 descriptor block
 - Address of VR1,TP1 descriptor block
 - (*)Address of VR2,TP2 descriptor block
3. VTAM attempts to assign the pending session to VR3, then VR4, then VR1, then VR2.

Example 2: Deleting entries from the list

1. The virtual route selection exit routine receives:
 - Address of VR1,TP2 descriptor block
 - Address of VR2,TP0 descriptor block
 - Address of VR3,TP2 descriptor block
 - (*)Address of VR4,TP1 descriptor block
2. The virtual route selection exit routine deletes VR2,TP0 from the list and returns to VTAM:
 - Address of VR1,TP2 descriptor block
 - 00 00 00 00 (4 bytes of zeros)
 - Address of VR3,TP2 descriptor block
 - (*)Address of VR4,TP1 descriptor block

3. VTAM attempts to use VR3,TP2 if VR1,TP2 cannot be activated.

Example 3: Using only a portion of the list

1. The virtual route selection exit routine receives:
 - Address of VR1,TP1 descriptor block
 - Address of VR2,TP1 descriptor block
 - Address of VR3,TP0 descriptor block
 - (*)Address of VR4,TP2 descriptor block
2. The virtual route selection exit routine turns on the high-order address bit in VR2,TP1 and returns to VTAM:
 - Address of VR1,TP1 descriptor block
 - (*)Address of VR2,TP1 descriptor block
 - Address of VR3,TP0 descriptor block
 - Address of VR4,TP2 descriptor block
3. VTAM uses only VR1,TP1 and then VR2,TP1 to establish the pending session. If neither route can be activated, the session-establishment request is terminated because no paths are available.

Example 4: Replacing a virtual route in the list

1. The virtual route selection exit routine receives:

Addresses of descriptor blocks	Descriptor block contents
VR1,TP2 descriptor block	VR1,TP2 . . .
VR2,TP1 descriptor block	VR2,TP1 . . .
VR3,TP0 descriptor block	VR3,TP0 . . .
(*)VR4,TP0 descriptor block	VR4,TP0 . . .

2. To replace the first entry in the virtual route list, overwrite the contents of the appropriate descriptor block:

Addresses of descriptor blocks	Descriptor block contents
VR1,TP2 descriptor block (This is still the original address; only the contents of the block change, not the address of the block.)	VR6,TP1 . . .
VR2,TP1 descriptor block	VR2,TP1 . . .
VR3,TP0 descriptor block	VR3,TP0 . . .
(*)VR4,TP0 descriptor block	VR4,TP0 . . .

3. After the replacement, the first address in the list points to the same descriptor block. However, the contents of the descriptor block have been overwritten. The new contents describe VR6,TP1. VTAM attempts to use VR6,TP1 before using the other virtual routes in the list.
4. If the virtual route is not valid (defined as being outside the range 0–7 or as having a transmission priority outside the range 0–2) or is not defined between the origin and destination subareas, VTAM continues through the set of virtual routes, attempting to activate the next one in the set (VR2,TP1).

Virtual route pacing window size calculation exit routine

The virtual route (VR) pacing window size calculation exit routine allows you to specify the bounds for virtual route pacing windows. A virtual route pacing window represents the quantity of path information units (PIUs) that can be transmitted on a virtual route before a virtual route pacing response is received. The virtual route pacing response indicates that the virtual route receiver is ready to receive more PIUs on the route. The VR pacing window size calculation exit routine is called when a virtual route is activated. It returns the minimum and maximum values for the window of the virtual route.

The IBM-supplied algorithm for window size calculation works with the route pacing algorithm used in the network. It is appropriate for most installations and configurations. VTAM calculates the minimum and maximum sizes of virtual route pacing windows based on the link protocol and the explicit route length (that is, the number of transmission groups in the explicit route used by the virtual route). While the virtual route is being used to transmit data, subarea nodes along the route request that the end points adjust the window sizes within the minimum and maximum limits according to traffic conditions along the route.

However, after tuning VTAM and analyzing traffic patterns and resource capabilities, you might want to choose your own bounds or code a replacement routine to set the window sizes to different values than those supplied by IBM. The VR pacing window size calculation exit routine is appropriate for systems where the number of resources could vary considerably from one day to the next. For example, you might find one or more resources whose capacities are not consistently used. This might warrant increasing the window sizes. Decreasing the window sizes is less likely to be useful, because network-flow control protocols are designed to prevent congestion, and setting window sizes too small could reduce traffic flow considerably.

The logic for calculating the maximum and minimum window sizes for virtual route pacing is contained in the VTAM exit routine named ISTPUCWC. The source code for ISTPUCWC is not provided with the VTAM program. It is available through the View Program Listings (VPL) application from ISM&D (IBM Software Manufacturing and Distribution), Boulder, Colorado. VPL replaces microfiche and provides the required logic online. You can access this information by using ServiceLink or Dial IBM.

Note: You can avoid coding the VR pacing window size calculation exit routine by specifying default minimum and maximum window sizes using the VRPWSnn operand on the PATH definition statement. For more information about specifying virtual route pacing window sizes on the PATH definition statement, see [z/OS Communications Server: SNA Resource Definition Reference](#).

For most routes, VTAM sets the minimum window size to the explicit route length and the maximum size to three times the explicit route length. However, if the virtual route ends in a subarea that is adjacent to VTAM, the maximum window size is set to the greater of 15 or $255-16n$, where n is the number of explicit routes (defined or operative) originating in the host and passing through and not ending in the adjacent subarea. This increases the maximum window size for a route to a channel-attached NCP that has only a few explicit routes passing through it.

The VR pacing window size calculation exit routine is called when VTAM is preparing to activate a virtual route. VTAM passes the following information to the exit:

- The particular virtual route number and its transmission priority
- The number and length of the explicit route that is associated with the particular virtual route
- The subarea addresses of the destination and adjacent subareas
- The number of defined and operative explicit routes that traverse, but do not end in, the adjacent subarea
- The address of the explicit route characteristics table, which describes the transmission group traversed by the explicit route associated with a virtual route. This table determines the link protocol being used between the host and the adjacent node on the explicit route. (See [Table 49 on page 79](#) for the format of this table.)

The following topics contain information you need to replace or modify the VR pacing window size calculation exit routine.

Initial register contents

When VTAM passes control to the VR pacing window size calculation exit routine, register contents are as follows:

Register 1:

Address of the parameter list described in [Table 48 on page 79](#)

Register 13:

Address of a standard 72-byte save area

Register 14:

Return address

Register 15:

Address of the entry point of this routine

Final register contents

The routine must leave the register status as follows:

Register 0:

Minimum VR window size (if register 15=0)

Note: This value must be greater than zero and less than or equal to the maximum value specified in register 1.

Register 1:

Maximum VR window size (if register 15=0)

Note: This value must be greater than or equal to the minimum value specified in register 0 and less than or equal to 255.

Registers 2–14:

Restored to entry contents

Register 15:

Return code of 0 if the minimum and maximum window size values are to be used by VTAM. Any nonzero return code (or values that are not valid specified in register 0 or 1) causes VTAM to ignore the values indicated by registers 0 and 1. Instead, it uses a minimum window size equal to the ER length and a maximum window size of three times the ER length, or the minimum and maximum window sizes that you defined on the PATH definition statement, if any.

For more information about specifying virtual route pacing window sizes on the PATH definition statement, see [z/OS Communications Server: SNA Resource Definition Reference](#).

Design requirements

Follow these procedures when writing this routine:

- Use standard linkage.
- Save and restore registers 2–14.

Consider the following restrictions when writing this routine:

- The name of the virtual route pacing window size calculation exit routine module must be ISTPUCWC.
- Do not issue any SVCs if this exit routine is running in SRB mode.
- This exit routine should use only conditional storage invocations. You can reduce the possibility of a VTAM abend during a storage shortage by coding conditional storage invocations.
- All data is addressable in 24-bit mode only.
- This exit routine must be link-edited into the appropriate library. See [“Installing VTAM exit routines” on page 173](#) for more information.

- This exit routine operates as an internal VTAM subroutine. VTAM performance might be degraded if the routine requires lengthy processing time. While this routine has control, VTAM does not process any new operator requests, session initiation requests, or session termination requests for any resource.
- System waits, including implied waits for I/O operations, should be avoided. System waits can cause VTAM failure in some timing-dependent situations.
- This exit routine operates enabled in pageable storage. The routine gains control in supervisor state with a VTAM storage key. Errors within the routine could damage VTAM or system control blocks and modules.

VR pacing window size calculation exit routine parameter list

When the exit gets control, the address of the following parameter list is in register 1.

Table 48. VR pacing window size calculation exit routine parameter list

Dec (Hex) offset	Size (Bytes)	Description
0 (0)	4	VR identifier: <ul style="list-style-type: none"> • Byte 0 = X'00' • Byte 1 = X'00' • Byte 2 = VR number • Byte 3 = Transmission priority
4 (4)	4	Explicit route number
8 (8)	4	Destination subarea number
12 (C)	4	Adjacent subarea number
16 (10)	4	Explicit route length. (This length is equal to the total number of transmission groups in the explicit route.)
20 (14)	4	Total number of defined or operative explicit routes that traverse, but do not end in, the given adjacent subarea
24 (18)	4	Address of explicit route characteristics table

Explicit route characteristics table

VTAM passes the address of this table to the exit in the VR pacing window size calculation exit routine parameter list. This table describes the transmission group traversed by the explicit route associated with a virtual route. This table determines the link protocol being used between the host and the adjacent node on the explicit route.

Table 49. Explicit route characteristics table

Dec (Hex) offset	Size (Bytes)	Description
0 (0)	1	Control block identifier (X'4C')
1 (1)	1	Reserved
2 (2)	2	Length of this table

The transmission group entry consists of the following data:

0 (0)	1	Control block identifier (X'4D')
1 (1)	1	Length of entry

Table 49. Explicit route characteristics table (continued)

Dec (Hex) offset	Size (Bytes)	Description
2 (2)	1	Not used
3 (3)	1	DLC protocol: X'01' SDLC X'02' Channel-to-communication controller X'03' Channel-to-channel X'04' LAN attachment

Using the VR window size calculation exit routine for IMS

Because of the specialized use of channel-to-channel virtual routes by the IMS resource lock manager (RLM), you might find it useful to modify the VTAM virtual route window size calculation exit routine, ISTEPUCWC. The RLM in one IMS system sends one message at a time to a correspondent IMS RLM, and waits for a message in response. With this level of message traffic on a virtual route, the virtual route window size algorithm does not indicate that a demand exists to increase the window size. Thus, in the absence of other message traffic flowing on the virtual route, the window size tends to stay at its minimum value. Although this situation does not impede throughput on the virtual route, it tends to increase the number of machine instructions executed by VTAM for each IMS transaction (because VTAM processes a virtual route pacing response for every, or nearly every, IMS RLM message).

Accordingly, if a channel-to-channel virtual route between two IMS systems is not also used for non-IMS message traffic, you can make the following changes to the module to reduce the number of virtual route pacing responses sent and processed by VTAM in each of the IMS hosts:

- Insert code to recognize the destination subarea address, virtual route number, and transmission priority that you have defined for use between the two IMS systems.
- Insert code so that when the module recognizes such a virtual route, it sets the minimum window size to a value of 6.

The maximum window size can be computed in the normal manner, but the output of that calculation must be checked against this new minimum and must not be allowed to fall below it. That is, if the maximum window size is calculated to be less than 6, the maximum should then be set to 6.

Note: You can avoid coding the VR pacing window size calculation exit routine for IMS by allocating IMS-to-IMS communication to a specific virtual route and transmission priority with a unique class of service. This enables you to use the VRPWSnn operand on the PATH definition statement to control pacing window size.

There are two possible ways to compute the maximum window size in the IBM-supplied version of the module. One is used when the destination subarea is the same as the adjacent subarea, and the other is used in all other cases. Both parts of this logic should be copied into any new section of code added to compute window sizes for IMS RLM routes. Again, the maximum window size values computed must be adjusted, if necessary, to be at least equal to the minimum window size selected.

These changes to the module are not advisable if the IMS RLM virtual route is shared with other virtual route users. The message traffic of the other users should keep the operating window size values at acceptable levels.

Session accounting exit routine

The session accounting exit routine allows you to collect statistics on the number of times that sessions start and end so that users can be charged accordingly.

Whenever possible, you should use a session management exit routine rather than an accounting exit routine to gather accounting information, because the session management routine allows you to combine session-related functions into one exit routine. The session management exit routine also fully supports cross-network and takeover processing. For example, information passed to the accounting exit routine for cross-network sessions might be misunderstood because LU names are not necessarily unique. The session management exit routine's accounting function avoids this problem because additional LU information is provided to the exit in the PLU and the SLU resource information control vectors. (See [“Session management exit routine” on page 1](#) for more information about its capabilities.)

The session accounting exit routine is scheduled only in SSCPs that are in the same domain as one of the LUs. If both the accounting and session management exit routines are provided within the LU's domain, VTAM schedules both exits.

IBM does not supply a skeleton session accounting exit routine. Write a session accounting exit routine if you want this function. If you do not write this routine, all session accounting information intended for the routine is discarded.

The following topics contain information you need to write a session accounting exit routine.

Initial register contents

When VTAM passes control to the session accounting exit routine, register contents are as follows:

Register 0:

X'00000000' if a session has been established; X'FFFFFFFF' if a session has been terminated

Register 7:

Address of the doubleword containing the name of the PLU

Register 11:

Address of the doubleword containing the name of the SLU

Register 13:

Address of a standard 72-byte save area

Register 14:

Return address

Register 15:

Address of the entry point of this routine

Final register contents

All general-purpose registers, except register 15, must be restored to entry contents. VTAM does not require a return code.

Design requirements

Follow these procedures when writing this routine:

- Use standard linkage.
- Save registers 0–14.

Consider the following restrictions when writing this routine:

- The name of the session accounting exit routine module must be ISTAUCAG.

Note: The user must re-IPL when replacing or modifying his own accounting exit routine.

- Do not issue any SVCs if this exit routine is running in SRB mode.

- This exit routine should use only conditional storage invocations. You can reduce the possibility of a VTAM abend during a storage shortage by coding conditional storage invocations.
- All data is addressable in 24-bit mode only.
- This exit routine must be link-edited into the appropriate library. See [“Installing VTAM exit routines” on page 173](#) for more information.
- This exit routine operates as an internal VTAM subroutine. VTAM performance might be degraded if the routine requires lengthy processing time. While this routine has control, VTAM does not process any new operator requests, session initiation requests, or session termination requests for any resource.
- System waits, including implied waits for I/O operations, should be avoided. System waits can cause VTAM failure in some timing-dependent situations.
- This exit routine should not contain VTAM macroinstructions.
- This exit routine operates enabled in pageable storage. Because the routine operates at VTAM's main task dispatching priority, there is a possibility of lockout if a wait requires another task action. The routine gets control in supervisor state with a VTAM storage key. Errors within the routine could damage VTAM or system control blocks and modules.
- This exit routine provides the two LU names involved in the session. You can write this information to the System Management Facility (or an equivalent facility), along with the time of day. You can use these records to determine the session connection time. This exit does not have access to the logon data, VTAM control blocks, or TSO control blocks.
- This exit routine is notified as part of an LU-LU session setup and takedown. You should design the routine to process requests involving only LU-LU sessions.
- If an application program can establish parallel sessions, this exit routine must be capable of processing more than one request from the same LU-LU pair.

Session authorization exit routine

The session authorization exit routine allows you to check or restrict the use of an application program or other LU.

Whenever possible, you should use a session management exit routine rather than a session authorization exit routine to authorize sessions, because a session management routine handles both same- and cross-network sessions and allows you to combine session-related functions into one exit routine. (See [“Session management exit routine” on page 1](#) for more information about its capabilities.)

The session authorization exit routine is scheduled only in SSCPs that are in the same domain as one of the LUs. For cross-network sessions, this means that any information provided to the session authorization routine could be misunderstood, because an LU name is not necessarily unique. You can avoid this problem by using the initial and secondary authorization functions of the session management exit routine instead of the session authorization exit routine.

If both the session authorization and session management exit routines exist within the LU's domain, VTAM calls the authorization exit routine first. If the session authorization exit routine authorizes the session, VTAM calls the session management exit routine; otherwise, VTAM does not call the session management exit routine and session setup is rejected.

For same-network sessions, VTAM calls the session authorization exit routine whenever it receives a request to establish a session between two LUs. For example, VTAM can call the exit as the result of a logon from a terminal, an automatic logon, or a VTAM operator logon. VTAM calls the exit for both the initial logon and all subsequent logons to a controlling application program (that is, one to which the terminal is logged on automatically).

For cross-domain sessions, VTAM calls the session authorization exit routine in each domain (the domain of the PLU and of the SLU) whenever it receives an initiate or cross-domain initiate request.

Failure of the session authorization exit routine to honor session-initiation requests can cause VTAM to authorize sessions that should not be authorized.

You can code the session authorization routine to contain a table of valid sessions against which the session-establishment request can be compared. For example, you can design an application program to establish a session with any LU, using the OPNDST OPTCD=ACCEPT macroinstruction in the application program's LOGON exit routine. The session authorization exit routine can compare the identity of any LU that attempts to establish a session with the application program to entries in such a table to determine whether authorization can be granted for that LU. For example, a particular LU could be authorized only at a particular time of day.

If an application program can establish parallel sessions, the exit routine must be capable of processing more than one request from the same LU-LU pair.

IBM does not supply a skeleton session authorization routine. If you want this function, write a session authorization exit routine. If you do not write a session authorization exit routine, all sessions are authorized (unless you have written a session management exit routine to authorize sessions).

Note: The user must re-IPL when replacing or modifying his own exit routine.

The topics that follow contain information you need to write a session authorization exit routine.

Initial register contents

When VTAM passes control to the session authorization exit routine, register contents are as follows:

Register 1:

Address of the parameter list described in [Table 50 on page 84](#)

Register 13:

Address of a standard 72-byte save area

Register 14:

Return address

Register 15:

Address of the entry point of this routine

Final register contents

The routine must leave the register status as follows:

Registers 0–14:

Restored to entry contents

Register 15:

Return code of 0 if the request is authorized. Any nonzero return code if the request is not authorized.

Design requirements

Follow these procedures when writing this routine:

- Use standard linkage.
- Save registers 0–14.

Consider the following restrictions when writing this routine:

- The name of the session authorization exit routine module must be ISTAUCAT.
- Do not issue any SVCs if this exit routine is running in SRB mode.
- This exit routine should use only conditional storage invocations. You can reduce the possibility of a VTAM abend during a storage shortage by coding conditional storage invocations.
- All data is addressable in 24-bit mode only.
- This exit routine must be link-edited into the appropriate library. See [“Installing VTAM exit routines” on page 173](#) for more information.

- The routine operates as an internal VTAM subroutine. VTAM performance might be degraded if the routine requires lengthy processing time. While this routine has control, VTAM does not process any new operator requests, session initiation requests, or session termination requests for any resource.
- System waits, including implied waits for I/O operations, should be avoided. System waits can cause VTAM failure in some timing-dependent situations.
- This routine operates enabled in pageable storage. Because the routine operates at VTAM's main task dispatching priority, there is a possibility of lockout if a wait requires other task action. The routine gains control in supervisor state with a VTAM storage key. Errors within the routine could cause damage to VTAM or system control blocks and modules.
- This exit routine should not contain VTAM macroinstructions.
- This exit routine must not modify the parameter list pointed to by register 1 (described in [Table 50 on page 84](#)) or any field pointed to from the parameter list.
- This exit routine must supply a return code to VTAM in register 15. A return code of 0 authorizes the session to be established. Any nonzero return code means that the request is not authorized. If the request is not authorized, VTAM informs the session initiator.

Session authorization exit routine parameter list

When the exit gets control, the address of the following parameter list is in register 1.

Table 50. Session authorization exit routine parameter list

Dec (Hex) offset	Size (Bytes)	Description
0 (0)	1	Reserved
1 (1)	1	Contains a decimal number 1–6 that identifies the request: No. Request 1 Primary LU initiated session (SIMLOGON or OPNDST macroinstruction with ACQUIRE option) 2 Reserved 3 Reserved 4 Operator initiated session (Logon request initiated by the network operator with a VARY ACT command or VARY LOGON command) 5 Third party initiated session (CLSDST macroinstruction with PASS option). (See offset 8 for pointer to name of application program for which logon is requested.) 6 Secondary LU initiated session (REQSESS macroinstruction)
2 (2)	2	Reserved
4 (4)	4	0 or address of doubleword with name of the LU that issued the request. (Not applicable if offset 1 is 4 or 5.)
8 (8)	4	0 or address of doubleword with name of the application program acting as the primary end of the session

Table 50. Session authorization exit routine parameter list (continued)

Dec (Hex) offset	Size (Bytes)	Description
12 (C)	4	Identifies the SLU associated with request. This field contains the address of the doubleword containing the LU name.
16 (10)	2	X'0001'

Note: The macroinstructions and options referred to in this table are described in [z/OS Communications Server: SNA Programming](#). The commands are described in [z/OS Communications Server: SNA Operation](#).

Configuration services XID exit routine

This exit routine provides a way for you to give VTAM information to create dynamic representations of switched devices without disrupting your switched network. You can also use the exit to specify to VTAM the name of the dynamic switched major node under which the dynamic devices are to be placed. Because the exit allows dynamic definition capability, you do not have to explicitly define a switched device to VTAM before the device attempts to dial in.

The configuration services XID exit routine provides several advantages:

- Provides a way for you easily maintain large switched networks
- Reduces network definition
- Reduces the amount of storage required for switched resource definitions
- Allows you to group dynamic switched devices under the dynamic switched major node name you specify

You can also use the exit to record connection and disconnection times of switched devices including subarea connection (with VTAM as PU type 5 and NCP as PU type 4).

This exit can also be called for switched devices that are already defined to VTAM. If the exit is called for a known device, you can code the exit to indicate to VTAM that it is to either process or deny requests for contact (REQCONT, or REQACTPU for devices supported by DLUS) from the known switched device.

Before you decide to use a configuration services XID exit routine for dynamic definition, keep in mind that there might be some devices you do not want to define dynamically. For example, if connection time for a particular PU is extremely important, you might want to define the PU in a switched major node. Likewise, you might not want to define dynamically a switched device that makes many connections in a short time.

An important difference between devices defined dynamically and devices defined statically is that when you deactivate a PU defined dynamically, both the device and its subnodes are deleted from VTAM. When a device is deleted from VTAM, you cannot reactivate that device with operator commands. There is one exception—you can deactivate a dependent LU defined dynamically and later activate that same LU with operator commands so long as its PU stays active. You cannot use this exit to define independent LUs dynamically.

VTAM supplies a functional configuration services XID exit routine named ISTEXCCS. The source for this sample exit routine is stored in SYS1.SAMPLIB. The exit runs under a subtask of VTAM. This allows VTAM's main task to continue running while the exit is running. See [Appendix D, "Sample configuration services XID exit routine,"](#) on page 267 for the ISTEXCCS source code.

You can use the sample or write your own configuration services XID exit routine. Whether you use the sample or write your own routine, assemble and link-edit the routine into the appropriate VTAM library.

If the exit routine is link-edited into the VTAM library, VTAM loads and starts the routine during VTAM initialization and calls the exit for begin processing. After initialization, if the exit is coded to support dynamic builds, VTAM calls the exit whenever VTAM receives any of the following:

- A REQCONT from a switched PU type 1, 2, or 2.1 device
- A REQCONT from a casually connected PU type 4 or 5 device that is dialing in and is not defined to VTAM
- A REQACTPU from a switched PU type 2 or 2.1 that is a DLUS-supported resource and is not defined to VTAM

VTAM sequentially queues REQCONTs or REQACTPUs to the exit. When the exit receives a REQCONT or a REQACTPU, the exit has the opportunity to define the resource dynamically.

If the exit is coded such that it is called even though the device is already defined, VTAM calls the exit whenever VTAM receives one of the following:

- A REQCONT from a switched PU type 1, 2, 2.1 device
- A REQCONT from a casually connected PU type 4 or 5 device that is dialing in and is recognized by VTAM
- A REQACTPU from a switched PU type 2 or 2.1 device
- A REQACTPU from a DLUS-supported resource that is recognized by VTAM

VTAM sequentially queues REQCONTs or REQACTPUs to the exit. When the exit receives a REQCONT or a REQACTPU, the exit has the opportunity to indicate to VTAM that it is to process or deny the REQCONT.

If the exit is coded to support connection status reporting, VTAM calls the exit whenever the connection status of a switched PU changes.

If the configuration services XID exit routine abends while processing, VTAM purges all outstanding REQCONTs or REQACTPUs and deactivates the exit; you can use the MODIFY EXIT,OPT=ACT command to reactivate the exit.

You can use the DISPLAY EXIT command to display information regarding a configuration services XID exit. You also can use a MODIFY EXIT command to activate, deactivate, or replace the configuration services XID exit routine without interrupting VTAM processing. See [z/OS Communications Server: SNA Operation](#) for more information about these commands.

If you use the MODIFY EXIT,OPT=REPL command to replace your current exit, the level of support specified in the begin vector of the replacement exit must be the same as the level of support specified in the begin vector of the current exit. If you want a different level of support in the replacement exit, deactivate the current exit (MODIFY EXIT,OPT=INACT) and activate the replacement exit (MODIFY EXIT,OPT=ACT).

If you need to trace information relative to configuration services XID exit routine activity, use the MODIFY TRACE and MODIFY NOTRACE commands. These commands allow you to trace the configuration services XID exit's input and output. You can specify which exit functions to trace on the OPTIONS operand of the MODIFY TRACE and MODIFY NOTRACE commands. To display the trace, use the DISPLAY TRACES command.

Note: Because of the amount of generated data, the tracing of all exit functions might negatively affect performance. For this reason, use extensive tracing selectively.

For information about trace output, see [z/OS Communications Server: SNA Diagnosis Vol 1, Techniques and Procedures](#). For information about the MODIFY TRACE command, the MODIFY NOTRACE command, and the DISPLAY TRACES command, see [z/OS Communications Server: SNA Operation](#).

The following topics contain information that will help you use the sample configuration services XID exit. They also contain information you need to write your own configuration services XID exit routine. You might also want to see [Appendix D, "Sample configuration services XID exit routine," on page 267](#).

Initial register contents

When the configuration services XID exit routine gains control, register contents are as follows:

Register 1:

Address of the parameter list described in [Table 51 on page 88](#)

Register 13:

Address of a standard 72-byte save area

Register 14:

Return address

Register 15:

Address of the entry point of this routine

Final register contents

The routine must leave the register status as follows:

Register 1:

Address of the parameter list described in [Table 52 on page 89](#) for begin, XID, and build vectors

0 for connection status, failure, and end vectors

Registers 2–14:

Restored to entry contents

Register 15:

Return code

X'00' (0)

Processing successfully completed

X'01'–X'4F' (1–79)

Definition selection failed; the XID exit remains active. VTAM issues message IST1183I displaying the return code.

X'50'–X'7F' (80–127)

Reserved

X'80'–X'EF' (128–239)

Definition Selection failed; the XID exit is deactivated. VTAM issues message IST1183I displaying the return code.

X'F0'–X'FF' (240–255)

Reserved

Note: If the exit returns a nonzero return code from an XID request, VTAM sends no more XIDs to the exit. If the exit returns a nonzero return code from a connection status request, VTAM sends no more connection status requests to the exit.

Design requirements

Follow these procedures when writing this routine:

- Use standard linkage.
- Save registers 2–14.

When designing this routine, consider the following information:

- If the exit returns a dynamic switched major node name:
 - The name must be 1–8 characters.
 - The first 3 characters must not be IST.
 - The first character must be alphabetic or national, and the remaining characters must be alphabetic, numeric, or national.
- If the exit returns the name of a major node that already exists:
 - The major node cannot be an application major node.
 - The major node must be a dynamic switched major node.

- The major node must be active.
- If the exit does not return a major node name, ISTDWMMN is returned as the default name.
- The name of the configuration services XID exit routine module should be ISTEXCCS; however, if you have written an alternate load module, use the load module name you assigned to your replacement module.
- Do not issue any SVCs if this exit routine is running in SRB mode.
- This exit routine should use only conditional storage invocations. You can reduce the possibility of a VTAM abend during a storage shortage by coding conditional storage invocations. Data is addressable in 24- or 31-bit mode. It is advisable that you use 31-bit addressing.
- Exit routines that perform I/O must be link-edited AMODE=31 and RMODE=24.
- This exit routine must be link-edited into the appropriate library. See [“Installing VTAM exit routines”](#) on page 173 for more information.
- This exit routine runs under a VTAM subtask that permits the exit to perform any necessary I/O or other processing without affecting the VTAM main task.
- Do not use VTAM macroinstructions in this routine.
- This exit routine operates enabled in pageable storage. The routine gains control in supervisor state with a VTAM storage key. Errors in the routine could damage VTAM or system control blocks or modules.

Configuration services XID exit routine parameter list

When the exit gets control, the address of the following parameter list is in register 1.

Table 51. Configuration services XID exit routine parameter list

Dec (Hex) offset	Size (Bytes)	Description
0 (0)	4	Address of the vector length field This field points to a 2-byte field that identifies the length of the entire vector.
4 (4)	4	Address of vector type field This field points to a 1-byte field that identifies the type of vector that is passed to the exit.
8 (8)	4	Address of vector data This field points to the vector data that is passed to the exit.
12 (C)	4	Address of exit work area This field points to an 8-byte field used by the exit to store the address of the exit's allocated storage. VTAM saves the contents of this user field after every successful invocation of the configuration services XID exit routine. Thus, the contents are available to the exit routine the next time the routine is called. The last 4 bytes could be used to uniquely identify an exit's level or version. The last 4 bytes will be displayed following subsequent use of the DISPLAY EXIT command (see z/OS Communications Server: SNA Operation for information about this command).

Table 51. Configuration services XID exit routine parameter list (continued)

Dec (Hex) offset	Size (Bytes)	Description
16 (10)	4	For activation or deactivation functions, offset XX'10' can contain the address of a parameter string if you specify a value for the PARMS operand on the MODIFY EXIT command. The parameter string is only available for vector types BEGIN (XX'00') and END (XX'FF'). The parameter string is in the form of a 2-byte length field followed by the actual character data.

Configuration services XID exit routine output parameter list

When required, the exit routine returns the following parameter list to VTAM. The address of this parameter list must be returned in register 1.

Table 52. Configuration services exit routine output parameter list

Dec (Hex) offset	Size (Bytes)	Description
0 (0)	4	Address of vector (this includes the vector header and vector data).

Vector header

Each vector has a header that is common to all vector types. The format of the vector header is described in [Table 53 on page 89](#).

Table 53. Vector header format

Dec (Hex) offset	Size (Bytes)	Description
0 (0)	2	Length of vector including the vector header
2 (2)	1	Vector Type X'00' Begin X'01' Exchange identification (XID) X'02' Build X'03' Connection status X'04' Failure X'FF' End

Vector data formats

Six vectors are defined for the configuration services XID exit routine. The vector data field starts at offset X'3' following the vector header and has a variable length field, depending on the vector type. The format for each vector is described in the topics that follow.

Begin vector (X'00')

VTAM calls the exit routine with the begin vector either during VTAM initialization or after you issue a MODIFY EXIT,ID=*exitname*,OPTION=ACT (or OPTION=REPL) command. See [z/OS Communications Server: SNA Operation](#) for more information about this command. The exit then performs its initial setup and returns the data in the begin vector to VTAM. VTAM uses the begin vector to determine which specific tasks the exit is to perform. The exit must return the address of the begin vector in register 1 of the configuration services XID exit routine output parameter list.

Table 54. Begin vector format

Dec (Hex) offset	Size (Bytes)	Description
0 (0)	1	<p>Bit Setting</p> <p>B'1...'</p> <p>This dynamic build bit is an indication to VTAM to call the exit when VTAM receives an XID from a switched device that is not defined to VTAM. This bit must be on if the unknown device is to be defined dynamically.</p> <p>If this bit is on, receipt of all XIDs that are not boundary function (BF)-based APPN TGs causes VTAM to call the exit with an XID vector. In addition, receipt of BF-based APPN TG XIDs causes VTAM to invoke the exit if requested using bits 3, 4, and 5 below.</p> <p>B'.1..'</p> <p>This connection status bit indicates whether the exit routine will be sent connection status information about all switched PUs. If this bit is on when the vector is returned from the exit, VTAM calls the exit with the connection status vector.</p> <p>B'..1.'</p> <p>This call-if-PU-defined bit is an indication to VTAM to call the exit routine with an XID vector when VTAM receives an XID for a PU that is already defined to VTAM.</p> <p>B'...1'</p> <p>This APPN connection network bit is an indication to VTAM to call the exit routine with an XID vector when VTAM receives an XID for a connection from a destination node through a connection network (the XID was received on a dial-in link connected to a connection network).</p> <p>B'.... 1...'</p> <p>This APPN ACTPU suppression bit is an indication to VTAM to not call the exit routine with an XID vector when VTAM receives an XID requesting a BF-based APPN TG connection that indicates ACTPU suppression.</p> <p>B'.... .1..'</p> <p>This APPN ACTPU bit is an indication to VTAM to not call the exit routine with an XID vector when VTAM receives an XID requesting a BF-based APPN TG connection requesting ACTPU.</p> <p>B'.... ..1.'</p> <p>This REB extension indicator bit is an indication to VTAM that the Extended Resource Entry Block (REB) is to be used.</p> <p>B'.... ...1'</p> <p>Reserved.</p>

Table 54. Begin vector format (continued)

Dec (Hex) offset	Size (Bytes)	Description
1 (1)	1	B'1...' This bit, when set on, indicates to VTAM that the exit wants to be called for EE XIDs. B'.111 1111' Reserved.
2 (2)	17	This field contains the network qualified control point name of the host in which the exit resides.

XID vector (X'01')

VTAM can call the exit with an XID vector when either a known or unknown PU attempts to dial in. The bit setting at offset 0 of the vector indicates whether the PU is to be defined dynamically or is already defined. See [Table 55 on page 92](#).

If the exit is called with an XID vector, the exit must return a build vector, as shown in [Table 56 on page 93](#). If the exit does not return a build vector, the connection fails.

If the exit routine is coded to support dynamic builds, VTAM calls the exit with an XID vector whenever an unknown PU attempts to dial in. The exit can use the information provided in the XID vector to construct and return a build vector and any necessary resource entry blocks, as shown in [Table 57 on page 94](#). The exit also returns a return code.

If the exit routine is coded such that VTAM calls the exit when the PU is already defined, VTAM calls the exit with an XID vector whenever a known PU attempts to dial in. This gives the exit the opportunity to examine the data in the XID vector and to instruct VTAM to either process or deny the REQCONT (or the REQACTPU for DLUS resources). The exit can use information provided in the XID vector to construct and return a build vector.

Table 55. XID vector format

Dec (Hex) offset	Size (Bytes)	Description
0 (0)	1	<p>Bit Setting</p> <p>B'1...' This bit tells the exit that the REQCONT or REQACTPU VTAM received is for a PU that VTAM will take over.</p> <p>B'.1..' This bit tells the exit that the REQCONT or REQACTPU VTAM received is for a previously defined switched PU.</p> <p>B'..1.' This bit indicates that the information being passed pertains to a DLUR. When this bit is on, the PU is being defined dynamically and a CPNAME is supplied in addition to the XID (see byte 0 of the XID at offset 79 (X'4F') below). The exit might use the CPNAME to select a PU instead of using the XID. The CP's NETID and name are stored in the fields at offsets 55 (X'37') and 63 (X'3F') below.</p> <p>B'...1' This bit tells the exit that the REQACTPU VTAM received contained an X'0E' vector for a PU name. VTAM did not find the PU defined. When this bit is on, the PU is being defined dynamically and will use the PU name that VTAM sends in the XID at offset 2 (X'2') as described below. The PU name is received in CV'OEF1' on REQACTPU and is passed in the 8-character field at offset 2.</p> <p>The remaining bits are reserved.</p>
1 (1)	1	This field is reserved.
2 (2)	8	This field contains the name of PU, if found by VTAM. Or, the name of the PU received on XID REQACTPU from the DLUR if bit X'10' is on at offset 0. This name will be used as the real PU name in the Resource Entry Block.
10 (A)	17	This field contains the network-qualified control point name of the host in which the exit resides.
27 (1B)	4	This field contains the subarea number from which the request was received.
31 (1F)	8	<p>This field contains one of the following:</p> <ul style="list-style-type: none"> • The name of the boundary function controlling resource • The CPNAME of the DLUR that controls the PU.
39 (27)	8	Network identifier of the DLUR supporting this PU.
47 (2F)	8	This field contains the name of the switched line over which VTAM received the XID.
55 (37)	8	This field contains the NETID operand coded on the PU definition statement or in the switched parameter list.
63 (3F)	8	This field contains the CPNAME operand coded on the PU definition statement or in the switched parameter list.

Table 55. XID vector format (continued)

Dec (Hex) offset	Size (Bytes)	Description
71 (47)	8	This field contains the station ID coded on the PU definition statement or in the switched parameter list. The station ID includes the station ID base and the station ID IDBLK or IDNUM.
79 (4F)	<i>n</i>	This is byte 0 of the XID, including control vectors. See <i>SNA Formats</i> for XID mapping.

Build vector (X'02')

The exit returns the build vector to VTAM in response to being called by the XID vector (X'01'). The exit must return the address of the build vector in register 1 of the configuration services XID exit routine output parameter list; the exit also must return a return code.

The CSISP macrolibrary is needed to assemble the XID exit routine. The CSISP macrolibrary must be the first macrolibrary in the global macrolibrary list. See [“Installing VTAM exit routines” on page 173](#) for more information.

If the exit is called for dynamic builds and VTAM is to process the REQCONT (or REQACTPU for DLUS resources), the build vector returns model PU and LU names and real PU and LU names for VTAM to use when defining the switched resource. The build vector also returns the major node name, when appropriate, under which the dynamic switched resource will be added. If the exit expects dynamic builds to occur, the X'80' bit of byte 0 of the build vector must be 0 and there must be resource entry blocks for each dynamic device.

For a description of how the sample exit identifies the switched resource and generates a valid unique name for the resource, see [Appendix D, “Sample configuration services XID exit routine,” on page 267](#).

If the exit routine requests that VTAM call the exit when VTAM receives a REQCONT (or a REQACTPU for DLUS-supported resources), for a PU that is already defined, VTAM calls the exit with an XID vector whenever a known PU attempts to dial in. The exit must return a build vector that indicates whether VTAM is to allow the connection. If VTAM allows the connection, and the device is already defined, there is no need for any resource entry blocks. In this situation, VTAM ignores any resource entry blocks that might be contained in the build vector.

Table 56. Build vector format

Dec (Hex) offset	Size (Bytes)	Description
0 (0)	1	<p>Flag byte (Bit Setting)</p> <p>B'1...' Indicates whether VTAM is to process the REQCONT (or the REQACTPU for DLUS-supported devices).</p> <p>If this bit is on, VTAM does not process the REQCONT or the REQACTPU.</p> <p>B'.... ...1' Indicates whether an 8-character major node name follows this flag byte.</p> <p>The remaining bits are reserved.</p>

This field contains one of the following indications based on whether a major node name is supplied in the build vector:

Table 56. Build vector format (continued)

Dec (Hex) offset	Size (Bytes)	Description
1 (1)	8	This is the 8-byte name of the dynamic switched major node under which the resources defined in the resource entry blocks that follow the build vector will be added. If the named major node does not exist, it will be created by VTAM.
0 (0)	1	This is byte 0 of the first resource entry block.

Resource entry blocks

The exit should build a set of resource entry blocks for each dynamic resource associated with the XID vector.

The first resource entry block must be for a PU definition. You can have one PU entry and a variable number of LU entries, depending on the PU type.

- PU type 2 devices, the maximum number of LUs allowed is 255
- PU type 1 devices, the maximum number of LUs allowed is 64

The format for each entry is the same. You need a block for every dependent LU attached to the PU. The exit cannot build independent LUs from a model. If the exit attempts to build an independent LU, the resource entry block for that LU will fail. If a resource entry block returns a PU, LU, or model resource name that is not valid, the resource entry block fails.

Table 57. Configuration services XID exit resource entry block format

Dec (Hex) offset	Size (Bytes)	Description
0 (0)	1	Resource entry block type X'71' This block is for a PU. X'81' This block is for an LU.
1 (1)	1	Bit setting B'1...' If this bit is on, the value coded in the address override field (offset X'12' below) overrides the address in the model definition. The remaining bits are reserved.
2 (2)	8	This is the minor node name of the PU or LU, created dynamically, that is represented by this resource entry block. VTAM will define the dynamic device by this name. If the X'10' bit was on in the XID vector received by the exit, then this will be the name received by VTAM in the REQACTPU from a DLUR when the PU was not found. VTAM passes the name to the exit to use as the name for the dynamic PU.
10 (A)	8	This is the name of a model resource defined in a model major node.

Table 57. Configuration services XID exit resource entry block format (continued)

Dec (Hex) offset	Size (Bytes)	Description
18 (12)	1	<p>This address override field contains one of the following addresses:</p> <ul style="list-style-type: none"> • If this resource entry block represents a PU, this field contains a station address. • If this resource entry block represents an LU, this field contains a local address. <p>The value in this field is ignored unless the bit in byte 1 of this resource entry block is set to indicate that the value is valid. Descriptions of the local address and station address fields are found in the z/OS Communications Server: SNA Resource Definition Reference.</p>
19 (13)	1	<p>Bit setting</p> <ul style="list-style-type: none"> • If this resource entry block represents a PU: <p>B'1...' If this bit is on, the value coded in the MAXOUT override field (offset X'14' below) overrides the field coded in the model definition.</p> <p>B'.1..' If this bit is on, the value coded in the MAXDATA override field (offset X'15' below) overrides the field coded in the model definition.</p> • If this resource entry block represents an LU: <p>B'1...' If this bit is on, the value coded in the DLOGMOD override field (offset X'14' below) overrides the field coded in the model definition.</p> <p>B'.1..' If this bit is on, the value coded in the LOGTAB override field (offset X'1C' below) overrides the field coded in the model definition.</p> <p>B'..1.' If this bit is on, the value coded in the MODE table override field (offset X'24' below) overrides the field coded in the model definition.</p> <p>B'...1' If this bit is on, the value coded in the LOGAPPL override field (offset X'2C' below) overrides the field coded in the model definition.</p> <p>B'... 1...' If this bit is on, the value coded in the USS table override field (offset X'3C' below) overrides the field coded in the model definition.</p>
If the resource entry block represents a PU, the following formats apply:		
20 (14)	1	MAXOUT override

Table 57. Configuration services XID exit resource entry block format (continued)

Dec (Hex) offset	Size (Bytes)	Description
21 (15)	2	MAXDATA override
23 (17)	45	The resource entry block for a PU must be equal that for an LU.
If the resource entry block represents an LU, the following formats apply:		
20 (14)	8	DLOGMOD—default LOGMODE name override
28 (1C)	8	LOGTAB—interpret table override
36 (24)	8	MODE table override
44 (2C)	16	LOGAPPL value override
60 (3C)	8	USS table override
68 (44)	0	This byte is the start of the next resource entry block, if any. If there are no more resource entry blocks, this byte does not appear.

Connection status vector (X'03')

VTAM calls the exit with the connection status vector whenever the connection status of a switched PU changes. You can use this vector to record the connection and disconnection times of switched devices. VTAM does not expect the exit to return any information other than a return code.

Table 58. Connection status format

Dec (Hex) offset	Size (Bytes)	Description
0 (0)	1	<p>Bit Setting</p> <p>B'1...' If this bit is on, the device named in the vector is connecting to the network.</p> <p>B'.1..' If this bit is on, the device named in the vector is disconnecting from the network.</p> <p>The remaining bits are reserved.</p>
1 (1)	8	<p>This field contains one of the following:</p> <ul style="list-style-type: none"> • The name of the switched line to which the PU is, or was, connected • For DLUS connections, the name of the DLUR that supports this PU.
9 (9)	8	This field contains the name of the PU whose status changed.
17 (11)	8	This field contains the time stamp that tells when the PU's status changed. The time stamp is retrieved from the system time-of-day clock.
25 (19)	8	This field contains the station ID of the PU, if available.
33 (21)	8	This field contains the CP name of the PU, if available.

Failure vector (X'04')

VTAM calls the exit with the failure vector when a dynamic build request fails. VTAM does not expect the exit to return any information.

Table 59. Failure vector format

Dec (Hex) offset	Size (Bytes)	Description
0 (0)	1	Bit setting B'1...' If this bit is on, the dynamic build of a PU failed. B'.1..' If this bit is on, the dynamic build of an LU failed. B'..1.' If this bit is on, the dynamic build of a switched major node failed. The remaining bits are reserved.
1 (1)	8	This field contains the name the exit provided for VTAM to use as the name of the device defined dynamically.
9 (9)	8	This field contains the name of the model resource VTAM used to try to build the device dynamically.

End vector (X'FF')

VTAM calls the exit with the end vector during normal HALT processing to indicate to the exit that VTAM is terminating or that a MODIFY EXIT,OPTION=INACT (or OPTION=REPL) command has been issued. VTAM does not expect the exit to return any information.

Table 60. End vector format

Dec (Hex) offset	Size (Bytes)	Description
0 (0)	2	Reserved

Selection of definitions for dependent LUs exit routine

The dynamic definition of dependent LUs function defines LUs to a VTAM network when the IBM 3174 control unit containing the LUs powers on, rather than during major node activation. To enable the function, define the device to support dynamic definition. See *z/OS Communications Server: SNA Resource Definition Reference* and *z/OS Communications Server: SNA Network Implementation Guide* for more information about how to enable and use the function. With this support, VTAM builds LU definitions from reusable model LU definitions rather than using predefined LUs. The LU definitions are replaced or changed each time the device containing the LUs powers on. You can use this function to add, change, or relocate dependent LUs from a VTAM network without reactivating the major node.

The SDDLUI exit routine allows you to specify the name of an LU being dynamically defined. The exit also allows you to specify which model LU VTAM uses to create the LU's definition. IBM supplies a default SDDLUI exit routine in the VTAMUSER LOADLIB under the VTAM module, ISTEXCSD. The source for this default exit routine is stored in SYS1.SAMPLIB.

See “Installing VTAM exit routines” on page 173 for more information. You can print a complete copy of the source code for the default exit from the library. VTAM uses the default exit unless you supply your own exit routine.

To enable the dynamic definition function, code the LUGROUP operand on the definition statement for the PU type 2 or type 2.1 device that supports the function. If you use the IBM supplied default exit, the LUSEED operand must also be coded on the PU's definition statement. See [z/OS Communications Server: SNA Resource Definition Reference](#) and [z/OS Communications Server: SNA Network Implementation Guide](#) for more information about these operands.

VTAM either initializes the SDDLUI exit during VTAM initialization, or activates the routine after you issue a MODIFY EXIT,ID=*exitname*,OPTION=ACT command.

You can use the DISPLAY EXIT command to display information regarding an SDDLUI exit. You also can use a MODIFY EXIT command to activate, deactivate, or replace the SDDLUI exit routine without interrupting VTAM processing. See [“Operator commands for VTAM exit routines”](#) on page 174 for more information about using the MODIFY EXIT command to modify VTAM exit routines. See [z/OS Communications Server: SNA Operation](#) for more information about these commands.

Naming dynamically defined dependent LUs

When a dependent LU is being dynamically defined for the first time on a PU, the SDDLUI exit must create a unique name for the LU. VTAM supplies the exit with fields of information to use to generate the LU name.

The IBM default exit uses the LUSEED operand on the PU's definition statement and the LU's local address to generate the name. See [z/OS Communications Server: SNA Network Implementation Guide](#) for a description of the algorithm used to create the LU name.

A user-written exit can use its own method to create the LU's name. VTAM supplies the exit with the following information that the exit can use to create the name:

- LUSEED operand (if coded)
- PU name
- Line name
- Local address
- LU encryption key name
- Major node name of the line
- Network ID
- Subarea number
- SSCP name
- Address of Reply/product-set identification (PSID) major vector sent to VTAM

If an LU was previously defined through the dynamic definition function, VTAM passes the current name of the LU to the exit, but the exit cannot change the name.

Specifying a model LU

In addition to naming the dynamically defined LUs, the SDDLUI exit can also indicate which model LU from the PU's model LU group VTAM is to use when creating the LU's definition.

VTAM supplies the exit with the 7-character EBCDIC machine type and model number (the product set ID) of the device that powered on. The IBM default exit uses the machine type and model number as the name of the model LU that VTAM uses to create the LU's definition.

A user-written exit can use the product set ID (PSID) supplied by VTAM for the model LU name, or it can derive any other value for the model LU name. Because VTAM provides the SDDLUI exit with the address of the Reply/PSID major vector, a user-written exit can use other device-specific subvectors that might be present in the major vector to select a model LU name. For example, subvectors X'04' (SNA Address List), X'10' (Product Set ID), and X'82' (Port-Attached Device Configuration Description) are present in the major vector passed to the exit. In addition, depending on the product, subvector X'88' (Detailed Data) might be present in the major vector. For additional information, see the following publications:

- Product publications for the PU type 2.0 or 2.1 devices that support Reply/PSID NMVTs for dynamic definition of dependent LUs contain detailed information about the other subvectors present in the major vector.
- See *SNA Formats* for detailed information about the Reply/PSID NMVT and its subvectors.
- [z/OS Communications Server: SNA Network Implementation Guide](#) contains an example of how VTAM builds dynamic definitions for dependent LUs.

Failing dynamic definition requests

When a device that is supported by the dynamic definition function powers on, VTAM passes a dynamic definition request to the SDDL exit. VTAM passes one dynamic definition request for each device that powers on. One dynamic definition request can represent one or more LUs being dynamically defined for the device.

A user-written exit can choose to fail the definition of specific LUs in the dynamic definition request. The exit can also choose to fail the definitions of all the LUs in the dynamic definition request. While the IBM default exit does not fail dynamic definition requests for individual LUs in a multiple LU request, it does fail the entire request if the LUSEED operand is not compatible with the default exit. The LUSEED operand is not compatible with the default exit if any of the following occur:

- The LUSEED operand is blank.
- Two or three adjacent # characters are not found in the LUSEED operand.
- The LUSEED operand's first character is a # character or a number.

The exit, not system definition, performs this error checking.

When VTAM calls the exit with a dynamic definition request vector, VTAM might have already failed some of the LU definitions in the request. For example, if one of the LUs was predefined, VTAM would have already failed the dynamic definition for the LU. This is because a dynamic definition cannot override an existing definition. The exit cannot change any fields of information in a dynamic definition request for an LU definition that VTAM has already failed.

The topics that follow contain information you need to write an SDDL exit routine.

Initial register contents

When the SDDL exit routine gains control, register contents are as follows:

Register 1:

Address of the parameter list described in [Table 61 on page 100](#)

Register 13:

Address of a standard 72-byte save area

Register 14:

Return address

Register 15:

Address of the entry point of this routine

Final register contents

The routine must leave the register status as follows:

Register 1:

Address of the parameter list passed to the exit on input

Register 2–14:

Restored to entry contents

Register 15:

Return code:

X'00' (0)

Processing successfully completed

X'01'–X'4F' (1–79)

Definition selection failed; the SDDL exit remains active. VTAM issues message IST1183I displaying the return code.

X'50'–X'7F' (80–127)

Reserved

X'80'–X'EF' (128–239)

Definition selection failed; the SDDL exit is deactivated. VTAM issues message IST1183I displaying the return code.

X'F0'–X'FF' (240–255)

Reserved

Note: When this return code is nonzero, all dynamic definitions in this request fail.

Design requirements

Follow these procedures when writing this routine:

- Use standard linkage.
- Save registers 2–14.

Consider the following restrictions when writing this routine:

- The name of the SDDL exit routine module should be ISTEXCSD; however, if you have written an alternate load module, use the load module name you assigned to your replacement module.
- Do not issue any SVCs if this exit routine is running in SRB mode.
- This exit routine should use only conditional storage invocations. You can reduce the possibility of a VTAM abend during a storage shortage by coding conditional storage invocations.
- Data is addressable in 31-bit mode only. Data is always presented with 31-bit addressability.
- This exit routine can be above or below the 16M line.
- This exit routine must be link-edited into the appropriate library. See [“Installing VTAM exit routines” on page 173](#) for more information.
- Exit routines that perform I/O must be link-edited AMODE=31 and RMODE=24.
- This exit routine runs under a VTAM subtask that permits the exit to perform any necessary I/O or other processing without affecting the VTAM main task.
- Do not use VTAM macroinstructions in this routine.
- This exit routine operates enabled in pageable storage. The routine gains control in supervisor state with a VTAM storage key. Errors in the routine could damage VTAM or system control blocks or modules.

SDDL exit routine parameter list

When the exit gets control, the address of the following parameter list is in register 1.

Table 61. SDDL exit routine parameter list

Dec (Hex) offset	Size (Bytes)	Description
0 (0)	4	Address of the vector length field This field points to a 2-byte field that identifies the length of the entire vector, including the length of the vector header.

Table 61. SDDL U exit routine parameter list (continued)

Dec (Hex) offset	Size (Bytes)	Description
4 (4)	4	<p>Address of vector type field</p> <p>This field points to a 1-byte field that identifies the type of vector that is passed to the exit.</p> <p>Type</p> <p>Description</p> <p>X'05' Dynamic definition request</p> <p>X'06' Begin</p> <p>X'07' End</p>
8 (8)	4	<p>Address of vector data</p> <p>This field points to the vector data that is passed to the exit. When VTAM calls the exit with a begin or end vector, this field is set to 0.</p>
12 (C)	4	<p>Address of exit work area</p> <p>This field points to an 8-byte field that a user-written exit can use. The default exit does not use this field because it does not allocate storage. VTAM saves the contents of this user field after every successful invocation of the SDDL U exit routine. In this way, the contents are available to the exit routine the next time the routine is called. A discussion on field usage follows:</p> <p>Byte</p> <p>X'00'–X'03' When VTAM calls the exit with a begin vector, the exit allocates storage and places the address of the storage in this field.</p> <p>When VTAM calls the exit with a dynamic definition request vector or an end vector, this field contains the value supplied as output from begin vector processing.</p> <p>X'04'–X'07' These can be used to uniquely identify an exit's level or version. The last 4 bytes are displayed following subsequent use of the DISPLAY EXIT command (see z/OS Communications Server: SNA Operation for information about this command).</p>
16 (10)	4	Address of the parameter string

For activation and deactivation functions, offset X'10' can contain the address of a parameter string if you specify a value for the PARM operand on the MODIFY EXIT command. The parameter string is only available for vector types BEGIN (X'00') and END (X'FF'). The parameter string is in the form of a 2-byte length field followed by the actual data.

SDDL U exit routine vectors

Three vectors are defined for the SDDL U exit routine. The vectors are described in the topics that follow.

Begin vector (X'06')

VTAM calls the exit routine with the begin vector when the exit is being activated or replaced. The exit performs its initial setup and returns to VTAM.

Vector data for the begin vector for the SDDLUI exit is:

Byte 0–1

Map of Begin vector

Bit 0:

1= Use REB extended length

0= Use base REB length

Bits 1–7

Unused

If a user-written SDDLUI exit allocates storage and places the address of the storage in the exit work area address specified at offset X'0C' in the input parameter list, the address of the storage is available to the exit during future calls.

Dynamic definition request vector (X'05')

This vector represents the local addresses of the LUs contained in the device when the device powered on. The vector data has two parts. The first part is a fixed-length structure that contains information relevant to all of the LUs in this request. The second part is a list of resource entry blocks that contain information specific to each local address.

The format of the dynamic definition request vector is described in [Table 62 on page 102](#).

Table 62. Dynamic definition request vector format

Dec (Hex) offset	Size (Bytes)	Symbolic name	Description
0 (0)	8	DDRSSCP	VTAM SSCP name
8 (8)	8	DDRNODE	Major node name of the line
16 (10)	8	DDRLINE	Real line name
24 (18)	8	DDRNETID	Network ID of the PU
32 (20)	8	DDRPU	Name of the PU where the terminal powered on
40 (28)	4	DDRSUBA#	Subarea number of the PU where the terminal powered on
44 (2C)	8	DDRLUGRP	Name of the LUGROUP operand specified on the PU's definition statement
52 (34)	8	DDRLUSD	Value of the LUSEED operand specified on the PU's definition statement, if any
60 (3C)	8	DDRTS	Time stamp indicating when the PU was activated
68 (44)	7	DDRSNUM	Serial number of the device that powered on
75 (4B)	1	DDR#LUS	Number of resource entry blocks that follow

Table 62. Dynamic definition request vector format (continued)

Dec (Hex) offset	Size (Bytes)	Symbolic name	Description
76 (4C)	4	DDRFAIL	Return code applicable to the entire request vector Bit Setting B'1...' This bit indicates that all dynamic builds in this request will fail. The remaining bits are reserved.
80 (50)	4	DDRRPMV	Address of the Reply/PSID major vector
84 (54)	1	DDRREB	Byte 0 of the first resource entry block

Resource entry blocks

There is a resource entry block for each LU contained in the device that powered on. For the resource entry blocks that have the REBERROR bit set to 0, the exit is responsible for ensuring that the resource entry block contains a model LU name and a valid VTAM name for the LU.

If the REBDEFND bit is set to 1, the resource entry block represents the redefinition of an existing LU. In this case, the LU name is already in the REBNAME field and cannot be changed by the exit. If the REBDEFND bit is set to 0, the resource entry block represents the definition of a dynamic LU that has not yet been named. The exit must generate an LU name and place it in the REBNAME field.

The REBMODEL field of the resource entry block contains the model acronym of the device that powered on. The exit can use the model acronym or override it with another model name from the model LU group specified on the PU definition statement for the device the dynamic definition request represents.

The format of a resource entry block is described in [Table 63 on page 103](#).

Table 63. SDDL U resource entry block format

Dec (Hex) offset	Size (Bytes)	Symbolic name	Description
0 (0)	1	REBTYP	Resource type This is always X'01' for the SDDL U exit, which indicates the resource is an LU.
1 (1)	1		Resource entry block flags
		REBADOV	These bits are reserved.
		REBERROR	B'1..' If this bit is set to 1 when the exit is called, this resource entry block does not represent a valid LU definition and must be ignored by the exit. If the exit sets this bit to 1, VTAM will not define the LU represented by this resource entry block.

Table 63. SDDL resource entry block format (continued)

Dec (Hex) offset	Size (Bytes)	Symbolic name	Description
		REBDEFND	<p>B'..0.' This resource entry block is for an LU that has not been defined previously at the local address. The exit must generate an LU name to be assigned to the LU.</p> <p>B'..1.' This resource entry block redefines an LU that was defined previously. The name of the LU is in the REBNAME field and the exit cannot change it.</p> <p>The remaining bits are reserved.</p>
		REBLUNMF	<p>B'..1.' This resource entry block contains an LU encryption key name field, REBLUKNM.</p>
2 (2)	8	REBNAME	<p>Name of dynamic resource</p> <p>If the REBDEFND bit is set to 0, this resource entry block represents the definition of a new LU. The exit must generate an LU name to be assigned to the LU.</p> <p>If the REBDEFND bit is set to 1, this field contains the current name of the LU. The exit cannot change this name.</p>
10 (A)	8	REBMODEL	<p>Model LU name</p> <p>VTAM uses this model name to select the model LU definition from the model LUGROUP operand specified on the PU's definition statement. On input, this field contains the model acronym of the device this dynamic definition request represents. The default exit does not modify this field. The exit uses this model acronym as the name of the model LU VTAM uses to define the LU.</p> <p>A user-written exit can use the model acronym supplied on input, or change this field to any model LU name from the model LU group for the device this dynamic definition request represents.</p>
18 (12)	1	REBADDR	Local address of the LU being defined

Table 63. SDDL resource entry block format (continued)

Dec (Hex) offset	Size (Bytes)	Symbolic name	Description
19 (13)	8	REBLUKNM	<p>Name of the LU encryption key.</p> <p>If the REBLUNMF bit is set to 1, this field is defined and is initialized to hex zeros. If the field is changed from hex zeros, VTAM uses this as the name of the LU's encryption key. If the field is not changed, the LU encryption key name defaults to the LU name.</p> <p>If the REBLUNMF bit is set to 0, this field is not defined and the LU encryption key name defaults to the LU name.</p>

End vector (X'07')

VTAM calls the exit with the end vector when the exit is being deactivated or replaced. There is no vector data associated with the end vector.

A user-written exit should free any storage it allocated during begin vector processing. The address of the storage is in the exit work area address at offset X'0C' in the input parameter list if the field was set during begin function processing.

Command verification exit routine

VTAM allows you to use a command verification exit routine to screen and manipulate VTAM commands entered by an operator. (The term operator includes a program operator.) You can use a command verification exit routine on VTAM DISPLAY, MODIFY, and VARY commands. When an operator enters a command, VTAM passes the address of the command string and the address of the command string length field to the exit. The exit verifies the specifications listed in the command and determines whether VTAM should process the command as entered by the operator. When necessary, the exit can change or remove operands and values from the command or indicate to VTAM not to process the command. The primary purpose of the command verification exit is to screen command requests that affect critical nodes in the network. You can create a list of critical nodes when you write the exit routine.

You might want to use this exit to:

- Discontinue a request to deactivate a critical network node
- Verify a password on activate, deactivate, or other system requests
- Disallow any VTAM command that might have a negative impact on the current network configuration

IBM does not supply a default command verification exit routine. If you want the function of the exit, write a command verification exit routine. Sample code for this exit is in [Appendix E, "Command verification exit routine,"](#) on page 275. If you write a routine and link-edit it into the appropriate library, VTAM will load the exit into pageable VTAM private storage and mark the exit active during VTAM initialization.

VTAM either initializes the command verification exit routine during VTAM initialization, or activates the routine after you issue a MODIFY EXIT,ID=exitname,OPTION=ACT command.

You can use the DISPLAY EXIT command to display information regarding a command verification exit. You also can use a MODIFY EXIT command to activate, deactivate, or replace the command verification exit routine without interrupting VTAM processing. See ["Operator commands for VTAM exit routines"](#) on page 174 for more information about using the MODIFY EXIT command to modify VTAM exit routines. See [z/OS Communications Server: SNA Operation](#) for more information about these commands.

The topics that follow contain information you need to write a command verification exit routine.

Initial register contents

When VTAM passes control to the command verification exit routine, register contents are as follows:

Register 0:

Entry code

X'04'

Exit invocation

X'10'

Exit activation

X'18'

Exit replacement

X'20'

Exit deactivation

Register 1:

Address of the command verification exit parameter list described in [Table 64 on page 107](#) or in [Table 65 on page 108](#)

Register 13:

Address of a standard 72-byte save area

Register 14:

Return address

Register 15:

Address of the command verification exit's entry point

Final register contents

Register 1:

Address of the command verification exit parameter list with modified values

Registers 2–14:

Restored to entry contents

Register 15:

Return code

X'00' (0)

Processing successfully completed

X'01'–X'4F' (1–79)

Exit processing did not work; the command verification exit remains active.

X'50'–X'7F' (80–127)

Reserved

Greater than X'7F' (127)

Invocation failed: command denied and command verification exit deleted and deactivated

X'F0'–X'FF' (240–255)

Reserved

Design requirements

Follow these procedures when writing this routine:

- Use standard linkage.
- Save and restore registers 2–14.

Consider the following restrictions when writing this routine:

- The name of the command verification exit routine module should be ISTCMMND; however, if you have written an alternate load module, use the load module name you assigned to your replacement module.

- Do not issue any SVCs if this exit routine is running in SRB mode.
- This exit routine should use only conditional storage invocations. You can reduce the possibility of a VTAM abend during a storage shortage by coding conditional storage invocations.
- This exit routine is addressable in 31-bit mode only. Data is always presented with 31-bit addressability.
- This exit routine can be above or below the 16M line.
- This exit routine must be link-edited into the appropriate library. See [“Installing VTAM exit routines” on page 173](#) for more information.
- This exit routine operates as an internal VTAM routine. VTAM performance might be degraded if the routine requires lengthy processing time. While this exit routine has control, VTAM does not process any new VTAM operator requests, session initiation requests, or session termination requests for any resource.
- A program check causes VTAM to abend.
- System waits, including implied waits for I/O operations, should be avoided. System waits can cause VTAM failure in some timing-dependent situations.
- This exit routine operates enabled in pageable storage. The routine gets control in supervisor state with a VTAM storage key. Errors in the routine could damage VTAM or system control blocks and modules.
- If VTAM calls the exit for processing, VTAM passes the addresses of the command string and the command string length field to the exit when the operator enters the command. Do not alter any storage area beyond the end of the string or attempt to add characters to the command string. This type of processing could damage VTAM or system control blocks and modules.

Return codes

Two return codes can be used to control exit processing. The first is the command verification return code in the command verification exit parameter list shown in [Table 64 on page 107](#). If this return code is set to 0, the exit has successfully modified the command string, if requested, and allowed the command to continue. If this return code is set to any value other than 0, the exit does not perform any processing and the operator command is denied. This causes message IST1201I COMMAND REJECTED BY ISTCMMND EXIT to be issued.

The second return code is in register 15 in the final register contents. Register 15 can be set to discontinue processing of the exit completely. If a severe error is detected within the exit, register 15 can be set to greater than X'7F' and VTAM will no longer call the exit for processing. Be sure to clear register 15 to 0 before returning from the exit to VTAM if the exit processing is working correctly.

Command verification exit parameter list

VTAM passes the following parameter list to the command verification exit routine in register 1, when the entry code in register 0 is X'04'.

Table 64. Command verification exit parameter list for X'04'

Dec (Hex) offset	Size (Bytes)	Description
0 (0)	4	Address of an 8-byte user field
4 (4)	8	Reserved
12 (C)	4	Address of a 4-byte command verification return code field
16 (10)	4	Address of the 2-byte command string length field
20 (14)	4	Address of the command string

VTAM passes the following parameter list to the command verification exit routine in register 1, when the entry code in register 0 is X'10', X'18', or X'20'.

Table 65. Command verification exit parameter list for X'10', X'18', and X'20'

Dec (Hex) offset	Size (Bytes)	Description
0 (0)	4	Address of an 8-byte user field
4 (4)	8	Reserved
12 (C)	4	Address of parameter string

For an activation, deactivation, and replacement function, offset X'C' can contain a parameter string, if you specify a value for the PARMS operand on the MODIFY EXIT command. The parameter string is only available for entry codes X'10', X'18', or X'20' in Register 0. The parameter string is in the form of a 2-byte length field followed by the actual data.

User data field

The exit routine can use this 8-byte user data field, originally initialized to 0, for any purpose. For example, the exit could use this field to store the address of a storage area obtained dynamically.

VTAM saves the contents of this user field after every successful invocation of the command verification exit routine. Thus, the contents are available to the exit routine the next time the routine is called. This field is cleared to zero when the command verification exit becomes inactive.

The last 4 bytes can be used to uniquely identify an exit's level or version. The last 4 bytes are displayed following subsequent use of the DISPLAY EXIT command. See [z/OS Communications Server: SNA Operation](#) for information about this command.

Examples of command verification processing

The following examples illustrate processing that the command verification exit can perform. In the examples, the exit is used to screen commands to deactivate nodes in a network. If the node's ID appears in the list of critical nodes, the exit screens the deactivation commands to look for the OVERRIDE option. In the examples, OVERRIDE is a keyword, defined in the exit routine, that can be added to a command string entered by the operator. The specification of OVERRIDE means that the command ignores any default verification checks. If the exit finds the OVERRIDE option, it removes the option from the command string. This prevents the operator from deactivating a critical node without first verifying the command.

The examples show the format of the command string as entered by the operator. Before VTAM passes control to the exit, VTAM converts the operator's character string into the format needed by the exit. The examples show the converted format of the character string. If the exit modifies the command string, the examples also show the format of the string the exit returns to VTAM.

The command string length field is the number of characters in the string as entered by the operator. If VTAM's format conversion reduces the number of characters in the string, blanks are added to the end of the string. Therefore, the length pointed to in the parameter list does not appear to match the length of the converted command string. A command string can be modified only by changing or deleting characters in the string. You cannot delete any of the blanks at the end of the string, or add characters to the string.

Examples of command verification processing follow:

Example 1: Deactivation of critical resource with OVERRIDE specified

1. The operator enters a command string:

```
V NET,INACT,ID=CRITICAL,OVERRIDE=YES
```

2. VTAM sends a pointer to the converted command string:

```
VARY INACT,ID=CRITICAL, OVERRIDE=YES
```

and a pointer to the command string length field (35) to the command verification exit.

3. In this example, CRITICAL is in a table of critical nodes contained in the exit. Therefore, the exit checks for the OVERRIDE option and removes the option from the command string.
4. The exit returns a pointer to the modified command string:

```
VARY INACT,ID=CRITICAL
```

and a pointer to a command verification return code of 0 to VTAM.

Example 2: Deactivation of critical resource with OVERRIDE not specified

1. The operator enters a command string:

```
V NET, INACT, ID=CRITICAL
```

2. VTAM sends a pointer to the converted command string:

```
VARY INACT,ID=CRITICAL
```

and a pointer to the command string length field (22) to the command verification exit.

3. In this example, CRITICAL is in a table of critical nodes contained in the exit. Therefore, the exit checks for the OVERRIDE option. Because OVERRIDE is not specified in the command string, the exit does not find the option and the exit fails.
4. The exit returns a pointer to a nonzero command verification return code to VTAM.

Example 3: Deactivation of a non-critical resource

1. The operator enters a command string:

```
V NET, INACT, ID=NONCRIT
```

2. VTAM sends a pointer to the converted command string:

```
VARY INACT,ID=NONCRIT
```

and a pointer to the command string length field (21) to the command verification exit.

3. In this example, NONCRIT is not in a table of critical nodes contained in the exit. Therefore the exit does not perform any processing on the command string.
4. The exit returns a pointer to a command verification return code of 0 to VTAM.

USERVAR exit routine

The user variable (USERVAR) exit routine can be used to translate USERVAR names to real names on a one-to-one basis.

For a dynamic or static USERVAR translation, a USERVAR destination logical unit (DLU) name is translated to the name of a real DLU that is maintained in a USERVAR table. The translated name is placed in the USERVAR value field in the USERVAR table. VTAM uses the DLU name in the USERVAR table in place of the USERVAR DLU name for subsequent session initiations where the USERVAR DLU name is specified. For a volatile USERVAR translation, VTAM establishes the session with the USERVAR value without updating the USERVAR table. The USERVAR value is subject to change with each session request.

VTAM allows you to use a USERVAR exit to provide the value for translation. You can specify that the exit is to be used for a specific USERVAR through the UVEXIT=YES option on the MODIFY USERVAR command. See *z/OS Communications Server: SNA Operation* for information about this command. The USERVAR exit enables you to customize the USERVAR translation and change the results of the USERVAR processing.

When VTAM receives a session initiation request specifying a USERVAR as a DLU name, the exit provides the value for the USERVAR.

You can use a MODIFY EXIT command to activate, deactivate or replace a USERVAR exit routine without interrupting VTAM processing. See [“Operator commands for VTAM exit routines”](#) on page 174 for more information about using the command to modify VTAM exit routines. See [z/OS Communications Server: SNA Operation](#) for information about the MODIFY EXIT command.

VTAM loads the USERVAR exit into fixed VTAM private storage and marks the exit active during initialization. If a user-managed USERVAR specifies use of the USERVAR exit, VTAM calls the exit whenever the USERVAR is added, updated, deleted, or needs to be translated. VTAM also calls the exit for exit activation and deactivation.

When the USERVAR needs to be added, updated, or deleted, VTAM builds the USERVAR exit parameter list as shown in [Table 66 on page 112](#), builds the USERVAR parameters as shown in [Table 68 on page 113](#), and drives the USERVAR exit.

When the SSCP of the DLU receives a session initiation request requiring USERVAR translation, it checks the USERVAR table to see whether the USERVAR specifies use of the exit. If the USERVAR does not specify use of the exit, the SSCP continues processing with a value retrieved from the USERVAR table. If the USERVAR does specify use of the exit, VTAM builds the USERVAR exit parameter list as shown in [Table 66 on page 112](#), builds the USERVAR parameters as shown in [Table 69 on page 113](#), and drives the USERVAR exit. The exit is then responsible for providing a translated DLU name.

If you want the function of the USERVAR exit, you must write a USERVAR exit routine and link-edit it into the appropriate library. (See [“Installing VTAM exit routines”](#) on page 173 for more information.)

Note: IBM supplies a sample USERVAR exit routine that is specific to the Transaction Processing Facility (TPF) environment. The sample routine is stored in SYS1.SAMPLIB under the VTAM module name ISTEXCUV. See [“USERVAR exit routine for TPF sessions”](#) on page 180 and Appendix F, [“Sample USERVAR exit routine for TPF sessions,”](#) on page 277 for information about the sample USERVAR routine. You can use the sample as it is coded or modify the sample to meet the needs of your installation.

The following topics contain information you need to write a USERVAR exit routine.

Initial register contents

When VTAM passes control to the USERVAR exit routine, register contents are as follows:

Register 0:

Entry code

X'04'

Invoke processing (see [“Invoke flags”](#) on page 112)

X'10'

Exit activation

X'18'

Exit replacing

X'20'

Exit deactivation

X'40'

VTAM termination

Register 1:

Address of the USERVAR exit parameter list described in [Table 66 on page 112](#)

Register 13:

Address of a standard 72-byte save area

Register 14:

Return address

Register 15:

Address of the USERVAR exit's entry point

Final register contents

The exit routine must leave the register status as follows:

Register 1:

Address of the USERVAR exit parameter list with modified values in the USERVAR parameters

Registers 2–14:

Restored to entry contents

Register 15:

Return code

X'0'

Processing successfully completed.

X'01'–X'4F'

The session request has failed; the USERVAR exit remains active.

X'50'–X'7F'

Reserved

X'80'–X'EF'

The session request has failed; the USERVAR exit is deactivated.

X'F0'–X'FF'

Reserved

Design requirements

Follow these procedures when writing the routine. Errors in the routine could damage VTAM or system control blocks and modules.

- Use standard linkage.
- Save and restore registers 2–14.

Consider the following restrictions when writing this routine:

- The name of the USERVAR exit routine module should be ISTEXCUV; however, if you have written an alternate load module, use the load module name you assigned to your replacement module.
- This routine operates enabled in pageable storage. Because the routine operates at VTAM's main task dispatching priority, there is a possibility of lockout if a wait requires other task action. The routine gains control in supervisor state with a VTAM storage key. Errors within the routine could cause damage to VTAM or system control blocks and modules.
- There can be no system waits, including implied waits for I/O operations. System waits cause VTAM failure in some timing-dependent situations.
- Do not issue any SVCs if this exit routine is running in SRB mode. The USERVAR exit is driven in SRB mode for all functions except invoke processing (entry code X'04'). See the description of register 0 at [“Initial register contents” on page 110](#).
- This exit routine should use only conditional storage invocations. You can reduce the possibility of a VTAM abend during a storage shortage by coding conditional storage invocations.
- Data is addressable in 31-bit mode only. Data is always presented with 31-bit addressability.
- This exit routine can be above or below the 16M line.
- Link-edit your routine into the appropriate library. See [“Installing VTAM exit routines” on page 173](#) for more information.
- This exit routine operates as an internal VTAM routine. VTAM performance might be degraded if the routine requires lengthy processing time. While this routine has control, VTAM does not process any new operator requests, session initiation requests, or session termination requests for any resource.

USERVAR exit routine parameter list

VTAM passes the following parameter list to the USERVAR exit routine in register 1, when the entry code in Register 0 is X'04'.

Note: For the sample USERVAR exit for TPF sessions, source code for this parameter list and the USERVAR parameters are within ELMCUVPL in SYS1.SISTMAC1.

Table 66. USERVAR exit parameter list for X'04'

Dec (Hex) offset	Size (Bytes)	Description
0 (0)	4	Address of an 8-byte user field
4 (4)	8	Reserved
12 (C)	4	Address of the invoke flags
16 (10)	4	Address of the USERVAR parameters

VTAM passes the following parameter list to the USERVAR exit routine in register 1, when the entry code in register 0 is X'10', X'18', or X'20'.

Table 67. USERVAR exit parameter list for X'10', X'18', and X'20'

Dec (Hex) offset	Size (Bytes)	Description
0 (0)	4	Address of an 8-byte user field
4 (4)	8	Reserved
12 (C)	4	Address of parameter string

For an activation, deactivation, and replacement function, offset X'C' can contain a parameter string, if you specify a value for the PARMS operand on the MODIFY EXIT command. The parameter string is only available for entry codes X'10', X'18', or X'20' in Register 0. The parameter string is in the form of a 2-byte length field followed by the actual data.

User data field

For the entry codes described in [Table 67 on page 112](#) and [Table 66 on page 112](#), the USERVAR exit routine can use this 8-byte user data field, originally initialized to 0, for any purpose. For example, the exit could use this field to store the address of a storage area obtained dynamically. VTAM saves the contents of this user field after every successful invocation of the USERVAR exit routine. Thus, the contents are available to the exit routine the next time the routine is called. This field is cleared to zero when the USERVAR exit becomes inactive.

The last 4 bytes can be used to uniquely identify an exit's level or version. The last 4 bytes are displayed following subsequent use of the DISPLAY EXIT command. See [z/OS Communications Server: SNA Operation](#) for information about this command.

Invoke flags

The address of the invoke flags is passed to the exit routine in the USERVAR exit parameter list. The invoke flags indicate the reason VTAM is calling the USERVAR exit for invoke processing. The invoke flags also determine which parameters VTAM will pass to the exit. The flags are:

X'01'

Request for USERVAR to be added

X'02'

Request for USERVAR to be updated

X'04'

Request for USERVAR to be translated

X'08'

Request for USERVAR to be deleted

USERVAR parameters

When VTAM calls the exit for invoke processing, it passes the address of certain parameters to the exit in register 1. There is one set of parameters for when the USERVAR is to be added, updated, or deleted, and another set of parameters for when a USERVAR is to be translated.

The address of the following parameters is passed to the USERVAR exit routine in the USERVAR exit parameter list when a USERVAR is to be added, updated, or deleted.

Table 68. USERVAR parameters for USERVAR addition, update, or deletion

Dec (Hex) offset	Size (Bytes)	Description
0 (0)	8	USERVAR name
8 (8)	8	USERVAR value
16 (10)	8	Reserved

The address of the following parameters is passed to the USERVAR exit routine in the USERVAR exit parameter list when a USERVAR is to be translated.

Table 69. USERVAR parameters for USERVAR translation

Dec (Hex) offset	Size (Bytes)	Description
0 (0)	8	OLU name
8 (8)	4	OLU subarea address in the DLU network
12 (C)	4	Address of the session partners list (see Table 70 on page 114)
16 (10)	8	CoS name for the session
24 (18)	8	Generic DLU name

Table 69. USERVAR parameters for USERVAR translation (continued)

Dec (Hex) offset	Size (Bytes)	Description
32 (20)	1	<p>Flags</p> <p>B'10..' The OLU is the SLU.</p> <p>B'01..' The OLU is the PLU.</p> <p>B'..1.' The USERVAR name was translated.</p> <p>B'..0.' The USERVAR name was not translated.</p> <p>B'...1 0...' The USERVAR is static.</p> <p>B'...0 1...' The USERVAR is dynamic.</p> <p>B'...1 1...' The USERVAR is volatile.</p> <p>B'.... .1..' In the OLU domain.</p> <p>B'.... ..1.' Network ID specified in the request.</p> <p>The remaining bit is reserved.</p>
33 (21)	3	Reserved
36 (24)	8	USERVAR value
44 (2C)	8	Reserved

If the translation flag indicates the name was not translated, VTAM considers the USERVAR to be unresolved and continues processing as though the USERVAR exit had not been invoked.

If the translation flag indicates the name was translated, VTAM uses the USERVAR value returned by the exit and continues processing. For a dynamic or static USERVAR, VTAM updates the USERVAR table and uses the USERVAR value returned by the exit as though the value were retrieved from the USERVAR table without the intervention of the exit. For a volatile USERVAR, VTAM establishes the session with the USERVAR value without updating the USERVAR table.

Session partners list

The address of the session partners list is passed to the exit when the exit is invoked for USERVAR translation. There is an entry in the list for each session partner with which the OLU is currently in session. The format of the list is shown in the following table.

Table 70. Session partners list

Dec (Hex) offset	Size (Bytes)	Description
0 (0)	2	Number of parallel sessions between this session partner and the OLU
2 (2)	2	Reserved

Table 70. Session partners list (continued)

Dec (Hex) offset	Size (Bytes)	Description
4 (4)	8	Reserved
12 (C)	8	Name of this session partner
20 (14)	4	Pointer to the next entry in the session partner list. This pointer is 0 if there are no more entries.

Directory services management exit routine

An installation-wide directory services management exit provides control over the extent to which VTAM conducts search requests in the node in which the exit is loaded. The directory services management exit is the interface between VTAM's directory services and user-written code. VTAM conducts searches based on information provided by user code.

The exit routine is supported by VTAM end nodes, network node servers, border nodes, and origin and alternate directory servers. It is also supported by intermediate network nodes, but on broadcast searches only. Intermediate network nodes on directed searches do not invoke the directory services management exit routine.

You can use the DISPLAY EXIT command to display information regarding a directory services management exit. If you specify ISTECDM on the ID operand of the DISPLAY EXIT command, you can see which functions are active for the directory services management exit. You also can use a MODIFY EXIT command to activate, deactivate, or replace the exit routine without interrupting VTAM processing. See [“Operator commands for VTAM exit routines” on page 174](#) for more information about using the MODIFY EXIT command to modify VTAM exit routines. See [z/OS Communications Server: SNA Operation](#) for more information about these commands.

If you need to trace information relative to directory services management exit routine activity, use the MODIFY TRACE and MODIFY NOTRACE commands. These commands allow you to trace the directory services management exit's input and output. You can specify which exit functions to trace on the OPTIONS operand of the MODIFY TRACE and MODIFY NOTRACE commands. To display the trace, use the DISPLAY TRACES command.

Note: Because of the amount of generated data, the tracing of all exit functions might negatively affect performance. For this reason, use extensive tracing selectively.

For information about trace output, see [z/OS Communications Server: SNA Diagnosis Vol 1, Techniques and Procedures](#). For information about the MODIFY TRACE command, the MODIFY NOTRACE command, and the DISPLAY TRACES command, see [z/OS Communications Server: SNA Operation](#).

This exit routine is optional. If you do not write an exit, all requests for directory services in your network are authorized. IBM does not supply a directory services management routine or sample code.

Initial register contents

When your code gets control, register contents are:

Register 1:

Address of the parameter list described in [Table 71 on page 118](#) and [Table 72 on page 119](#).

Register 13:

Address of a standard 72-byte save area

Register 14:

Return address

Register 15:

Address of the exit's entry point

Final register contents

Set up the following registers when returning control to VTAM:

Register 1:

Address of the exit's parameter list with modified values

Registers 2–14:

Restored to entry contents

Register 15:

Return code

The following descriptions are general descriptions of return codes. For more specific information, see the function descriptions starting at [“Begin function \(function code X'FE'\)”](#) on page 120.

X'00'–X'4F' (0–79)

Indicates exit output

X'50'–X'7F' (80–127)

Reserved

Greater than X'7F' (127)

Invocation failed; exit deleted and deactivated

X'F0'–X'FF' (240–255)

Reserved

If the exit return code is greater than X'7F', VTAM ignores any options you specified and issues error message IST985I.

Design requirements

Follow these procedures when writing this routine:

- Use standard linkage.
- Save registers 2–14.

Consider the following restrictions when writing this exit routine:

- The exit name is ISTECDM.
- The exit module name is the name of the exit load module. For a description of using the MODIFY EXIT command to modify this exit, see [z/OS Communications Server: SNA Operation](#).
- This exit routine must run in supervisor state, key 6, and SRB mode.

This routine operates enabled in pageable storage. Because the routine operates at VTAM's main task dispatching priority, there is a possibility of lockout if a wait requires other task action. The routine gains control in supervisor state with a VTAM storage key. Errors within the routine could cause damage to VTAM or system control blocks and modules.

- This exit routine must be serially reusable.
- Do not issue any SVCs if this exit routine is running in SRB mode.
- This exit routine should use only conditional storage invocations. You can reduce the possibility of a VTAM abend during a storage shortage by coding conditional storage invocations.
- All functions run under abend protection.
- This routine will be above the 16M line.
- This exit routine is addressable in 31-bit mode only. Data is always presented with 31-bit addressability.
- Do not use VTAM API macroinstructions in the exit routine.
- Link-edit your routine into the appropriate library. See [“Installing VTAM exit routines”](#) on page 173 for more information.
- This exit routine is called directly from VTAM modules. However, because this exit is invoked running in SRB mode, no other VTAM processes are degraded as a result of this exit being processed. Although

extended processing in this exit routine could degrade session setup time for APPN sessions, other VTAM processes are not affected.

Diagnostic information

If the directory services management exit routine returns either data or a return code that is not valid in register 15, VTAM issues message IST1183I (messages are described in [z/OS Communications Server: SNA Messages](#)).

Message IST1183I includes a diagnostic code that can be used to determine the error that occurred. The diagnostic codes have the following meanings:

Code (hex)

08

For the initial authorization function, the exit routine returned a return code that was not valid.

09

For the initial authorization function, the exit returned data that was not valid.

11

For the end function, the exit returned a return code that was not valid.

12

For the begin function, the exit returned a return code that was not valid.

14

The exit routine abended.

15

For any function, the exit routine cannot be invoked because insufficient storage exists for proper parameter list construction.

20

For the CDS selection function, the exit routine returned a return code that was not valid.

21

For the alternate CDS selection function, the exit routine returned a return code that was not valid.

22

For the central resource registration selection function, the exit routine returned a return code that was not valid.

23

For the replacing function, the exit routine returned a return code that was not valid.

24

For the replaced function, the exit routine returned a return code that was not valid.

25

An abend occurred either before or after the call to the exit routine.

26

For the ICN selection function, the exit routine returned a return code that was not valid.

27

For the ICN selection function, the exit routine returned data that was not valid.

Parameter lists

Several functions are defined for this installation-wide exit routine. For each function being performed, VTAM passes a list of parameters to this exit. All functions except the begin function are optional. A function code indicates which function the exit is to process and determines which parameters VTAM passes to the exit. Function codes are listed in [Table 71 on page 118](#) and [Table 72 on page 119](#); the parameters VTAM passes to the exit routine are described in topics following the tables.

Although the parameter lists vary, depending on the use of the exit routine, the basic format is the same for all the functions for which VTAM calls the exit routine.

The following functions are possible input parameter list functions:

- Begin
- Initial authorization
- Border node selection
- Interchange node selection
- CDS selection
- Alternate CDS selection
- Central resource registration selection
- Replacing
- Replaced
- End

Parameter list structure

Table 71 on page 118 is a summary of the parameter lists associated with each function in the directory services management exit routine.

Table 71. Directory services management exit parameter summary for function codes X'FE', X'00', X'01', X'02', X'03'

FUNCTION		BEGIN	INITIAL AUTHORIZATION	BORDER NODE SELECTION	INTERCHANGE NODE SELECTION	CDS SELECTION
Offset		Function code X'02'				
Dec	Hex	Function code X'FE'	Function code X'00'	Function code X'01'	Function code X'02'	Function code X'03'
0	0	Address of 8-byte user field	Address of 8-byte user field	Address of 8-byte user field	Address of 8-byte user field	Address of 8-byte user field
4	4	Reserved	Reserved	Reserved	Reserved	Reserved
8	8	Reserved	Reserved	Reserved	Reserved	Reserved
12	C	Environment vectors address: See Table 74 on page 127	Environment vectors address: See Table 74 on page 127	Environment vectors address: See Table 74 on page 127	Environment vectors address: See Table 74 on page 127	Environment vectors address: See Table 74 on page 127
16	10	Function code and related search information address: See Table 75 on page 128	Function code and related search information address: See Table 75 on page 128	Function code and related search information address: See Table 75 on page 128	Function code and related search information address: See Table 75 on page 128	Function code and related search information address: See Table 75 on page 128
20	14	User-supplied parameter data address	User-supplied parameter data address	User-supplied parameter data address	User-supplied parameter data address	User-supplied parameter data address
24	18	Exit options address: See Table 76 on page 133	OLU information structure: See Table 79 on page 136	OLU information structure: See Table 79 on page 136	OLU information structure: See Table 79 on page 136	OLU information structure: See Table 79 on page 136
28	1C	Border node and interchange node options, see Table 77 on page 134	DLU information structure: See Table 80 on page 136	DLU information structure: See Table 80 on page 136	DLU information structure: See Table 80 on page 136	DLU information structure: See Table 80 on page 136

Table 71. Directory services management exit parameter summary for function codes X'FE', X'00', X'01', X'02', X'03' (continued)

FUNCTION		BEGIN	INITIAL AUTHORIZATION	BORDER NODE SELECTION	INTERCHANGE NODE SELECTION	CDS SELECTION
Offset					Function code X'02'	Function code X'03'
Dec	Hex	Function code X'FE'	Function code X'00'	Function code X'01'		
32	20	Parameter string address from MODIFY EXIT, PARMS= command. The first 2 bytes of the field pointed to by this field contain the length of the PARMS <i>character_string</i> value or zeros.	Network- qualified adjacent CP name vector: See Table 81 on page 137	Network- qualified adjacent CP name vector: See Table 81 on page 137	Network- qualified adjacent CP name vector: See Table 81 on page 137	Network- qualified adjacent CP name vector: See Table 81 on page 137
36	24	N/A	Search correlator structure: See Table 82 on page 138	Search correlator structure: See Table 82 on page 138	Search correlator structure: See Table 82 on page 138	Search correlator structure: See Table 82 on page 138
40	28	N/A	PCID modifier structure: See Table 83 on page 138	PCID modifier structure: See Table 83 on page 138	PCID modifier structure: See Table 83 on page 138	PCID modifier structure: See Table 83 on page 138
44	2C	N/A	Search Task List: See Table 84 on page 138	Subnetwork routing list structure: See Table 85 on page 139	Interchange node list structure: See Table 78 on page 135	CDS List Structure: See Table 86 on page 141
48	30	N/A	Application GDS variable	N/A	N/A	N/A
Reg. 15 Return Code	0		0,4,8,12,16,20,24,28,32,36,40	0,4,8	0,4,8	0,4

Table 72. Directory services management exit parameter summary for function codes X'04', X'05', X'08', X'09', X'FF'

FUNCTION		ALTERNATE CDS SELECTION	CENTRAL RESOURCE REGISTRATION SELECTION	REPLACING	REPLACED	END
Offset						
Dec	Hex	Function code X'04'	Function code X'05'	Function code X'08'	Function code X'09'	Function code X'FF'
0	0	Address of 8-byte user field	Address of 8-byte user field	Address of 8-byte user field	Address of 8-byte user field	Address of 8-byte user field
4	4	Reserved	Reserved	Reserved	Reserved	Reserved
8	8	Reserved	Reserved	Reserved	Reserved	Reserved
12	C	Environment vectors address: See Table 74 on page 127	Environment vectors address: See Table 74 on page 127	Environment vectors address: See Table 74 on page 127	Environment vectors address: See Table 74 on page 127	Environment vectors address: See Table 74 on page 127
16	10	Function code and related search information address: See Table 75 on page 128	Function code and related search information address: See Table 75 on page 128	Function code and related search information address: See Table 75 on page 128	Function code and related search information address: See Table 75 on page 128	Function code and related search information address: See Table 75 on page 128
20	14	User-supplied parameter data address	User-supplied parameter data address	User-supplied parameter data address	User-supplied parameter data address	User-supplied parameter data address

Table 72. Directory services management exit parameter summary for function codes X'04', X'05', X'08', X'09', X'FF' (continued)

FUNCTION		ALTERNATE CDS SELECTION	CENTRAL RESOURCE REGISTRATION SELECTION	REPLACING	REPLACED	END
Offset						
Dec	Hex	Function code X'04'	Function code X'05'	Function code X'08'	Function code X'09'	Function code X'FF'
24	18	OLU information structure: See Table 79 on page 136	CDS List Structure: See Table 86 on page 141	Parameter String address from MODIFY EXIT, PARMS= command. The first two bytes of the field pointed to by this field contain the length of the PARMS <i>character_string</i> or zeros.	Parameter String address from MODIFY EXIT, PARMS= command. The first two bytes of the field pointed to by this field contain the length of the PARMS <i>character_string</i> or zeros.	Parameter String address from MODIFY EXIT, PARMS= command. The first two bytes of the field pointed to by this field contain the length of the PARMS <i>character_string</i> or zeros.
28	1C	DLU information structure: See Table 80 on page 136	N/A	N/A	N/A	N/A
32	20	Network- qualified adjacent CP name vector: See Table 81 on page 137	N/A	N/A	N/A	N/A
36	24	Search correlator structure: See Table 82 on page 138	N/A	N/A	N/A	N/A
40	28	PCID modifier structure: See Table 83 on page 138	N/A	N/A	N/A	N/A
44	2C	CDS List Structure: See Table 86 on page 141	N/A	N/A	N/A	N/A
48	30	N/A	N/A	N/A	N/A	N/A
Reg. 15 Return Code	0,4	0,4	0	0	0	0

Begin function (function code X'FE')

The begin function of the directory services management exit can be activated by VTAM initialization or by issuing a MODIFY EXIT,ID=ISTEXCDM,OPTION=ACT command. See [z/OS Communications Server: SNA Operation](#) for more information about this command.

The begin function is required if you write this exit routine; all other functions are optional. This function is processed during exit initialization before any of the other functions are processed. During the begin function processing, the exit selects the other functions to process. The exit is invoked for only those functions specified by your routine.

The address of the exit options is passed to the exit routine only during begin processing. The exit options field indicates the functions that the exit routine supports. This field is modified by the exit routine during processing of the begin function.

The address of the border node and interchange node options is also passed to the exit routine during begin processing. The border node and interchange node options field provides control information relating to the border node and interchange node selection functions of the exit routine. Bit definitions are described in [Table 77 on page 134](#).

If errors occur while processing the begin function, VTAM deactivates the exit and no functions are supported. The APPN network continues as if no exit routine exists. All search requests are authorized until the exit is reactivated.

The exit returns to VTAM with a return code:

Register 15:

Return code

X'00' (0)

Processing successfully completed

X'01'–X'4F' (1–79)

Exit processing did not work; the exit remains active.

X'50'–X'7F' (80–127)

Reserved

Greater than X'7F' (127)

Invocation failed; exit deleted and deactivated

X'F0'–X'FF' (240–255)

Reserved

If the exit return code is greater than X'7F', VTAM ignores any options you specified and issues error message IST985I.

Initial authorization function (function code X'00')

The initial authorization function allows you to either reject or limit the scope of a search. For example, you might use initial authorization to reject a search based on a search's originating logical unit (OLU) or its destination logical unit (DLU) because the OLU or the DLU is not authorized to use the network.

Or, you might use initial authorization to limit the scope of a search to the domain of a node, because a resource resides in that node's domain, and further searching would be pointless.

The function provided by this exit is applicable only for new search requests with:

- Application general data stream (GDS) variable CDINIT
- Application GDS variable INIT_OTHER_CD
- No application GDS variable

These types of Application GDS variables generally flow when the search request will result in a session setup (CDINIT, IOCDINIT) or when the search is initiated by network management (no application GDS variables).

Table Table 73 on page 121 correlates exit return codes and exit sense codes with the search steps performed by directory services following initial authorization. A copy of the application GDS variable is supplied to the exit if application GDS variable support is indicated by the exit during begin processing.

Table 73. Exit return codes, exit sense codes, and VTAM search steps							
Exit return code	EXIT	Directory services search steps performed					
	Sense code	Domain search	APPN directed search	APPN network broadcast	Subarea search of this node	Subarea search of other nodes	Rejected search
0	X'00000000'	X	X	X	X	X	
4	X'080A000D'						X
8	X'00000000'	X					
12	X'00000000'	X			X		
16	X'00000000'	X	X	X		X	
20	X'00000000'	X	X	X	X		
24	X'00000000'	X	X	X			

Table 73. Exit return codes, exit sense codes, and VTAM search steps (continued)							
Exit return code	EXIT	Directory services search steps performed					
	Sense code	Domain search	APPN directed search	APPN network broadcast	Subarea search of this node	Subarea search of other nodes	Rejected search
28	X'00000000'	X			X	X	
32	X'00000000'	Directed searches only					
36	X'00000000'	Use search task list. See Table 84 on page 138 .					
40	X'00000000'	Fail this directed search only. Valid only on redrive.					
None of the above	X'080A000D'						X

If the user routine returns a return code corresponding to the None of the above entry in [Table 73 on page 121](#), VTAM issues message IST1183I.

Although the exit can limit the scope of a search performed by VTAM, it is not allowed to expand the scope. For example, if VTAM receives a request that does not allow the subarea to be searched, but the exit indicates that a subarea search is allowed, a subarea search is not performed.

The exit can limit the scope of the search performed by VTAM by using return code 36 and turning off individual search tasks in the Search Task List. The exit is allowed to turn off any of the tasks in the list but is not allowed to turn on any task that is not already turned on. Any additional search tasks that the exit attempts to add will be ignored. [Table 84 on page 138](#) explains the individual search task that can be controlled by the exit.

Guideline: Turning off search tasks, using the Search Task List, without fully understanding the network configuration and the dynamics of APPN searching can result in failed searches or other unexpected results. This function should be used with extreme caution.

When the exit is called for initial authorization with an alias DLU NETID value, the exit can indicate the real NETID value on the reply. If the exit determines that the NETID value in the CV82 vector is the real value, the exit can set the NETID authentic bit in the search task options to indicate that the NETID value is the real value. If the exit determines that a different NETID value should be used as the real NETID value, the exit can rebuild the DLU FIND control vector 82 with the real NETID value and set the NETID authentic bit in the search task options, indicating that the NETID value is the real value. The name cannot be changed; only the NETID value can be changed. If the NETID field length changes, the CV82 length must also be changed to match. Padding was added to the DLU area to accommodate the expansion of the CV82 vector length to a maximum 17 characters. Only the CV82 vector can be changed.

Redrive of exit for search authorization (function code X'00' with redrive bit on)

The exit can be redriven for authorization after the initial authorization call to reject the search or limit the scope of the search. When the exit is called for initial authorization, the DLU information might not be complete or accurate. Redriving the exit for authorization enables the exit to verify the DLU information before a directed search that could result in a session establishment is sent.

The exit indicates that it is to be redriven by setting the redrive bit in the search task options and setting return code 36 on the initial authorization call. If directory services management obtains DLU information that results in the sending of a directed search, the exit is redriven for authorization before the directed search is sent. When the exit is redriven, the initial authorization parameter list is used with the updated DLU information and the redrive bit is set to the value on to indicate that it is a redrive of the authorization function. When the NETID value is an alias and a border node is searching other networks, the DLU NETID value is updated with the NETID value of the adjacent network that is being searched. The exit cannot change the NETID value from the alias value to the real value on a redrive call.

When redriven for authorization, the exit has three options:

- Authorize this search
- Do not authorize this directed search but continue searching
- Reject this search

Setting the return code to 0 authorizes the search; if the directed search fails, then searching continues. Setting the return code to 40 fails the current directed search but searching continues. For return code 0 or 40, the exit can set the redrive bit to be redriven for any additional directed searches. Setting the return code to 4 rejects the search, terminating this locate request.

Border node selection function (function code X'01')

This function is optional. If processed, when VTAM builds a subnetwork routing list for sending a directed locate search to a border node or nonnative network node, VTAM forwards a copy of the subnetwork routing list to the exit.

The user exit can change the subnetwork routing list by:

- Removing entries from the list
- Reordering the entries in the list
- Adding entries to the list
- Replacing entries in the list

Modifying the subnetwork routing list might involve the physical movement of data within the storage. For example, to remove an entry from the list, the entries following the deleted entry must be moved to be contiguous with the entry before the deleted entry, and the *number-of-entries* count must be decremented by one.

In addition to modifying the subnetwork routing list, the exit can indicate that no border node search procedure should be attempted.

The user-written exit returns to VTAM with a return code:

Register 15:

Return code

X'00' (0)

Border node searching is allowed using the subnetwork routing list returned by the exit.

X'04' (4)

Border node selection was not performed. The original subnetwork routing list should be used.

X'08' (8)

Border node searching is not allowed.

All other return codes are treated as return code X'04'.

If the exit return code is greater than X'7F', VTAM ignores any options that you specified and issues error message IST985I.

Interchange node selection function (function code X'02')

This function is optional. If this function is processed, then when VTAM builds an interchange node list for sending directed locate searches to interchange nodes, VTAM forwards a copy of the interchange node list to the exit.

The user exit can change the interchange node list by using one of the following methods:

- Removing entries from the list
- Reordering the entries in the list

Modifying the interchange node list might involve the physical movement of data within storage. For example, to remove an entry from the list, the entries following the deleted entry must be moved

to be contiguous with the entry before the deleted entry, and the number-of-entries count must be decremented by 1. If the exit returns an empty list, the return code is set to X'04', indicating that no interchange node search procedure should be attempted. If the exit returns an interchange node that was not in the original list, diagnostic code X'27' is issued by message IST1183I, and the return code is set to X'08'.

In addition to modifying the interchange node list, the exit can indicate that no interchange node search procedure should be attempted.

The user-written exit returns to VTAM with the following return code:

Register 15:

Return code

X'00' (0)

Interchange node searching is allowed using the interchange node list returned by the exit.

X'04' (4)

Interchange node selection was not performed. The original interchange node list should be used.

X'08' (8)

Interchange node searching is not allowed.

All other return codes are treated as return code X'04'.

If the exit return code is greater than X'7F', VTAM ignores any options you specified and issues error message IST985I.

CDS selection function (function code X'03')

The CDS selection function is invoked at a network node before it refers a Locate search request to a central directory server (CDS). The exit receives an ordered list of all known CDSs in the network. The first entry in the list is the CDS that VTAM uses for the first search request, followed by the remaining CDSs within the range of the CDSREFER start option, followed by the remaining CDSs in the network. The exit has the following options:

- Approve the CDS list as received
- Reorder the CDS list

For any given Locate search, VTAM uses the first CDS specified in the list. If the request to the selected CDS fails, VTAM tries the request again by using the second CDS returned in the list. The remaining CDSs are not used for that request.

VTAM ensures that at least one entry is returned by the exit. The following conditions in the returned CDS list are not valid and result in VTAM issuing a FFST probe, deactivating the function, and continuing with the original list:

- No entries are returned in the list.
- Entries are added that are not in the original list.
- Duplicate entries are found in the list.
- Entries are found that contain corrupted data.

The user-written exit returns to VTAM with a return code:

Register 15:

Return code

X'00' (0)

The CDS list has been modified.

X'04' (4)

The CDS list has not been modified.

An unexpected return code is treated as return code X'04', but message IST1183I is issued to record the unexpected return code.

Alternate CDS selection function (function code X'04')

The alternate CDS selection function is invoked at a CDS before it forwards a Locate search to any alternate CDSs in the network. An alternate CDS is a central directory server that is used to receive Locate search requests when a search of the database of another CDS fails.

When VTAM passes the CDS list to the exit, the first entry in the list identifies the CDS with the least weighted path. The remainder of the list continues in least weighted order for all other known CDSs in the network. The exit can approve the list as received, reorder the list, or remove entries from the list. If no entries are returned in the CDS list, VTAM does not forward the Locate request to any alternate CDS and, instead, perform a broadcast search for the resource. If entries are returned in the list, VTAM searches the alternate CDSs in the order specified in the CDS list. The search terminates when a CDS in the list returns information about the resource or all CDSs in the list have been searched.

The following conditions in the returned CDS list are considered not valid and result in VTAM issuing a FFST probe, deactivating the function, and continuing with the original list:

- Entries are added that are not in the original list.
- Duplicate entries are found in the list.
- Entries are found that contain corrupted data.

The user-written exit returns to VTAM with a return code:

Register 15:

Return code

X'00' (0)

The CDS list has been modified.

X'04' (4)

The CDS list has not been modified.

An unexpected return code is treated as return code X'04', but message IST1183I is issued to record the unexpected return code.

Central resource registration selection function (function code X'05')

The central resource registration selection function is invoked at a network node before it sends a central resource registration request to a CDS. When VTAM passes the CDS list to the exit, the first entry in the list identifies the CDS with the least weighted path. The remainder of the list continues in least weighted order for all other known CDSs in the network. The exit can approve the list as received or reorder the list. VTAM sends all central resource registration requests to the first CDS returned in the list and any other CDS entries returned are not used. VTAM continues to use this CDS for registration requests until a topology change occurs in the network involving one or more CDSs. When a change occurs and more than one CDS exists in the network, the exit is reinvoked to select the CDS to use.

The following conditions in the returned CDS list are not valid and result in VTAM issuing a FFST probe, deactivating the function, and continuing with the original list.

- No entries are returned in the list.
- Entries are added that are not in the original list.
- Duplicate entries are found in the list.
- Entries are found that contain corrupted data.

No search-related information is passed to the exit for central resource registration selection.

The user-written exit returns to VTAM with a return code:

Register 15:

Return code

X'00' (0)

The CDS list has been modified.

X'04' (4)

The CDS list has not been modified.

An unexpected return code is treated as return code X'04', but message IST1183I is issued to record the unexpected return code.

Replacing function (function code X'08')

The optional replacing function can be useful for cleanup. For example, you might want to do some cleanup based on your table maintenance or based on the status of your data sets. Possibly you need to do some cleanup based on checkpoints you are tracking, or storage you have freed.

The installation-wide exit currently in use can be replaced by issuing a `MODIFY EXIT,ID=ISTEXCDM,OPTION=REPL` command. See [z/OS Communications Server: SNA Operation](#) for more information about this command. You can replace the exit, but not the exit options. If you want to replace the exit options, deactivate the exit, then reactivate it. If the exit is active when you issue the `MODIFY EXIT` command with `OPTION=REPL` specified, the exit is allowed to complete processing before it is driven for the replaced function.

For an exit that is being replaced, VTAM sets the function code to X'08', builds a parameter list, and calls the exit to indicate to the exit that it is being driven the last time before VTAM replaces it.

Replaced function (function code X'09')

This optional function indicates that the previous directory services management exit routine was replaced by the current routine through a `MODIFY EXIT,OPTION=REPL` command. For more information about replacing exits, see [“Replacing VTAM exit routines” on page 175](#). For more information about this command, see [z/OS Communications Server: SNA Operation](#).

When an exit is replaced, VTAM sets the function code to X'09', builds a parameter list, and calls the exit to indicate to the exit that it is being driven the first time after being reloaded.

End function (function code X'FF')

This optional function, like the optional replacing function, provides an opportunity for cleanup. For example, you might want to call the end function to free storage obtained from the exit.

The end function is processed when VTAM is halted (`HALT` or `HALT QUICK`) or when you issue a `MODIFY EXIT,ID=ISTEXCDM,OPTION=INACT` command. If the exit is active when you attempt to halt it with the `MODIFY EXIT` command, the exit is allowed to complete processing before it is driven for the end function.

For an exit that is being ended, VTAM sets the function code to X'FF', builds a parameter list, and calls the exit to indicate to the exit that it is being ended.

VTAM resets all functions designated by the begin function.

Parameter descriptions

Parameter lists representing individual functions are passed to the user's exit routine. All parameter lists are summarized in [Table 71 on page 118](#) and [Table 72 on page 119](#). The parameters in each list are described in the following topics in the order they appear in the parameter lists.

Note: The control vectors described in the following topics are in length key format.

Environment vector list

The environment vector list provides information specific to the host in which the exit is operating (see [Table 74 on page 127](#)). The information includes the network ID, the SSCP name, and other information that identifies the environment in which the exit routine is operating. The vector list is preceded by a 2-byte header field.

Table 74. Environment vectors

Dec (Hex) offset	Size (Bytes)	Description
A header precedes the list of vectors. This header is followed by several vectors that can be in any order:		
0 (0)	2	Total length of the parameter list including this length field (m)
The network identification vector:		
0 (0)	1	Length of the vector including this field (n)
1 (1)	1	ID of the vector (X'06')
2 (2)	n-2	ID of the network in which this exit routine is operating (from the NETID start option)
Control point name vector:		
0 (0)	1	Length of the vector including this field (n)
1 (1)	1	ID of the vector (X'07')
2 (2)	n-2	SSCP name of the host in which this exit is operating. The start option SSCPNAME is required.
Node characteristics vector:		
0 (0)	1	Length of the vector including this field (n)
1 (1)	1	ID of the vector (X'0C')
2 (2)	n-2	Node characteristics:
Bit Value		
B'1.....'.		
This node is a network node		
B'0.....'.		
This node is not a network node		
B'.1.....'.		
This node is an end node		
B'.0.....'.		
This node is not an end node		
B'..1.....'.		
This node is an interchange network node		
B'..0.....'.		
This node is a pure network node		
B'...1....'.		
Reserved		
B'...0....'.		
Reserved		
B'....1...'.		
This node is a central directory server		
B'....0...'.		
This node is not a central directory server		
B'.....1..'.		
This node is a border node		
B'.....0..'.		
This node is not a border node		

Function code and related search information

The function codes determine which parameters VTAM passes to the directory services management exit routine. In addition to function codes, VTAM passes search-specific information to the exit routine.

Table 75 on page 128 describes:

- The function codes that VTAM passes to the installation-wide exit routine
- Additional search-specific information

Table 75. Function code and related search information

Byte	Description
0	Function code: X'00' The exit routine has been called for initial search authorization. X'01' The exit has been called for the border node selection function. X'02' The exit has been called for the interchange node selection function. X'03' The exit has been called for the CDS selection function. X'04' The exit has been called for the alternate CDS selection function. X'05' The exit has been called for the central resource registration selection function. X'08' The exit is being driven the last time before being replaced; an operator issued the MODIFY EXIT command with OPTION=REPL specified. X'09' The exit was replaced and is being driven the first time after being reloaded. X'FE' The exit routine has been called to select the functions it will process. This begin function is processed only once during VTAM initialization. X'FF' One of two conditions exists: <ul style="list-style-type: none">• An operator has issued the MODIFY EXIT...OPTION=INACT command; this ends the installation-wide exit. Or,• The exit routine has been called to perform required cleanup during VTAM termination. This end function is processed only once during VTAM termination.

Table 75. Function code and related search information (continued)

Byte	Description
1	<p>Related search information:</p> <p>This 3-bit network ID authenticity indicator indicates one of the following:</p> <p>B'0...' The CP(OLU) cannot accept a different name for the DLU on the response than what was specified on the request.</p> <p>B'1...' The CP(OLU) will accept a different name for the DLU on the response than what was specified on the request.</p> <p>B'.0..' The resource name represents a <i>generic</i> name (this might not be the real name of the resource).</p> <p>B'.1..' The resource name is the real name of the resource.</p> <p>B'..0.' The resource network ID is the real network ID of the resource.</p> <p>B'..1.' The resource network ID might not be the real network ID of the resource.</p>
2	<p>Related search information:</p> <p>This 1-byte field specifies the number of hops that a broadcast search can traverse before terminating.</p>

Table 75. Function code and related search information (continued)

Byte	Description
3	<p>Related search information</p> <p>The following bit definitions indicate the role of this node:</p> <p>Bit value</p> <p>B'1...' This node is a CP(OLU)</p> <p>B'0...' This node is not a CP(OLU)</p> <p>B'.1..' This node is a CP(DLU)</p> <p>B'.0..' This node is not a CP(DLU)</p> <p>B'..1.' This node is a network node(OLU)</p> <p>B'..0.' This node is not a network node(OLU)</p> <p>B'...1' This node is a network node(DLU)</p> <p>B'...0' This node is not a network node(DLU)</p> <p>B'.... 1...' This node is an origin directory server</p> <p>B'.... 0...' This node is not an origin directory server</p> <p>B'.... .1..' This node is an alternate directory server</p> <p>B'.... .0..' This node is not an alternate directory server</p> <p>B'.... ..1.' This node processes an interchange node search</p> <p>B'.... ..0.' This node does not process an interchange node search</p> <p>B'.... ...1' This node is an intermediate network node on broadcast</p> <p>B'.... ...0' This node is not an intermediate network node on broadcast</p>

Table 75. Function code and related search information (continued)

Byte	Description
4	<p>Search origin information</p> <p>The following bit definitions indicate the source of the current search:</p> <p>B'1...' The search originated from the subarea network</p> <p>B'0...' The search did not originate from the subarea network</p> <p>B'.1..' The search was received over an APPN subnetwork link</p> <p>B'.0..' The search was not received over an APPN subnetwork link</p> <p>B'..1.' The search was received from an end node</p> <p>B'..0.' The search was not received from an end node</p> <p>B'...1' The search was received from a network node</p> <p>B'...0' The search was not received from a network node</p> <p>B'.... 1...' The search was received from a border node</p> <p>B'.... 0...' The search was not received from a border node</p>
5	<p>Search type information</p> <p>The following bit definitions indicate the purpose of the search:</p> <p>B'1...' Search is a resource discovery search</p> <p>B'0...' Search is not a resource discovery search</p> <p>B'.001' Application GDS variable is CDINIT</p> <p>B'.010' Application GDS variable is IOCD</p> <p>B'.011' Application GDS variable is Notify</p> <p>B'.101' No Application GDS variable present</p> <p>B'.... 1...' This is a PLU-initiated search that was sent on behalf of a SLU-initiated session</p> <p>B'.... 0...' This is not a PLU-initiated search that was sent on behalf of a SLU-initiated session</p>

User data field

For the entry codes described in [Table 71 on page 118](#) and [Table 72 on page 119](#), the directory services management exit routine can use the 8-byte user data field, initialized to 0, for any purpose. For example, the exit could use this field to store the address of a storage area obtained dynamically. VTAM saves the contents of this user field after every successful invocation of the directory services management exit routine. Thus, the contents are available to the exit routine the next time the routine is called. This field is cleared to 0 when the directory services management exit becomes inactive.

The last 4 bytes can be used to uniquely identify the level or version of an exit. The last 4 bytes are displayed following subsequent use of the DISPLAY EXIT command. See [z/OS Communications Server: SNA Operation](#) for information about this command.

User-supplied parameter data

The exit routine can use this 4-byte user data field, initialized to zero, for any purpose. For example, the exit can use this field to store the address of a storage area obtained dynamically.

VTAM saves the contents of this user field after every successful invocation of the installation-wide exit routine; VTAM then passes the contents back to the user-written routine the next time it is called.

Exit options

The exit options field is a 2-byte field that indicates the functions for which VTAM will call the exit routine. This field is modified by the exit routine during begin function processing. The bit definitions are described in [Table 76 on page 133](#).

Table 76. Exit options

Dec (Hex) offset	Size (Bytes)	Description
0 (0)	1	<p>Functional Support Flags</p> <p>B'1...' The exit routine processes the initial authorization function (function code X'00').</p> <p>B'0...' The exit routine does not process the initial authorization function (function code X'00'). The default is 0.</p> <p>B'1...' The exit routine processes the interchange node selection function (function code X'02').</p> <p>B'0...' The exit routine does not process the interchange node selection function (function code X'02'). The default is 0.</p> <p>B'...1' The exit routine processes the border node selection function (function code X'01').</p> <p>B'...0' The exit routine does not process the border node selection function (function code X'01'). The default is 0.</p> <p>B'.... 1...' The exit routine processes the end function (function code X'FF').</p> <p>B'.... 0...' The exit routine does not process the end function (function code X'FF'). The default is 0.</p> <p>B'.... .1..' The exit routine processes the CDS selection function (function code X'03').</p> <p>B'.... .0..' The exit routine does not process the CDS selection function (function code X'03'). The default is 0.</p> <p>B'.... ..1.' The exit routine processes the alternate CDS selection function (function code X'04').</p> <p>B'.... ..0.' The exit routine does not process the alternate CDS selection function (function code X'04'). The default is 0.</p> <p>B'.... ...1' The exit routine processes the central resource registration function (function code X'05').</p> <p>B'.... ...0' The exit routine does not process the central resource registration selection function (function code X'05'). The default is 0.</p>

Table 76. Exit options (continued)

Dec (Hex) offset	Size (Bytes)	Description
1 (1)	1	<p>Functional support flags</p> <p>B'1...' The exit routine supports application GDS processing.</p> <p>B'0...' The exit routine does not support application GDS processing. The default is 0.</p> <p>B'.1...' The exit routine requests caching of local DLU resources.</p> <p>B'.0..' The exit routine does not request caching of local DLU resources.</p> <p>B'.... ..0.' The exit routine does not support MODIFY EXIT processing. The default is 0.</p> <p>B'.... ..1.' The exit routine supports MODIFY EXIT processing.</p> <p>Bits that are not shown are reserved.</p>

Tip: When caching local resources is enabled, directory services caches local DLU resources when they are found during locate search processing. This results in a directory services database storage increase of 218 bytes for each local resource found as the DLU of a search.

Border node and interchange node options

The border node and interchange node options field is a 2-byte field that includes certain control information that is used for the border node selection function and the interchange node selection function of the directory services management exit. This structure is valid only if one of the following is true:

- The node is defined as a border node and the border node selection function of the directory services management exit is allowed.
- The node is defined as a network node and the interchange node selection function of the directory services management exit is allowed.

Table 77. Border node and interchange node options

Dec (Hex) offset	Size (Bytes)	Description
0 (0)	1	Specifies the maximum number of node additions allowed during border node selection. A value from 0 (X'0') to 255 (X'FF') can be specified.

Table 77. Border node and interchange node options (continued)

Dec (Hex) offset	Size (Bytes)	Description
1 (1)	1	<p>Border node selection exit invocation flags</p> <p>B'1...' A search can be sent to a non-native node without invoking the border node selection function if the search request is due to existing database information.</p> <p>B'0...' A search cannot be sent to a non-native node without invoking the border node selection function even if the search request is due to existing database information.</p> <p>B'.1..' A search can be sent for a cross-domain subarea resource without invoking the border node selection function or the interchange node function. This is the default.</p> <p>B'.0..' A search cannot be sent for a cross-domain subarea resource without invoking the border node selection function or the interchange node function.</p> <p>The remaining bits are reserved.</p>

Interchange node list structure

The interchange node list is part of the interchange node selection function and has the structure shown in [Table 78 on page 135](#).

Table 78. Interchange node list structure

Dec (Hex) offset	Size (Bytes)	Description
0 (0)	4	ICN list structure ID (ICNL)
4 (4)	2	Length of each entry in the interchange node list
6 (6)	2	Total number of entries in the interchange node list
8 (8)	$n-8$	ICN list entries (where n is equal to the total length of the structure and 8 is the header length)
The header is followed by the entries in the interchange node list, each of which has the following structure:		
Interchange node list entry structure		
0 (0)	8	Network ID of the node
8 (8)	8	Name of the node
16 (10)	1	Length of the network ID
17 (11)	1	Length of the name

OLU information structure

The statements in the OLU information structure in [Table 79 on page 136](#) are required unless indicated otherwise.

Table 79. OLU information structure

Dec (Hex) offset	Size (Bytes)	Description
A header precedes the list of vectors. This header is followed by several vectors that can be in any order:		
0 (0)	1	Total length of the vector list including this length field
Optional network node server control vector X'3C' (associated resource entry). Provides information about the network node server:		
0 (0)	1	Length of the vector including this field
1 (1)	1	ID of the vector (X'3C')
2 (2)	2	Type of resource
4 (4)	n-4	Name of resource
Optional CP control vector X'3C' (associated resource entry). Provides information about the CP:		
0 (0)	1	Length of the vector including this field
1 (1)	1	ID of the vector (X'3C')
2 (2)	2	Type of resource
4 (4)	n-4	Name of resource
OLU control vector X'3D'. Provides information about the OLU:		
0 (0)	1	Length of the vector including this field
1 (1)	1	ID of the vector (X'3D')
2 (2)	2	Type of resource
4 (4)	n-4	Name of resource

DLU information structure

The statements in the following DLU information structure are required unless indicated otherwise. This structure contains the following information:

Table 80. DLU information structure

Dec (Hex) offset	Size (Bytes)	Description
A header precedes the list of vectors. This header is followed by several vectors that can be in any order:		

Table 80. DLU information structure (continued)

Dec (Hex) offset	Size (Bytes)	Description
0 (0)	1	Total length of the vector list including this length field

Optional network node server (DLU) find control vector X'81' (associated resource entry). Provides information about the network node server(DLU):

0 (0)	1	Length of the vector including this field
1 (1)	1	ID of the vector (X'81')
2 (2)	2	Type of resource
4 (4)	n-4	Name of resource

Optional CP(DLU) find control vector X'81' (search argument associated resource). Provides information about the CP(DLU):

0 (0)	1	Length of the vector including this field
1 (1)	1	ID of the vector (X'81')
2 (2)	2	Type of resource
4 (4)	n-4	Name of resource

DLU find control vector X'82' (search argument directory entry). Provides information about the DLU:

0 (0)	1	Length of the vector including this field
1 (1)	1	ID of the vector (X'82')
2 (2)	2	Type of resource
4 (4)	n-4	Name of resource

Network-qualified adjacent CP name vector

This control vector contains the following information:

Table 81. Network-qualified adjacent CP name vector

Dec (Hex) offset	Size (Bytes)	Description
0 (0)	1	Total length of the vector list including this length field
1 (1)	n-1	If the search was originated by the host node, then this field contains the host CP name. Otherwise, this field contains the network-qualified CP name of the adjacent node from which the search was received.

Search correlator structure

This structure contains the following information:

Table 82. Search correlator structure

Dec (Hex) offset	Size (Bytes)	Description
0 (0)	1	Total length of the structure including this length field
1 (1)	1	Length of the vector list including this length field
2 (2)	1	ID of the vector X'60'
3 (3)	8	Procedure correlation ID
11 (B)	1	Length of the network-qualified CP name
12 (C)	n-12	Network-qualified CP name

PCID modifier structure

This structure contains the following information:

Table 83. PCID modifier structure

Dec (Hex) offset	Size (Bytes)	Description
PCID modifier vector:		
0 (0)	1	Total length of the structure including this length field
1 (1)	1	Length of the vector including this length field
2 (2)	1	ID of the vector (X'81')
		Provides information about the PCID modifier
3 (3)	2	Procedure resubmit number
5 (5)	1	Last significant half byte
6 (6)	n-6	Modifier data

Search task list

This structure contains the following information:

Table 84. Search task list

Byte	Bit value	Description
0	B'1...'	Domain broadcast: Domain broadcast to served end nodes that allow broadcast searching. This includes a border node that acts as an end node.
	B'.1..'	Network broadcast: Network broadcast to all adjacent nodes.
	B'..1.'	Directed to CDS: Directed search from network node to a central directory server (CDS).
	B'...1'	Directed to alternate directory server: A directed search from a CDS to an alternate CDS.

Table 84. Search task list (continued)

Byte	Bit value	Description
1	B'.... 1...'	Forward network broadcast: Forward a network broadcast that was received from a network node to all other adjacent network nodes.
	B'.... .1..'	Forward NNS: Forward a search from the subarea side of an end node to the network node server.
	B'.... ..1.'	Reserved.
	B'....1'	Reserved.
	B'1...'	Border node directed: Directed search from a border node to a nonnative node when the DLU location is known.
	B'.1..'	Border Node Broadcast: Serial search from a border node to all nodes in the subnet routing list when the DLU location is not known.
	B'.1.'	Reserved.
	B'...1'	Reserved.
	B'.... 1...'	Local Subarea: Search the subarea side of this node. This includes all resources owned by this node.
	B'.... .1..'	Subarea Network: Search the subarea network attached to this node. (The bit allowing search of the subarea side of this node must also be on for this search task.)
	B'.... ..1.'	Serial ICN Search: Search subarea networks attached to other interchange nodes in this APPN subnet.
	B'....1'	Reserved.
2	B'1...'	Redrive for authorization.
	B'0...'	Do not redrive for authorization.
	B'.1..'	Set NETID as authentic; this is set by the exit.
	B'.0..'	NETID setting remains unchanged.

Notes:

- When the exit is on an end node, only Forward NNS and Local Subarea search tasks apply.
- When the exit is called for initial authorization, the redrive for authorization bit will be set to off. The exit can enable the bit to request that the exit be redriven for authorization. When the exit is redriven, the bit is enabled. On the redrive call, the exit can set the redrive bit to on when return codes 0 or 40 are set, to be redriven for any additional directed searches.

Subnetwork routing list structure

The subnetwork routing list is part of the border node selection function and has the following structure:

Table 85. Subnetwork routing list structure

Dec (Hex) offset	Size (Bytes)	Description
---------------------	-----------------	-------------

A header precedes the list of entries:

Table 85. Subnetwork routing list structure (continued)

Dec (Hex) offset	Size (Bytes)	Description
0 (0)	4	Structure identifier
4 (4)	2	Length of the header, including the structure identifier and this length field
6 (6)	2	Length of each entry in the subnetwork routing list
8 (8)	2	Number of total entries allowed in the subnetwork routing list
10 (A)	2	Number of initialized entries in the subnetwork routing list

The header is followed by the entries in the subnetwork routing list. Each entry has the following structure:

Subnetwork routing list entry structure

0 (0)	8	Network ID of the node
8 (8)	8	Name of the node
16 (10)	1	Length of the actual network ID
17 (11)	1	Length of the actual name
18 (12)	1	Subnetwork visit count associated with this node
19 (13)	1	Reserved
20 (14)	1	Node information flags

Bit Value

B'1...'

The node is a border node.

B'0...'

The node is a network node.

B'.1..'

The node is adjacent to this node.

B'.0..'

The node is not adjacent to this node.

B'..1.'

The node is not in the same subnetwork as this node. The node is nonnative.

B'..0.'

The node is in the same subnetwork as this node. The node is native.

Table 85. Subnetwork routing list structure (continued)

Dec (Hex) offset	Size (Bytes)	Description
21 (15)	1	Entry information flags Bit value B'000.' The entry was defined using an adjacent cluster routing table definition B'001.' The entry was added by VTAM due to search processing results B'010.' The entry was added by VTAM because DYNAMICS=FULL was specified on the adjacent cluster routing table used for this subnetwork search B'011.' The entry was added by VTAM because DYNAMICS=LIMITED was specified (or defaulted) for the adjacent cluster routing table used for this subnetwork search B'101.' The entry was added by VTAM due to directory database information B'...0 0...' The entry has not been used previously for a search of this network ID B'...0 1...' The previous cross-subnetwork search for this network ID that used this entry was not successful B'...1 0...' The previous cross-subnetwork search for this network ID that used this entry was successful

CDS list structure

This structure contains the following information:

Table 86. Central directory server list

Comments	Dec (Hex) offset	Size (Bytes)	Description
	0 (0)	2	Length of each CDS entry in this list.
	2 (2)	2	Number of CDS entries in this list.
	4 (4)	*	CDS list entries
Each entry in the list is mapped as follows			
	0 (0)	8	Network ID of CDS
	8 (8)	8	Unqualified CP name of CDS
	16 (10)	1	Length of network identifier of CDS
	17 (11)	1	Length of CP name

Generic resource resolution exit routine

The generic resource resolution exit enables you to change the results of the generic resource resolution. When VTAM receives a session initiation request that specifies a generic resource name as a DLU name, the exit provides the values for the application program network names of the generic resource members.

VTAM either initializes the generic resource resolution exit routine during VTAM initialization, or activates the routine after you issue a `MODIFY EXIT,ID=exitname,OPTION=ACT|REPL` command.

You can use the `DISPLAY EXIT` command to display information about a generic resource resolution exit. You also can use a `MODIFY EXIT` command to activate, deactivate, or replace the exit routine without interrupting VTAM processing. See “Operator commands for VTAM exit routines” on [page 174](#) for more information about using the `MODIFY EXIT` command to modify VTAM exit routines. The output from the `DISPLAY GRPREFS` and `DISPLAY ID=generic resource name` commands indicate whether the generic resource exit is called for a specific generic resource based on the generic resource preferences. See the generic resource preference table in [z/OS Communications Server: SNA Resource Definition Reference](#) for more information about the `MODIFY EXIT`, `DISPLAY EXIT`, `DISPLAY GRPREFS`, and `DISPLAY ID` commands.

VTAM loads the generic resource resolution exit into fixed VTAM private storage and marks the exit active during initialization. The primary function of the exit routine is generic resource resolution. However, VTAM also calls the exit for exit activation and deactivation. By default, VTAM does not call the generic resource exit during resolution. For the generic resource exit to be called during resolution, the exit must be active and the generic resource preferences for the generic name being resolved must be defined to call the generic resource exit. See the generic resource preference table information in [z/OS Communications Server: SNA Resource Definition Reference](#) for more information about defining generic resource preferences.

When the generic resource needs to be resolved, VTAM builds the generic resource resolution exit parameter list and drives the exit routine. The parameter list is described in “Generic resource resolution exit routine parameter list” on [page 144](#).

When a control point (CP) in a sysplex receives a session initiation request that specifies a generic resource name, and the generic resource resolution exit routine is active, VTAM calls the exit routine for resolution.

If you want the function of the generic resource resolution exit, you must write the exit routine and link-edit it into the appropriate library. (See “Installing VTAM exit routines” on [page 173](#) for more information.)

Tip: IBM supplies a sample generic resource resolution exit routine that can be used for the generic resources function. The sample routine is stored in `SYS1.SAMPLIB` under the VTAM module name `ISTEXCGR`. You can use the sample as it is coded or you can modify the sample to meet the needs of your installation.

The following information is information that you need to write a generic resource resolution exit routine.

Initial register contents

When VTAM passes control to the generic resource resolution exit routine, the register contents are as follows:

Register 0:

Entry code:

X'04'

Invoke processing (see “Invoke flags” on [page 145](#))

X'10'

Exit activation

X'18'

Exit replacement

X'20'

Exit deactivation

X'40'

VTAM termination

Register 1:

Address of the generic resource resolution exit parameter list described in [“Generic resource resolution exit routine parameter list” on page 144](#)

Register 13:

Address of a standard 72-byte save area

Register 14:

Return address

Register 15:

Address of the generic resource resolution exit's entry point

Final register contents

The exit routine must leave the register status as follows:

Register 1:

Address of the generic resource resolution exit parameter list with modified values

Registers 2–14:

Restored to entry contents

Register 15:

Return code

X'0'

Processing successfully completed

X'20'

Resolution has failed as a result of a storage shortage

X'3A'

Resolution has failed as a result of an invoke flag that is not valid

X'3C'

Resolution has failed as a result of an entry code that is not valid

X'40'

Indicate to VTAM to fail this session with sense code 0801002E (generic resource is not available).

Design requirements

When writing the routine, use standard linkage and save and restore registers 2-14.

Errors in the routine could damage VTAM or system control blocks and modules. Consider the following restrictions when writing this routine:

- The name of the generic resource resolution exit routine module should be ISTEXCGR. However, if you have written an alternate load module, use the load module name that you assigned to your replacement module.
- This routine operates enabled in pageable storage. Because the routine operates at the VTAM main task dispatching priority, there is a possibility of lockout if a wait requires other task action. The routine gains control in supervisor state with a VTAM storage key. Errors within the routine could cause damage to VTAM or system control blocks and modules
- Do not issue any supervisor calls (SVCs) if this exit routine is running in service request block (SRB) mode.
- System waits, including implied waits for I/O operations, are not permitted. System waits cause VTAM failure in some situations that depend on timing.
- This exit routine should use only conditional storage invocations. You can reduce the possibility of a VTAM abend during a storage shortage by coding conditional storage invocations.

- Data is addressable in 31-bit mode only. The addresses of the data in the exit are 31-bit addresses.
- An abend of this exit routine causes VTAM to abend.
- This exit routine can reside above or below the 16-megabyte address.
- You must link-edit your routine into the appropriate library. See [“Installing VTAM exit routines”](#) on page 173 for more information.
- This exit routine operates as an internal VTAM routine. VTAM performance might be degraded if the routine requires lengthy processing time. While this routine has control, VTAM does not process any new operator requests, session initiation requests, or session termination requests for any resource.

Generic resource resolution exit routine parameter list

VTAM passes the following parameter list to the generic resource resolution exit routine in register 1, when the entry code in register 0 is X'04'.

Table 87. Generic resource resolution exit parameter list for entry code X'04'

Dec (Hex) offset	Size (Bytes)	Description
0 (0)	4	Address of an 8-byte user field
4 (4)	8	Reserved
12 (C)	4	Address of the invoke flags
16 (10)	4	Address of the generic resource parameter list

VTAM passes the following parameter list to the generic resource resolution exit routine in register 1, when the entry code in register 0 is X'10', X'18', or X'20'.

Table 88. Generic resource resolution exit parameter list for entry codes X'10', X'18', and X'20'

Dec (Hex) offset	Size (Bytes)	Description
0 (0)	4	Address of an 8-byte user field
4 (4)	8	Reserved
12 (C)	4	Address of parameter string

For an activation, deactivation, and replacement function, offset X'C' contains an address of a parameter string. The parameter string is in the form of a 2-byte length field followed by the value specified for the PARMs operand on the MODIFY EXIT command.

User data field

For the entry codes described in Table 88 on page 144 and Table 87 on page 144, the generic resource resolution exit routine can use the 8-byte user data field, originally initialized to 0, for any purpose. For example, the exit could use the field to store the address of a storage area obtained dynamically. This field is cleared to 0 when the generic resource resolution exit becomes inactive.

Use the last 4 bytes to uniquely identify an exit's level or version. The last 4 bytes are displayed following subsequent use of the DISPLAY EXIT command. See [z/OS Communications Server: SNA Operation](#) for a description of the DISPLAY EXIT command.

VTAM saves the contents of the user field after every successful invocation of the exit routine. Thus, the contents are available to the exit routine the next time the routine is called.

Invoke flags

The address of the invoke flags is passed to the exit routine in the generic resource resolution exit parameter list. The invoke flags indicate the reason that VTAM is calling the generic resource resolution exit for invoke processing. The invoke flags also determine which parameters VTAM passes to the exit. The invoke flag value that VTAM passes to this exit is X'01', which indicates that this is a request for generic resource resolution.

Generic resources parameter list

The address of the parameter list described in Table 89 on page 145 is passed to the exit routine when it is invoked for generic resource resolution. Source code for this parameter list is in ISTGREPL in SYS1.SISTMAC1.

Table 89. Input parameters for generic resource resolution

Dec (Hex) offset	Size (Bytes)	Description
0 (0)	8	Originating logical unit name.
8 (8)	8	Originating logical unit network ID.
16 (10)	8	Generic resource name (GNAME on the NIB macroinstruction).
24 (18)	8	Generic resource network ID (NETID on the NIB macroinstruction).
32 (20)	4	Address of the generic resource member list entry from VTAM.
36 (24)	4	Address of the generic resource member list entry from MVS work load manager.
40 (28)	4	Reserved
44 (2C)	4	Address of the generic resource member list.
48 (30)	1	Reserved

Generic resource member list

The address of the generic resource member list is passed to the exit when the exit is invoked for generic resource resolution. There is an entry in the list for each application program that is a generic resource member; the entry uses the specified generic resource name. The format of the list is shown in [Table 90 on page 145](#).

Table 90. Generic resource member list

Dec (Hex) offset	Size (Bytes)	Description
0 (0)	4	Reserved
4 (4)	2	Entry length
6 (6)	1	Reserved
7 (7)	1	Reserved
8 (8)	8	Application program network name.
16 (10)	8	Application program network ID value.
24 (18)	4	Reserved

Table 90. Generic resource member list (continued)

Dec (Hex) offset	Size (Bytes)	Description
28 (1C)	4	Address of the next entry in the generic resource member list. This address is 0 if there are no more entries.
32 (20)	4	Number of active sessions that have this generic resource member.
36 (24)	4	Number of LUs with generic resource session requests pending for this application program.
40 (28)	1	Length of network qualified name of the VTAM that owns the generic resource member.
41 (29)	17	Network qualified name of the VTAM that owns the generic resource member.
58 (3A)	2	Reserved
60 (3C)	4	Reserved
64 (40)	32	Reserved
(60)	4	Maximum number of sessions allowed for this application (TSO applications only)

Output parameters for generic resource resolution

The address of the parameter list described in [Table 91 on page 146](#) is passed to VTAM during generic resource resolution and the exit routine returns control to VTAM. The exit routine is expected to supply an address at X'28'.

Table 91. Output parameters for generic resource resolution

Dec (Hex) offset	Size (Bytes)	Description
0 (0)	8	Originating logical unit name.
8 (8)	8	Originating logical unit network ID value.
16 (10)	8	Generic resource name (GNAME on the NIB macroinstruction).
24 (18)	8	Generic resource network ID (NETID on the NIB macroinstruction) value.
32 (20)	4	Address of the generic resource member list entry from VTAM.
36 (24)	4	Address of the generic resource member list entry from MVS work load manager.
40 (28)	4	Address of the generic resource member list entry selected by the exit routine.
44 (2C)	4	Address of the generic resource member list (see Table 90 on page 145).
48 (30)	1	Reserved.

Generic resource resolution processing

For VTAM's generic resources function, the generic resource resolution exit routine can determine which resource will be used from among a list of application programs that are using the same generic resource name.

During exit routine processing, both VTAM and the MVS work load manager provide the name of one generic resource member to be used for session establishment. The exit routine can select one of the provided application program network names, or can continue processing user-specified criteria to select the member to be used.

The sample exit routine for generic resource resolution is stored in SYS1.SAMPLIB under the VTAM module name ISTEXCGR. If you write your own exit routine instead of using the sample, you need to follow the instructions described in [“Design requirements” on page 143](#).

VTAM invokes the exit routine for generic resource (GR) resolution only if the GR exit is active and the GR preferences table (GRPREFS) indicates that the GR exit should be called for the specified GR name. Use the GRPREFS table to define the default GR preferences, the GR preferences for individual GR names, or both.

The normal generic resource resolution process is based only on session or workload balancing. However, for a session to a generic resource from an origin logical unit (OLU) (terminal or application) that resides on the same node as an instance of the generic resource, VTAM prefers resolution to a local instance of the generic resource over instances on other nodes. In this case, the exit has a choice of selection from local resources only.

This preference occurs because there is an overhead associated with routing a session to other nodes; a local resource should not have to overcome this overhead. In many situations this local generic resource selection is preferred, but it can create a temporary or minor imbalance in session distribution.

If the OLU is a session manager application through which most of the sessions in your network are started, or if you have a large number of local LUs on the same host as the destination generic resource, then VTAM always selects a local instance of a generic resource. If you are using a single instance of a generic resource, that instance would rapidly be overloaded. By setting the appropriate generic resource preferences in the generic resource preference table, you can override the preference for picking local instances of a generic resource for either applications or local LUs. See [z/OS Communications Server: SNA Resource Definition Reference](#) for more information about defining generic resource preferences.

The generic resource exit can also be used to fail a specific session. The following process might be done, for example, if the workload on a given instance of a generic resource has reached its capacity. A session fails with the sense code X'0801002E' (generic resource is not available) if the exit returns the return code of X'40'.

Processing the sample generic resource resolution exit routine

When the sample exit routine receives a request to resolve a generic resource name, it processes the request. The sample exit routine does not select an application program network name; instead, it sets the network name field to 0.

During generic resource resolution, VTAM selects either the name chosen by the work load manager or the name chosen by VTAM through session load balancing:

- If the generic resource resolution exit is called and selects a valid resource name, this selected name is always used.
- If the exit does not select a resource name or selects an invalid resource name, and the MVS work load manager was invoked and selected a valid name, then the name chosen by the work load manager is always used.
- If neither the exit or the work load manager makes a choice, then the name that VTAM selects, based on session load balancing, is used.

Performance monitor exit routine

The performance monitor exit routine (ISTEXCPM) is available as part of the performance monitor interface for the purposes of retrieving statistical data from VTAM. Only a performance monitor application program (or monitor) using the performance monitor interface should implement this exit routine. For a description of the performance monitor interface, see [z/OS Communications Server: SNA Programming](#).

The primary function of the exit routine is to pass collection data from VTAM to the monitor. The exit routine can be invoked for one of the following conditions:

- A monitor requests the data from VTAM.
- An event occurs within VTAM that warrants sending unrequested data to each monitor.
- SMF collections are in progress and the SMF interval expires.
- An error occurs within VTAM trying to initiate data collection processing for an SMF interval that expired. In this case, no data is actually passed; the invocation is for notification purposes only.

You can use the DISPLAY EXIT command to display information regarding the performance monitor exit routine. Only performance monitors can use a MODIFY EXIT command to activate, deactivate, or replace performance monitor exits without interrupting VTAM processing. See “Operator commands for VTAM exit routines” on page 174 for more information about using the MODIFY EXIT command to modify VTAM exit routines. See [z/OS Communications Server: SNA Operation](#) for more information about coding the MODIFY EXIT and the DISPLAY EXIT commands.

When the performance monitor exit routine needs to be called, VTAM builds the performance monitor exit parameter list and drives the exit routine. The parameter list is discussed in “Performance monitor exit routine parameter list” on page 150.

The topics that follow contain information needed to write a performance monitor exit.

Multiple exit support

VTAM allows multiple instances of the performance monitor exit routine to be active concurrently.

Within this context, it is important to distinguish between the terms *exit routine* and *instance* or *exit*. The exit routine refers to only the function and interface defined by VTAM for the exit routine. An instance or exit refers to the code that is driven when the exit routine is invoked.

Instances of the exit routine can be one of the following:

- *Base exit* refers to the exit code that is activated using the MODIFY EXIT command without a qualified exit name (MODIFY EXIT,ID=ISTEXCPM).
- *Multiple exit* refers to exit code that is activated using the MODIFY EXIT command with a qualified exit name (MODIFY EXIT,ID=ISTEXCPM.*instance_name*). Use the load module name containing the exit code for *instance_name*.

The following rules apply to multiple exit support:

- The base exit or a multiple exit cannot have the same name as any other active exit. That is, no two load modules can have the same name.
- Replacing the exit with a new module automatically renames the instance to the name of the replacement module.
- When the exit routine is invoked for its primary function, all instances are driven.

If active, the base exit is always called first. The multiple instances are then called synchronously and in arbitrary order.

There is no provision for invoking a particular instance of the exit routine. For example, if there are three instances of the exit routine (P1, P2, and P3), there is no provision to call only P2. Any given invocation causes all three exits to be called.

- Only the address of the user field in the exit routine parameter list is updated by VTAM between calls to each instance.

Each instance has its own user field and each instance is called with the address of its user field in the parameter list. However, the other fields in the parameter list are not reset by VTAM between calls to each exit. As a consequence, changes to the parameter list by one instance are apparent to successive instances.

- An instance of the exit routine is not given any information about other instances nor about the order in which it is called. No formal mechanism is provided for communication among instances of the exit routine.
- Using the DISPLAY EXIT command, the base exit is displayed regardless of its state. However, multiple exits that are inactive are not displayed.

Initial register contents

When VTAM passes control to a performance monitor exit, register contents are as follows:

Register 0:

Entry code:

X'04'

Invoke processing

X'10'

Exit activation

X'18'

Exit replacement

X'20'

Exit deactivation

Register 1:

Address of the performance monitor exit parameter list described at [“Performance monitor exit routine parameter list” on page 150](#)

Register 13:

Address of a standard 72-byte save area

Register 14:

Return address

Register 15:

Address of the performance monitor exit's entry point

Final register contents

The exit must leave the register status as follows:

Registers 0–14:

Restored to entry contents

Register 15:

Return code

X'0'

Processing successfully completed.

X'8F'

Terminal error encountered; VTAM must deactivate this exit.

Design requirements

Follow these procedures when writing the exit:

- Use standard linkage.

- Save and restore registers 0–14.

Consider the following restrictions when implementing the exit:

- Each monitor should activate its exit as a multiple of the ISTECPM exit routine, not as the base exit.
- This routine operates enabled in pageable storage. Because the exit operates at VTAM's main dispatching priority, there is a possibility of lockout if a wait requires other task action. The exit gains control in supervisor state with a VTAM storage key. Errors within the exit could cause damage to VTAM or system control blocks and modules.
- This exit routine operates as an internal VTAM routine. VTAM performance might be degraded if the exit requires lengthy processing time. While this exit has control, VTAM does not process any new operator requests, session initiation requests, or session termination requests for any resource. Processing should be limited to copying any data received from VTAM into storage addressable by the monitor and notifying the monitor that the data has been delivered.
- Do not issue any SVCs if this exit is running in SRB mode.
- There can be no system waits, including implied waits for I/O operations. System waits can cause VTAM failure in some timing-dependent situations.
- This exit should use only conditional STORAGE invocations. You can reduce the possibility of a VTAM abend during a storage shortage by coding conditional STORAGE invocations.
- Data is addressable in 31-bit mode only. Data is always presented with 31-bit addressability.
- This exit can be above or below the 16M line.
- This exit must be reentrant.
- This exit should not modify the performance data parameter list (see [“Performance data parameter list” on page 151](#) for details).
- Link-edit the exit into the appropriate library. See [“Installing VTAM exit routines” on page 173](#) for more information.

Performance monitor exit routine parameter list

When the entry code in register 0 is X'04', VTAM passes the following parameter list to the performance monitor exit routine in register 1:

Table 92. Performance monitor exit parameter list for entry code X'04'

Dec (Hex) offset	Size (Bytes)	Description
0 (0)	4	Address of an 8-byte user field
4 (4)	8	Reserved
12 (C)	4	Address of the performance data parameter list (see “Performance data parameter list” on page 151)

When the entry code in register 0 is X'10', X'18', or X'20', VTAM passes the following parameter list to the performance monitor exit in register 1:

Table 93. Performance monitor exit parameter list for entry codes X'10', X'18', and X'20'

Dec (Hex) offset	Size (Bytes)	Description
0 (0)	4	Address of an 8-byte user field
4 (4)	8	Reserved
12 (C)	4	Address of parameter string

For an activation, deactivation, or replacement function, offset X'C' points to a parameter string that was specified as a value for the PARMS operand on the MODIFY EXIT command. This parameter string is in the form of a 2-byte length field followed by the actual data.

User data field

For the entry codes described in [Table 93 on page 150](#) and [Table 92 on page 150](#), the performance monitor exit can use this 8-byte user data field, originally initialized to 0, for any purpose. For example, the exit could use this field to store the address of a storage area obtained dynamically. This field is cleared to zero when the performance monitor exit becomes inactive.

The last 4 bytes can be used to uniquely identify an exit's level or version. The last 4 bytes are displayed following subsequent use of the DISPLAY EXIT command. See [z/OS Communications Server: SNA Operation](#) for a description of the DISPLAY EXIT command.

Performance data parameter list

For the invocation function, (the entry code in register 0 is X'04'), bytes 12–15 of the performance monitor exit routine parameter list points to a VTAM-supplied buffer containing a performance data parameter list. This buffer can be a maximum of 4088 bytes and is composed of:

- A parameter list header
- Any number of performance data vectors

The format of the parameter list header is in [Table 94 on page 151](#), followed by the formats of all supported performance data vectors. Source code for this parameter list is in ISTXPL in SYS1.MACLIB.

If the data to be delivered for a single request is larger than 4088 bytes, multiple invocations of each active performance monitor exit are made with data segments of 4088 bytes or fewer, until all of the requested data is delivered. In case of a multi-part invocation:

- All segments are correlated together as described under the reason for transmission field (X'4') in [Table 94 on page 151](#).
- Chaining indicators identify which segment is first and last.
- A given performance data vector is not split across multiple invocations.

Parameter list header format

The format of the parameter list header is shown in [Table 94 on page 151](#).

Table 94. Performance list header format

Dec (Hex) offset	Size (Bytes)	Description
0 (0)	2	Length of data segment (including length field)
2 (2)	2	Length of parameter list header only (including both length fields)

Table 94. Performance list header format (continued)

Dec (Hex) offset	Size (Bytes)	Description
4 (4)	1	Reason for transmission Hex value X'00' Because of a Collect Performance Data CNM RU (correlate multiple segments using the PRID and ACB name field starting at offset 5) X'01' VTAM event notification (multiple segments do not apply) X'02' SMF interval data (correlate multiple segments using the interval time stamp field starting at offset 16) X'03' SMF-related errors (multiple segments do not apply; no performance data vectors accompany this transmission)
5 (5)	10	Correlator Byte 5–6 PRID <ul style="list-style-type: none"> • Same as PRID from Collect Performance Data CNM RU when the reason for transmission is X'00' • Zero otherwise 7–14 ACB Name <ul style="list-style-type: none"> • Name of the issuing monitor when the reason for transmission is X'00' • Blanks otherwise

Table 94. Performance list header format (continued)

Dec (Hex) offset	Size (Bytes)	Description
15 (F)	1	<p>Flags</p> <p>General Use</p> <p>Bit setting</p> <p>B'.... .1..' SMF intervals missed since last SMF transmission (only valid when the reason for transmission is X'02')</p> <p>Chaining Indicators</p> <p>Bit setting</p> <p>B'.... ..10' First segment</p> <p>B'.... ..00' Middle segment</p> <p>B'.... ..01' Last segment</p> <p>B'.... ..11' Complete segment</p> <p>Bits not shown are reserved.</p>
16 (10)	8	<p>Interval time stamp</p> <ul style="list-style-type: none"> • SMF interval to which data should be correlated when the reason for transmission is X'02' • Zero otherwise

Performance data vector general format

The general format of the performance data vector is shown in [Table 95 on page 153](#).

Table 95. Performance data vector

Dec (Hex) offset	Size (Bytes)	Description
0 (0)	2	Length of entire performance data vector (including length field)
2 (2)	6	<p>Vector identifier</p> <p>Byte</p> <p>Description</p> <p>2–3 Major data category</p> <p>4–5 Data subcategory</p> <p>6–7 Subcategory record type</p>
8 (8)	n	Performance data (based on vector identifier)

The 6-byte vector identifier determines the performance data starting in byte 8. The vector identifier, which is a concatenation of the major data category, the data subcategory, and the subcategory record type, can be one of the following values in [Table 96 on page 154](#).

Table 96. Performance data vector identifiers

Vector identifier	Description
X'000100010001'	Global environment vector
X'000100020001'	Global installation-wide exit vector
X'000100030001'	Global storage buffer pool vector
X'000100030002'	Global storage private storage vector
X'000100030003'	Global storage CSA vector
X'000100030004'	Global storage GETBLK vector
X'000100040001'	Global session vector
X'000100050001'	Global APPN directory services vector
X'000100050002'	Global APPN directory services border node vector
X'000100060001'	Global APPN topology data vector
X'000100070001'	CSM buffer pool vector
X'000100070002'	CSM storage usage vector
X'000200010001'	Basic route data vector
X'000300010001'	RTP data vector
X'000300010002'	RTP data vector - RSCV information
X'000300010003'	RTP data vector - COS/TPF information
X'000400010001'	MNPS application recovery data vector
X'000400010002'	MNPS application data vector
X'000500010001'	MNPS coupling facility structure vector

The performance data for each vector is described in [Table 97 on page 154](#) through [Table 110 on page 170](#). In the following tables, resettable data refers to data that is reset when requested. Rolling counter data refers to data that is never reset to zero, but rather, rolls over to zero when the counter reaches its capacity.

Global environment vector (X'000100010001')

The format of the global environment vector is shown in [Table 97 on page 154](#). One global environment vector is built per request for global environment data.

Table 97. Global environment vector

Byte	Type	Description
8–11	binary	VTAM ASCB address
12–19	EBCDIC, up to 8 bytes left-adjusted	VTAM start list name
20–27	EBCDIC, up to 8 bytes left-adjusted	VTAM configuration file name

Table 97. Global environment vector (continued)

Byte	Type	Description
28–35	EBCDIC, up to 8 bytes left-adjusted	VTAM configuration restart file name
36–39	binary	Current number of outstanding D/L/R requests
40–43	binary resettable	Maximum number of outstanding D/L/R requests since last reset
44–47	binary	Maximum number of outstanding D/L/R requests since VTAM initialization

Table 97. Global environment vector (continued)

Byte	Type	Description
48–49	binary	<p>Session manager exit active options (Part 1)</p> <p>First byte</p> <p>Bit setting</p> <p>B'1.....' Initial authorization will be processed</p> <p>B'.1..' Secondary authorization will be processed</p> <p>B'..1.' Initial and final accounting functions will be processed</p> <p>B'...1' Gateway path selection will be processed</p> <p>B'.... 1...' End function will be processed</p> <p>B'.... .1..' Session takeover accounting will be processed</p> <p>B'.... ..1.' SSCP selection function will be processed</p> <p>B'.... ...1' Adjacent SSCP selection for DSRLST will be processed</p> <p>Second byte</p> <p>B'1...' Exit functions allowed for LU session takeover</p> <p>B'.1..' Initial authorization allowed for Init Other CD</p> <p>B'..1.' Alias selection will be processed</p> <p>B'...1' ALS (adjacent link station) selection will be processed</p> <p>B'.... 1...' ALS (adjacent link station) selection will be processed for DSSIB requests</p> <p>B'.... .1..' ALS (adjacent link station) selection will be processed for CDRSC requests</p> <p>B'.... ..1.' SME supports the MODIFY REPLACE command</p> <p>B'.... ...1' VR selection will be processed</p>

Table 97. Global environment vector (continued)

Byte	Type	Description
50	binary	<p>Global environment flags (Part 1)</p> <p>Bit setting</p> <p>B'1...' VTAM SSCP gateway capability specified</p> <p>B'.1..' APPN network node capability</p> <p>B'..1.' APPN end node capability</p> <p>B'...1' APPN central directory server capability</p> <p>B'.... 1...' APPN interchange node capability</p> <p>B'.... .1..' APPN migration data host capability</p> <p>B'.... ..1.' APPN border node capability</p> <p>Bits not shown are reserved.</p>
51–52	binary	<p>Directory services manager exit (DSME) active options</p> <p>First byte</p> <p>Bit setting</p> <p>B'1...' Initial authorization will be processed</p> <p>B'...1' Border node selection option supported</p> <p>B'.... 1...' End function will be processed</p> <p>B'.... .1..' CDS selection option supported</p> <p>B'.... ..1.' ADS selection option supported</p> <p>B'.... ...1' CDS selection supported for central resource registration option</p> <p>Second byte</p> <p>B'.... ..1.' DSME supports the MODIFY REPLACE command</p> <p>Bits not shown are reserved.</p>
53–54	binary	Tuning statistics timing interval

Table 97. Global environment vector (continued)

Byte	Type	Description
55	binary	<p>Tuning statistics settings</p> <p>Bit setting</p> <p>B'1...' Tuning statistics is active</p> <p>B'.1..' Tuning statistics task is attached</p> <p>B'..1.' Display TNSTATs record at console</p> <p>Bits not shown are reserved.</p>
56–59	binary	DIRSIZE start option value
60–61	binary	<p>Session manager exit active options (Part 2)</p> <p>First byte</p> <p>Bit setting</p> <p>B'1...' MNPS recovery calls will be processed</p> <p>B'.1..' HPR virtual route selection option supported</p> <p>B'..11 1111' Reserved for SME usage</p>
62	EBCDIC, up to 8 bytes left-adjusted	Local network ID
70	EBCDIC, up to 8 bytes left-adjusted	Local CP name
78	binary	<p>Global environment flags (Part 2)</p> <p>Bit setting</p> <p>B'1.....' Multinode persistent session capability specified</p> <p>B'.11.....' HPR capability</p> <p>HPR capability</p> <p>Bit setting</p> <p>B'.00.' No HPR support</p> <p>B'.10.' ANR-level HPR support</p> <p>B'.11.' RTP-level HPR support</p> <p>Bits not shown are reserved.</p>

Global installation-wide exit vector (X'000100020001')

The format of the global installation-wide exit vector is shown in [Table 98 on page 159](#). One global installation-wide exit vector is built for each installation-wide exit defined to VTAM per request for global installation-wide exit data.

Table 98. Global installation-wide exit vector

Byte	Type	Description
8–15	EBCDIC, left-adjusted	Exit name
16–23	EBCDIC, left-adjusted	Exit module name
24–27	binary	Exit level
28	binary	Exit status X'10' Exit status inactive X'20' Exit status pending inactive X'30' Exit status pending inactive replace X'40' Exit status pending active replace X'50' Exit status pending active X'60' Exit status active

Global storage buffer pool vector (X'000100030001')

The format of the global storage buffer pool vector is shown in [Table 99 on page 159](#). One global storage buffer pool vector is built for each buffer pool that exists in VTAM per request for global storage usage data.

Table 99. Global buffer pool vector

Byte	Type	Description
8–11	EBCDIC, left-adjusted	Buffer pool ID name
12–13	binary	Size of each buffer
14–17	binary	Base number of buffers defined at VTAM initialization
18–21	binary	Total number of buffers currently defined
22–25	binary	Total number of buffers available
26–29	binary	Number of static buffers available
30–33	binary resettable	Maximum number of buffers in use since last reset

Table 99. Global buffer pool vector (continued)

Byte	Type	Description
binary	Expansion size in terms of buffers. This value is the same as the number of buffers per extent.	
36–39	binary	Expansion size in terms of bytes
40–41	binary	Expansion threshold Note: If value is X'FFFF', pool is not eligible for expansion.
42–45	binary	Expansion limit
46–47	binary	Contraction threshold Note: If value is X'7FFF', pool has no expansion extents.
48–51	binary rolling counter	Number of expansions
52–55	binary	Slowdown threshold
56–59	binary	Number of queued requests (RPHs)
60–63	binary	Number of buffers needed to satisfy queued requests
64–67	binary resettable	Maximum number of requestors waiting since last reset
68–71	binary resettable	Maximum number of bytes allocated since last reset
72–73	binary	Pool attribute flags First byte Bit setting B'1..' Fixed storage B'..1.' Pool resides in fetch protected storage B'.... 1...' Storage allocated below 24-bit line Second byte B'1...' User request fixed buffer Bits not shown are reserved.
74–75	binary	Subpool number
76–77	binary	User-specified base number of buffers (BASENO)
78–79	binary	User-specified number of expand buffers (XPANNO)
80–83	binary	Number of active extents

Table 99. Global buffer pool vector (continued)

Byte	Type	Description
84–87	binary	Number of available buffers in extents

Global storage private storage vector (X'000100030002')

The format of the global storage private storage vector is shown in Table 100 on page 161. One global storage private storage vector is built per request for global storage usage data.

Table 100. Global storage private storage vector

Byte	Type	Description
8–11	binary	Current private storage allocated
12–15	binary	Maximum private storage allocated

Global storage CSA vector (X'000100030003')

The format of the global storage Common Storage Area (CSA) vector is shown in Table 101 on page 161. One global storage CSA vector is built per request for global storage usage data.

Table 101. Global storage CSA vector

Byte	Type	Description
8–11	binary	Maximum size CSA allowed
12–15	binary	Maximum size CSA allowed for VTAM below 16M line
16–19	binary	Current allocation of total CSA for VTAM
20–23	binary resettable	Total CSA high water mark since last reset
24–31	binary resettable	Timestamp of total CSA high water mark since last reset
32–35	binary resettable	Total CSA low water mark since last reset
36–43	binary resettable	Timestamp of total CSA low water mark since last reset
44–47	binary	Value of maximum total CSA used since VTAM initialization
48–51	binary	Current allocation of CSA below the 16M line
52–55	binary	Value of maximum total CSA 24 used since VTAM initialization
56–59	binary resettable	CSA 24 highwater mark since last reset
60–67	binary resettable	Timestamp of CSA 24 high water mark since last reset

Table 101. Global storage CSA vector (continued)

Byte	Type	Description
68–71	binary resettable	CSA 24 low water mark since last reset
72–79	binary resettable	Timestamp of CSA 24 low water mark since last reset

Global storage GETBLK vector (X'000100030004')

The format of global storage GETBLK vector is shown in Table 102 on page 162. One global storage GETBLK vector is built for each GETBLK subpool that exists in VTAM per request for global storage usage data. A GETBLK subpool is identified by its pool ID and buffer size. A given GETBLK pool might also be made up of multiple subpools, each of which have a different buffer size.

Table 102. Global storage GETBLK vector

Byte	Type	Description
8–15	EBCDIC, left-adjusted	Pool name
16–19	binary	Subpool buffer length
20–21	binary	Number of buffers per page
22–25	binary	Page size
26–29	binary	Number of bytes in use in subpool ¹
30–33	binary resettable	Maximum number of bytes in use in subpool since last reset ¹
34–41	binary resettable	Timestamp for maximum number of bytes in use in subpool since last reset
42–45	binary	Number of bytes allocated in subpool ¹
46–49	binary resettable	Maximum number of bytes allocated in subpool since last reset ¹
50–57	binary resettable	Timestamp for maximum number of bytes allocated in subpool since last reset

Note:

1. If bit 0 is off, these fields contain a byte count. If bit 0 is on, bits 1-31 contain a megabyte count.

Global session vector (X'000100040001')

The format of the global session vector is shown in Table 103 on page 162. One global session vector is built per request for global session data.

Table 103. Global session vector

Byte	Type	Description
8–11	binary rolling counter	The number of successful LU-LU session starts and takeovers

Table 103. Global session vector (continued)

Byte	Type	Description
12–15	binary rolling counter	The number of LU-LU session stops
16–19	binary rolling counter	The number of LU-LU session setup failures
20–23	binary resettable	Maximum number of active LU-LU sessions since last reset
24–31	binary resettable	Timestamp for maximum number of active LU-LU sessions since last reset
32–35	binary resettable	Minimum number of active LU-LU sessions since last reset
36–43	binary resettable	Timestamp for minimum number of active LU-LU sessions since last reset
44–47	binary	The current number of LU-LU sessions in queued initiate status
48–51	binary	The current number of LU-LU sessions in pending reallocation status
52–55	binary	The current number of LU-LU sessions in pending active status
56–59	binary	The current number of LU-LU sessions in active status
60–63	binary	The current number of LU-LU sessions in pending termination status
64–67	binary	Number of defined application LUs
68–71	binary	Number of defined non-application dependent LUs
72–75	binary	Number of defined CDRSCs
76–79	binary	Number of defined PUs
80–83	binary	Number of active application LUs
84–87	binary	Number of active non-application dependent LUs
88–91	binary	Number of active CDRSCs
92–95	binary	Number of active PUs
96–99	binary rolling counter	Number of CP-CP session failures (can retry)
100–103	binary rolling counter	Number of CP-CP session failures (cannot retry)
104–107	binary rolling counter	Number of successful CP-CP session initiations

Table 103. Global session vector (continued)

Byte	Type	Description
108–111	binary	Number of Dependent LU Requestor (DLUR) end nodes
112–115	binary	Number of Dependent LU Requestor (DLUR) network nodes
116–119	binary	Number of Dependent LU Requestor (DLUR) served PUs
120–123	binary	Number of Dependent LU Requestor (DLUR) served LUs
124–127	binary	Number of active independent LUs
128–131	binary	Number of end nodes with which this network node has CP-CP sessions. This number is only valid on a network node.
132–135	binary	Number of network nodes with which this network node has CP-CP sessions. This number is only valid on a network node.

Global APPN directory services vector (X'000100050001')

The format of the global APPN directory services vector is shown in Table 104 on page 164. One global APPN directory services vector is built per request for global APPN directory services data.

Table 104. APPN directory services vector

Byte	Type	Description
8–11	binary rolling counter	Number of resources successfully found in the local database (cache). (Statistic kept in an origin central directory server or a network node server of an OLU that originated the search request.)
12–15	binary rolling counter	Number of resources successfully found from the central directory server. (Statistic kept in an origin central directory server or a network node server of an OLU that originated the search request.)
16–19	binary rolling counter	Number of resources successfully found using domain broadcast. (Statistic kept in an origin central directory server or a network node server of an OLU that originated the search request.)
20–23	binary rolling counter	Number of resources successfully found using network broadcast. (Statistic kept in an origin central directory server or a network node server of an OLU that originated the search request.)
24–27	binary rolling counter	Number of failed searches completed by receipt of a negative reply. (Statistic kept in an origin central directory server or a network node server of an OLU that originated the search request.)
28–31	binary rolling counter	Number of times a network broadcast is not performed due to a negative cache entry. (Statistic kept in an origin central directory server or a network node server of an OLU that originated the search request.)

Table 104. APPN directory services vector (continued)

Byte	Type	Description
32–35	binary rolling counter	Number of searches where this node received a search request from another node and the requested resource was found. (Statistic kept only in nodes that did not originate the search.)
36–39	binary rolling counter	Number of searches where this node received a search request from another node and the requested resource was not found. (Statistic kept only in nodes that did not originate the search.)
40–43	binary rolling counter	Number of referrals received. (Statistic kept at a central directory server only.)
44–47	binary rolling counter	Number of times a positive reply is received from an alternate central directory server. (Statistic kept at a central directory server only.)
48–51	binary rolling counter	Number of times a negative reply is received from an alternate central directory server. (Statistic kept at a central directory server only.)
52–55	binary rolling counter	Number of resources removed from local database due to the value specified for either the DIRSIZE or DIRTIME start options.
56–59	binary	Number of adjacent nodes in a different subnetwork
60–63	binary rolling counter	Number of times a cache entry is discarded due to DIRSIZE maximum being reached
64–67	binary	Number of registered resources
68–71	binary	Current number of entries in the cache

Global APPN directory services border node vector (X'000100050002')

The format of the global APPN directory services border node vector is shown in Table 105 on page 165. One global APPN directory services border node vector is built for each network node adjacent to this VTAM that this VTAM can send or receive inter-subnetwork searches. One vector is built for each such node per request for global APPN directory services data.

This data is reported only on an APPN border node.

Table 105. Global APPN directory services border node vector

Byte	Type	Description
8–23	EBCDIC, left-adjusted	The network-qualified CPNAME 8-byte NetID. 8-byte CPNAME
24–27	binary rolling counter	Number of subnetwork searches sent to the adjacent node
28–31	binary rolling counter	Number of subnetwork searches received from the adjacent node

Global APPN topology data vector (X'000100060001')

The format of the global APPN topology data vector is shown in [Table 106 on page 166](#). One global APPN topology data vector is built per request Global APPN Topology data.

Table 106. Global APPN topology data vector

Byte	Type	Description
8–11	binary rolling counter	Number of routes calculated using an existing tree
12–15	binary rolling counter	Number of routes calculated using a modified tree
16–19	binary rolling counter	Number of routes calculated using a new tree
20–23	binary rolling counter	Number of topology database updates (TDUs) originated by this node
24–27	binary rolling counter	Number of topology database updates (TDUs) propagated by this node
28–31	binary rolling counter	Number of TDUs received by this node resulting in a topology database update
32–35	binary rolling counter	Number of TDUs received by this node and discarded for normal reasons. Resource sequence number (RSN) is valid, but data already seen.
36–39	binary rolling counter	Number of TDUs received by this node and discarded because the RSN is not valid. (TDU is rebroadcast.)
40–43	binary rolling counter	Number of TDUs received by this node and discarded because of data inconsistency. (TDU is rebroadcast.)
44–47	binary	Number of network nodes in the topology database
48–51	binary	Number of end nodes in the topology database
52–55	binary	Number of unidirectional TGs in the topology database
56–59	binary rolling counter	Number of TG failures at this node
60–63	binary rolling counter	Number of dynamic PU allocations
64–67	binary rolling counter	Number of times a routing tree was discarded due to the tree cache being full
68–71	binary	Current number of trees in the tree cache
72–75	binary	Number of virtual nodes in the topology database

Table 106. Global APPN topology data vector (continued)

Byte	Type	Description
76–79	binary	Number of central directory server nodes in the topology database
80–83	binary	Number of interchange nodes in the topology database
84–87	binary	Number of end nodes connected to this node
88–91	binary	Number of network nodes connected to this node
92–95	binary	Number of virtual nodes connected to this node

CSM buffer pool vector (X'000100070001')

The format of the CSM buffer pool vector is shown in [Table 107 on page 167](#).

Table 107. CSM buffer pool vector

Byte	Type	Description
8–11	binary	Buffer size
12	binary	Buffer source Bit setting B'10..' ECSA buffer pool B'01..' Data space buffer pool Bits not shown are reserved.
13–16	binary	Number of buffers in the pool that are in use ¹
17–20	binary	Number of available buffers in the pool ¹

Note:

1. For data space, these fields contain the accumulation of both the 31- and 64-bit backed pools.

CSM storage usage vector (X'000100070002')

The format of the CSM fixed storage usage vector is shown in [Table 108 on page 167](#).

Table 108. CSM storage usage vector

Byte	Type	Description
8–11	binary	Installation maximum for fixed storage usage ¹
12–15	binary	Current fixed storage usage ¹
16–19	binary	Installation maximum for ECSA storage usage
20–23	binary	Current ECSA storage usage

Note:

1. If bit 0 is off, these fields contain a byte count. If bit 0 is on, bits 1-31 contain a megabyte count.

Basic route data vector (X'000200010001')

The format of the basic route data vector is shown in Table 109 on page 168. A basic route data vector will be built for each virtual route being collected on (excluding those that are reset, inactive, or pending active) per request for virtual route data. A virtual route is identified by destination subarea, virtual route number, and transmission priority.

Table 109. Basic route data vector

Byte	Type	Description
8–11	binary	Destination subarea
12	binary	Transmission priority
13	binary	Virtual route number
14–17	binary	Adjacent subarea
18	binary	Explicit Route Numbers Bit setting B'1111' Initial explicit route number B'.... 1111' Actual explicit route number
19	binary	Reverse explicit route number
20–21	binary	Number of sessions on the virtual route
22	binary	Current minimum window size
23	binary	Current maximum window size
24	binary	Default minimum window size
25	binary	Default maximum window size
26–29	binary rolling counter	Number of times window size is reset to minimum
30–33	binary	Number of PIUs on the VR waiting to be sent
34	binary	Virtual route state: Hex value X'01' Inactive X'02' Pending inactive X'03' Flush in progress X'05' Active X'06' DACTVR FORCE in progress

Table 109. Basic route data vector (continued)

Byte	Type	Description
35	binary	Flow control state Bit setting B'.... ..00' Route is blocked B'.... ..01' Route is held B'.... ..11' Route is open Bits not shown are reserved.
36–39	binary	Number of times that the virtual route was in the held state (not including current instance)
40–47	binary	Time that the virtual route has been in the held state if currently held
48–55	binary rolling counter	Total time that the virtual route was in the held state (not including current held time)
56–63	binary resettable	Maximum time that the virtual route was in the held state (not including current time) for any given held instance since last reset
64–71	binary resettable	Timestamp indicating when maximum held instance since last reset left the held state
72–75	binary rolling counter	Number of times the virtual route was blocked (not including current instance)
76–83	binary	Time that the virtual route has been in the blocked state if currently blocked
84–91	binary rolling counter	Total time that the virtual route was in the blocked state (not including current blocked time)
92–99	binary resettable	Maximum time that the virtual route was blocked (not including current time) for any given blocked instance since last reset
100–107	binary resettable	Timestamp indicating when maximum blocked instance since last reset left the blocked state
108–109	binary	TOTAL number of LU-LU sessions on the virtual route
110–111	binary	Current pacing request send count
112	binary	Current pacing window size
113–116	binary	Outbound PIU count over VR
117–120	binary	Inbound PIU count over VR

RTP data vector (X'000300010001')

The format of the RTP data vector is shown in [Table 110 on page 170](#). RTP data vectors are built for each RTP specified in a collect request.

Table 110. RTP data vector

Byte	Type	Description
8–9	binary	First byte Bit setting B'0000 0000' Normal collect B'1000 0000' Exit called due to HPR path switch B'0100 0000' Exit called because RTP is being deleted B'0010 0000' Exit called due to HPR path switch with CP name change. The New remote network ID field and the New remote CP name field are set. Second byte is reserved.
10–17	binary	Local NCE
18–25	binary	Local TCID
26–33	binary	Rapid Transport Protocol (RTP) Physical Unit
34–41	EBCDIC	Remote network ID
42–49	EBCDIC	Remote CP name
50–57	EBCDIC	New Remote network ID
58–65	EBCDIC	New remote CP name
66–73	binary	Remote NCE
74–81	binary	Remote TCID
82	binary	RTP state
83–86	binary	HPR path switch timer
87–90	binary	Round-trip delay
91–94	binary rolling counter	Number of bytes sent over the RTP
95–98	binary rolling counter	Number of bytes received over the RTP
99–102	binary rolling counter	Number of PIUs that were first in segment sent over the RTP
103–106	binary rolling counter	Number of PIUs that were middle in segment sent over the RTP

Table 110. RTP data vector (continued)

Byte	Type	Description
107–110	binary rolling counter	Number of PIUs that were last in segment sent over the RTP
111–114	binary rolling counter	Number of PIUs that were not segmented sent over the RTP
115–118	binary rolling counter	Number of PIUs that were first in segment received over the RTP
119–122	binary rolling counter	Number of PIUs that were middle in segment received over the RTP
123–126	binary rolling counter	Number of PIUs that were last in segment received over the RTP
127–130	binary rolling counter	Number of PIUs that were not segmented received over the RTP
131–134	binary rolling counter	Number of times the allowed send rate changed
135–136	binary rolling counter	HPR path switch attempts initiated by other RTP endpoint
137–138	binary rolling counter	Total number of HPR path switch attempts initiated by this VTAM
139–140	binary rolling counter	HPR path switch attempts due to operator or VTAM command. Path switch attempts due to any other failure (such as link failure or node failure) can be calculated by subtracting this value from the total number of path switch attempts.
141–142	binary rolling counter	Total number of unsuccessful route selection attempts initiated by this VTAM during HPR path switch
143–144	binary rolling counter	Unsuccessful HPR path switch due to operator command. The number of unsuccessful path switches that originated due to any other failure (such as link failure or node failure) can be calculated by subtracting this value from the total number of unsuccessful path switch attempts.
145–148	binary	Number of active LU-LU sessions using the RTP
149–152	binary rolling counter	Amount of time in seconds that back pressure has been applied
153–154	binary rolling counter	Total number of times that back pressure has been applied

Table 110. RTP data vector (continued)

Byte	Type	Description
155–156	binary rolling counter	Number of times that back pressure has been applied due to HPR path switch
157–158	binary rolling counter	Number of times that back pressure has been applied due queue depth exceeded. The number of times that back pressure has been applied due to storage shortage can be calculated by subtracting the above two counts from the total count.
159–162	binary rolling counter	Number of retransmitted NLPs
163–166	binary rolling counter	Current number of unacknowledged buffers

RTP data vector – RSCV information (X'000300010002')

If RSCV information is sent, an individual vector is created because RSCV information is variable length. For details, see *SNA Formats*.

RTP data vector – COS/TPF information (X'000300010003')

Because COS/TPF information is variable length, an individual vector is created. For details, see *SNA Formats*.

MNPS application recovery data vector (X'000400010001')

The format of the MNPS application recovery data vector is shown in Table 111 on page 172. One MNPS application data vector is built for each multinode persistent session application program that has recovered on this VTAM.

Table 111. MNPS application recovery data

Byte	Type	Description
8–15	EBCDIC, left-adjusted	Application program name
16–19	binary	Number of sessions recovered successfully by this VTAM
20–23	binary	Number of session recoveries attempted
24–27	binary	Number of HPR pipes recovered successfully by this VTAM
28–31	binary	Number of HPR pipes recoveries attempted
32–39	binary	Total recovery time spent restoring session (TOD clock format)

MNPS application data vector (X'000400010002')

The format of the MNPS application data vector is shown in [Table 112 on page 173](#). One MNPS application data vector is built for each multinode persistent application program.

Table 112. MNPS Application Data

Byte	Type	Description
8–15	EBCDIC, left-adjusted	Application program name
16–31	EBCDIC, left-adjusted	Coupling facility structure name containing the application data
32–35	binary rolling counter	Number of coupling facility Writes for this application
36–39	binary rolling counter	Number of bytes written to the coupling facility structure

MNPS structure data vector (X'000500010001')

The format of the MNPS structure data vector is shown in [Table 113 on page 173](#). One global MNPS structure vector is built for each coupling facility structure being used for multinode persistent data.

Table 113. MNPS structure data vector

Byte	Type	Description
8–23	EBCDIC, left-adjusted	Coupling facility structure name
24–27	binary	Coupling facility structure size (in 1-KB blocks)
28	binary	Percentage of 1-KB blocks in use

Installing VTAM exit routines

To install VTAM exit routines you assemble and link-edit the exits.

Procedure

Follow these steps to install the appropriate VTAM exits:

1. Assemble the exits.
2. Link-edit the exits to the appropriate VTAM library as indicated in [Table 114 on page 173](#).

Table 114. Libraries for VTAM exit routines

Exit routine	MVS library
Session management exit routine	SYS1.VTAMLIB
Virtual route selection exit routine	SYS1.VTAMLIB
Virtual route pacing window size calculation exit routine	SYS1.VTAMLIB
Configuration services XID exit routine	SYS1.VTAMLIB

Table 114. Libraries for VTAM exit routines (continued)

Exit routine	MVS library
Selection of definitions for dependent LUs exit routine	SYS1.VTAMLIB
USERVAR exit routine	SYS1.VTAMLIB
Command verification exit routine	SYS1.VTAMLIB
Directory services management exit routine	SYS1.VTAMLIB
Generic resource resolution exit routine	SYS1.VTAMLIB
Performance monitor exit routine	SYS1.VTAMLIB
Session accounting exit routine	SYS1.LPALIB ¹
Session authorization exit routine	SYS1.LPALIB ¹

Note:

- a. If you place your user-written exit in SYS1.LPALIB, re-IPL to reformat the link-pack area. For details, see [z/OS MVS Initialization and Tuning Reference](#).

Link-edit VTAM exits before starting VTAM. Link-editing after VTAM is started might lead to subsequent load errors during exit activation. The names of the user-written exits must be those shown in Table 115 on page 174 unless you have provided an alternate exit or a multiple exit using the MODIFY EXIT command. For a discussion of this command, see [z/OS Communications Server: SNA Operation](#).

Table 115. VTAM exit names for exit routines

Exit routine	VTAM module name
Session management exit routine	ISTEXCAA
Virtual route selection exit routine	ISTEXCVR
Virtual route pacing window size calculation exit routine	ISTPUCWC
Configuration services XID exit routine	ISTEXCCS
Selection of definitions for dependent LUs exit routine	ISTEXCSD
Command verification exit routine	ISTCMMND
USERVAR exit routine	ISTEXCUV
Directory services management exit routine	ISTEXCDM
Performance monitor exit routine	ISTEXCPM
Session accounting exit routine	ISTAUCAG
Session authorization exit routine	ISTAUCAT
Generic resource resolution exit routine	ISTEXCGR

Operator commands for VTAM exit routines

The DISPLAY EXIT command can be used for displaying information about certain VTAM exit routines. The MODIFY EXIT command can be used for activating, deactivating, and replacing certain VTAM exits. Table 116 on page 175 shows the exit routines that you can either display or modify. For more information about the use of these commands, see [z/OS Communications Server: SNA Operation](#).

Table 116. Modifiable VTAM exit routines

Exit routine	VTAM exit name
Session management exit routine	ISTEXCAA
Virtual route selection exit routine	ISTEXCVR
Configuration services XID exit routine	ISTEXCCS
Selection of definitions for dependent LUs exit routine	ISTEXCSD
Command verification exit routine	ISTCMMND
USERVAR exit routine	ISTEXCUV
Directory services management exit routine	ISTEXCDM
Generic resource resolution exit routine	ISTEXCGR
Performance monitor exit routine	ISTEXCPM

Activating and deactivating VTAM exit routines

You can use the MODIFY EXIT command to activate or deactivate an exit without interrupting VTAM processing. You can activate or deactivate an exit that was loaded during VTAM initialization, or that was activated or deactivated with an earlier MODIFY EXIT command.

If the MODIFY EXIT command is successful, message IST984I indicates the exit's status. Otherwise, one of the following messages is issued: IST985I, IST452I, IST453I, IST454I, IST456I. See [z/OS Communications Server: SNA Messages](#) for more information.

An exit might become inactive during processing without a MODIFY EXIT command having been issued. In this case, message IST984I indicates that the exit is inactive.

Replacing VTAM exit routines

You can use the MODIFY EXIT command to replace an exit with an alternate copy of the exit without interrupting VTAM processing.

Follow these steps to prepare the alternate copy of the exit that will replace the current copy of the exit:

1. Assemble the exit. You should use the VTAM module name shown in [Table 116 on page 175](#) unless you provide an alternate exit.
2. Place the exit in the appropriate VTAM library as indicated in [Table 114 on page 173](#).

If the replacement is successful, message IST984I is issued by VTAM. The replacement might fail because VTAM cannot find the alternate copy of the exit or VTAM might not be able to load the alternate copy of the exit. Also, an error in the MODIFY EXIT command can cause replacement failure. VTAM issues any of the following messages to indicate replacement failure: IST985I, IST452I, IST453I, IST454I, IST456I. See [z/OS Communications Server: SNA Messages](#) for more information.

If VTAM cannot load or activate the alternate exit, VTAM attempts to reactivate the original exit and regain the function of the exit. If the exit cannot be activated, message IST984I indicates that the original exit is inactive.

Passing user-defined data to installation-wide exit routines

You can pass user-defined data to an exit with the MODIFY EXIT,PARMS= command. The following exit routines support this function:

ISTCMMND

Command verification exit routine

ISTEXCGR

Generic resource resolution exit routine

ISTEXCPM

Performance monitor exit routine

ISTEXCUV

USERVAR exit routine

User-defined data can be passed to the exit to allow the exit to tailor its processing. See the description of the exit routine for how to access the data from the exit.

Chapter 2. Writing logon manager installation-wide exit routines

This topic provides information about using exit routines to control Transaction Processing Facility (TPF) logon manager processing. The exits are specific to the TPF environment. TPF is a host processor, dedicated to high-speed transaction processing, that can handle a large volume of transactions submitted from many remote terminals. The logon manager is a VTAM application program. If TPF is defined to VTAM as a type 2.1 peripheral node, the logon manager provides load balancing and logon services for TPF.

Logon manager exit routines

This topic discusses two exit routines that apply to the TPF environment: a USERVAR exit routine and a control logical unit (CLU) search exit routine. The USERVAR routine is an exit from VTAM that enables the logon manager to provide a single-system image to PLUs initiating sessions with TPF. This allows the user to request a session with a TPF application program using a generic name. The CLU search routine is an exit from the logon manager that allows you to customize the logon manager's load balancing.

A sample USERVAR exit routine is stored in SYS1.SAMPLIB under the VTAM module name ISTEXCUV. The sample USERVAR exit routine is an implementation of a VTAM exit routine that is specific to the TPF environment. A sample CLU search exit routine is also stored in SYS1.SAMPLIB under the module name ELMCLUEx. You can use the samples as they are coded. However, if you want logic different from the sample exit routines, you can modify the samples, or write your own USERVAR or CLU search exit routine. Although the CLU search exit is a user-replaceable module, the logon manager cannot operate without it. Likewise, the logon manager must use a USERVAR exit to process any session requests other than those from dependent SLUs.

This topic describes the function of the sample exit routines. It also contains interface information for the CLU search exit. See [“USERVAR exit routine” on page 109](#) for interface information for that exit. You might also see [Appendix F, “Sample USERVAR exit routine for TPF sessions,” on page 277](#) and [Appendix G, “Sample CLU search exit routine for TPF sessions,” on page 281](#).

The logon manager defines a USERVAR for each application program it supports whenever it opens an ACB for the application program. The logon manager defines the USERVAR in the host where the logon manager is running. The logon manager initializes the USERVAR value to the name of the logon manager ACB for the application program and specifies TYPE=VOLATILE. Being volatile, TPF USERVARs are not stored by the SSCP that owns the OLU and the USERVAR table is not updated. Instead, the USERVAR exit at the DLU SSCP provides a new value for the USERVAR translation each time VTAM drives the exit. The USERVAR exit is driven for every TPF session request so that the exit can assign a TPF destination logical unit based on current load conditions. The USERVAR exit does this by using the REQTAIL macro to invoke the logon manager for PLU-initiated sessions. The logon manager invokes the CLU search exit routine, which is responsible for choosing the best CLU for a session.

Criteria for contending CLUs

The CLU search exit routine compares contending CLUs and selects the best CLU for the session. In the sample CLU search exit, the logon manager ensures that the contending CLUs meet certain minimum requirements. These requirements vary, depending on whether the session request comes from a dependent SLU, an independent or dependent PLU, or from TPF.

Contending CLUs for dependent SLU-initiated sessions

If the session request comes from a dependent SLU, the logon manager ensures that the contending CLUs have the following characteristics:

- Support the class of service

- Support the application program
- Are active
- Are within their session limits

The sample exit uses the following selection algorithm to compare two contending CLUs and select the better CLU of the pair. The better CLU is the one that has the lower hop count and the lower session count.

Contending CLUs for PLU-initiated sessions

If the session request comes from an independent or dependent PLU, the logon manager ensures that the contending CLUs have the following characteristics:

- Support the class of service
- Support the application program
- Are active
- Are within their session limits
- Have disjoint session tails
- Are on the same host as any CLUs that have current active sessions with this PLU

The sample exit uses the following selection algorithm to compare two contending CLUs and select the better CLU of the pair. The better CLU is the one that has the lower value for the following:

- Hop count
- Number of parallel sessions with the OLU
- Session count

Contending CLUs for TPF-initiated sessions

If the session request comes from TPF, the logon manager ensures that the contending CLUs have the following characteristics:

- Support the class of service
- Support the application program
- Are active
- Are within their session limits
- Have disjoint session tails
- Are on the same host as the CLU that sent the original REQTAIL RU

The sample exit uses the following selection algorithm to compare two contending CLUs and select the better CLU of the pair. The better CLU is the one that has the lower hop count and the lower session count.

TPF sessions

The logon manager supports session requests from dependent SLUs, session requests from independent and dependent PLUs, and sessions initiated by TPF. Processing is different for each type of session. This topic outlines the basic session flow for each session type. The session flows indicate when the USERVAR and CLU search exits are invoked. The logon manager always invokes the CLU search exit for any TPF session request. There are two potential paths into the logon manager: one is through a logon exit, the other is through a REQTAIL macro issued by a USERVAR exit. Both paths result in the invocation of the CLU search exit.

Session requests from dependent SLUs

When the user of a dependent SLU initiates a session request with a TPF application program, VTAM search processing begins. In response to a CDINIT from the SSCP that owns the SLU, the SSCP that owns

the DLU (and has the logon manager) drives the USERVAR exit for the USERVAR representing the TPF application program.

The sample USERVAR exit processes only volatile USERVARs. When the sample exit receives a request to translate a static or dynamic USERVAR, the USERVAR exit turns off the translation bit and returns a nonzero return code to indicate that the session establishment request failed.

For a volatile USERVAR, the sample USERVAR exit does not translate the name. The exit instead sets the value of the USERVAR equal to the generic USERVAR name. Although the name is not translated, the exit turns on the translation flag. The translation flag must be turned on to ensure that VTAM drives the logon manager through the logon exit.

The logon manager then drives the CLU search exit for each pair of contending CLUs for the session. The CLU search exit compares each pair of contending CLUs and identifies the better CLU from the pair to service the application program. After the list of contending CLUs is exhausted and all pairs have been compared, the CLU search exit returns the name of the best CLU to the logon manager. The best CLU is the one that meets the criteria described in [“Contending CLUs for dependent SLU-initiated sessions” on page 177](#). The logon manager sends a session initiation request to the best CLU. The logon exit rejects the session so that the SLU will be in session with TPF rather than with the logon manager. The logon manager issues the CLSDST with sense code 0842FFFF so that the LU is not notified of session failure. TPF uses the session initiation request to build a BIND. TPF sends the BIND from the application program LU associated with the best CLU to the network resource that requested the session.

Session requests from independent and dependent PLUs

An independent or dependent PLU requests a session with a TPF application program by issuing a BIND with a generic application program LU name.

The sample USERVAR exit processes only volatile USERVARs. When the sample exit receives a request to translate a static or dynamic USERVAR, the exit turns off the translation bit and returns a nonzero return code to indicate that the session establishment request failed.

For a volatile USERVAR, the USERVAR exit at the logon manager host (the DLU host) uses the REQTAIL macro to invoke the logon manager. The logon manager then drives the CLU search exit for each pair of contending CLUs for the session. The CLU search exit compares each pair of contending CLUs and identifies the better CLU from the pair to service the application program. After the list of contending CLUs is exhausted and all pairs have been compared, the CLU search exit returns the name of the best CLU to the logon manager. The best CLU is the one that meets the criteria described in [“Contending CLUs for PLU-initiated sessions” on page 178](#). The logon manager finds the application program LU name associated with the best CLU and returns that name to the REQTAIL macro. The REQTAIL macro returns the application program LU name and a completion code indicating that the search was successful to the USERVAR exit. The USERVAR exit turns on the translation flag and returns the translated name to the SSCP. The SSCP then establishes the session.

The logon manager and TPF work together to ensure that parallel LU 6.2 sessions are sent to the same TPF host. For LU 6.2 initiated sessions, the USERVAR exit passes a list of current session partners (for the session requestor) to the REQTAIL macro. The logon manager scans this list to determine whether the LU already has a session with the same TPF resource. The logon manager uses this information to select only those CLUs that are on the same TPF host.

Sessions initiated by TPF

TPF can ask the logon manager for the name of the CLU that would provide the best path to a DLU. TPF requests a CLU assignment by sending an unsolicited REQTAIL RU to the logon manager. When the logon manager receives a REQTAIL RU with a DLU name, but no DLU subarea address, the logon manager issues a SIMLOGON to find the DLU subarea address and the COS name for the session in this network. When driven with the subsequent CINIT, the logon manager rejects the session, but uses CINIT and the original REQTAIL RU to find the best CLU. The best CLU is the one that meets the criteria described in [“Contending CLUs for TPF-initiated sessions” on page 178](#). TPF will not request parallel sessions from disjoint hosts. If TPF receives a parallel session on the wrong host, it rejects the BIND with sense code 08014001. The logon manager sends a SESINIT (bind image) to the application program LU associated with the best CLU.

TPF builds the BIND and sends it to the application program LU. The application program LU sends the BIND to the DLU.

USERVAR exit routine for TPF sessions

The sample USERVAR exit routine for TPF sessions is stored in SYS1.SAMPLIB under the VTAM module name ISTEXCUV. If you write your own USERVAR routine instead of using the sample, you need to follow the instructions described in “USERVAR exit routine” on page 109. You might also see [Appendix F, “Sample USERVAR exit routine for TPF sessions,”](#) on page 277.

VTAM invokes the USERVAR exit routine for exit activation, exit deactivation, and whenever a USERVAR needs to be added, updated, deleted, or translated.

Activation processing

When VTAM invokes the USERVAR exit routine for activation processing, the USERVAR exit obtains an area of pageable VTAM private storage and passes a pointer to the area back to VTAM. The USERVAR exit frees the area when called for exit deactivation. The storage area is called the USERVAR exit control block (EXCB). When the USERVAR exit is invoked, it ensures that the EXCB is built. If the EXCB is not built, the exit allocates storage for the EXCB. If the storage allocation fails, the exit sets a nonzero return code. There must be an EXCB for the USERVAR exit.

Processing a USERVAR

When VTAM invokes the USERVAR exit routine to process a USERVAR, the exit attempts to allocate a USERVAR control block (UVCB) in VTAM private storage to represent the USERVAR, if one has not already been allocated for that USERVAR. If the exit allocates the UVCB successfully, the exit initializes the UVCB and adds it to the UVCB queue. The EXCB contains the addresses of the beginning and end of the UVCB queue. Therefore, if a UVCB is the first entry in the queue, its address is stored in the EXCB. If a UVCB follows another entry in the queue, its address is stored in the preceding UVCB. When the exit is called for termination, it releases the storage for any UVCBs.

Sample USERVAR exit processing can continue even when the exit is unable to allocate a UVCB because a UVCB is a local copy of information and is not vital to processing. Although USERVARs can be added and updated without a UVCB, the exit attempts to build a UVCB for the USERVAR at the next opportunity. USERVAR translation does not use a UVCB at all.

Adding a USERVAR

When VTAM invokes the USERVAR exit routine because a USERVAR is being added, the exit attempts to allocate a UVCB for the USERVAR. If the exit allocates the UVCB successfully, the exit initializes the UVCB and adds it to the UVCB queue. If the exit does not allocate the UVCB successfully, the exit sets the return code to 0 and returns control. Likewise, if any of the processing required to add the USERVAR fails, the exit sets the return code to 0, returns control, and processing continues.

Updating a USERVAR

When VTAM invokes the USERVAR exit because a USERVAR is being updated, the exit overlays the value stored in the UVCB for that USERVAR with the new value. If there is no UVCB for the USERVAR, the exit attempts to create one. If the exit does not allocate the UVCB successfully, it sets the return code to 0 and returns control. Likewise, if any of the processing required to update the USERVAR fails, the exit sets the return code to 0 and the exit returns control.

Deleting a USERVAR

When VTAM invokes the USERVAR exit because a USERVAR is being deleted, the exit removes the UVCB for that USERVAR and frees the storage. If the exit cannot find the correct UVCB, the exit sets the return code to 0 and returns control. Likewise, if any of the processing required to delete the USERVAR fails, the return code is set to 0 and the exit returns control.

Translating a USERVAR

The logon manager defines its USERVARs as volatile. The sample USERVAR exit routine processes only volatile USERVARs so that the logon manager can ensure proper load balancing and maintain CLU session limits. When the sample exit receives a request to translate a static or dynamic USERVAR, the USERVAR exit turns off the translation flag and returns a nonzero return code to indicate that the session establishment request failed.

When the sample exit receives a request to translate a volatile USERVAR, the exit processes the request. The exit must first check the OLU status because translation processing when the OLU is the SLU differs from when the OLU is the PLU.

If the OLU is the SLU for the session, the sample USERVAR exit does not translate the name. The exit instead sets the value of the USERVAR equal to the generic USERVAR name. Although the name is not translated, the exit turns on the translation flag. The translation flag must be turned on to ensure that VTAM drives the logon manager through the logon exit.

If the OLU is the PLU for the session, the sample USERVAR exit processes the translation. All USERVAR translations occur in the logon manager host. The exit queries the logon manager data base by issuing the REQTAIL macro for function code X'04' processing. When called with function code X'04', the REQTAIL macro transfers control to the logon manager. The logon manager invokes the CLU search exit routine to compare all contending CLUs. After the CLU search exit has compared all contending CLUs, it identifies the best CLU for the session. The CLU search exit returns the name of the best CLU to the logon manager. The logon manager finds the application program LU name associated with the best CLU and returns that name to the REQTAIL macro. The REQTAIL macro returns the application program LU name to the USERVAR exit. The REQTAIL macro also returns a completion code indicating the search was successful.

After the successful return from the REQTAIL macro, the USERVAR exit does the following:

- Replaces the value it received from VTAM in the USERVAR parameter list with the name returned by the REQTAIL macro (the USERVAR table is not updated)
- Turns on the translation flag
- Sets the return code to 0
- Returns control to VTAM

CLU search exit routine

The CLU search exit routine allows you to customize the logon manager's load balancing capabilities to meet the needs of your installation. The exit is called for initialization at the completion of logon manager initialization. The logon manager provides information about the session and the contending CLUs to the exit. The logon manager invokes the exit to compare all contending CLUs and to select the best CLU to service the application program. There is one exception — if the session request is being tried again, the CLU search exit is not called for any CLUs that have already failed. A list of CLUs that have failed is presented in the REQTAIL RU.

If there is only one contending CLU, the exit is called with information about only that CLU. If there is more than one contending CLU, the exit compares them a pair at a time. In each iteration, the exit selects the better of the two CLUs to be the current best CLU. In the first iteration, the current best CLU is the first contending CLU the logon manager encounters. In each subsequent iteration, the exit compares the current best CLU from the previous iteration to another contending CLU. The exit continues to compare pairs of CLUs until there are no more contending CLUs. At that point, the exit returns the name of the better CLU from the final iteration to the logon manager as the best CLU. If no best CLU is found after the list of possible CLUs is exhausted, the session request fails.

In the sample CLU search exit, the logon manager ensures that the contending CLUs meet certain minimum requirements. These requirements vary, depending on whether the session request comes from a dependent SLU, an independent or dependent PLU, or TPF. For a discussion of the requirements, see [“Criteria for contending CLUs” on page 177](#).

The CLU search exit returns the name of the best CLU to the logon manger. For volatile USERVARs in PLU-initiated sessions, the logon manager finds the application program LU name associated with the best CLU and returns that name to the REQTAIL macro. The REQTAIL macro returns the name to the USERVAR exit.

The sample CLU search exit routine is stored in SYS1.SAMPLIB under the module name ELMCLUEx. If you write your own CLU search routine instead of using the sample, the following topics contain information you need. You might also want to see [Appendix G, “Sample CLU search exit routine for TPF sessions,”](#) on page 281.

Initial register contents

When the logon manager passes control to the CLU search exit routine, register contents are as follows:

Register 1:

Address of the CLU search exit parameter list

Register 13:

Address of a standard 72-byte save area

Register 14:

Return address

Register 15:

Address of the entry point of this routine

Final register contents

The CLU search exit routine must leave the register status as follows:

Register 1:

Address of the CLU search exit parameter list

Registers 2–14:

Restored to entry contents

Register 15:

Return code

The implications of the return codes vary, depending on the function of the exit. The return codes are described in [“Return codes”](#) on page 182.

Return codes

A nonzero return code returned during exit activation causes the activation to fail.

A nonzero return code returned during function code X'04' processing indicates to the logon manager that the CLU was not found and the session request has failed.

For all function codes other than X'04', the following return codes are valid:

X'00'

Processing successfully completed

X'01'–X'4F'

The session request has failed; the CLU search exit remains active

X'50'–X'7F'

Reserved

X'80'–X'EF'

The session request has failed; the CLU search exit is deactivated

X'F0'–X'FF'

Reserved

Design requirements

Errors within the routine could damage VTAM or system control blocks and modules. Follow these procedures when writing this routine:

- Use standard linkage.
- Save registers 2–14.

Consider the following restrictions when writing this routine:

- The name of the CLU search exit routine module must be ELMCLUX.
- This routine operates disabled in fixed storage. The routine gains control in supervisor state and with VTAM's storage key.
- There can be no system waits, including implied waits for I/O operations. System waits cause VTAM failure in some timing-dependent situations.
- This exit routine should use only conditional STORAGE invocations. You can reduce the possibility of a VTAM abend during a storage shortage by coding conditional STORAGE invocations.
- Data is addressable in 31-bit mode only. Data is always presented with 31-bit addressability.
- This exit routine can be above or below the 16M line.
- Assemble your routine and link-edit it into SYS1.VTAMLIB. See [“Installing VTAM exit routines” on page 173](#) for more information.
- The routine operates as an internal VTAM routine. Therefore, VTAM performance might be degraded if the routine requires lengthy processing time. While this routine is being executed, no new VTAM operator requests or requests to initiate or terminate sessions are processed by VTAM for any resource.
- This exit routine must be reentrant.

CLU search exit parameter descriptions

The parameters that the logon manager passes to the CLU search exit routine are described in this topic. There is a parameter list header that is common to all functions of the exit. The header contains a function code that indicates the reason the exit is being invoked. For all function codes, a set of function-specific parameters follows the parameter list header. The format of the header is described in [Table 117 on page 184](#).

Table 117. CLU search exit parameter list header

Dec (Hex) offset	Size (Bytes)	Description
0 (0)	4	Function code X'01' Exit initialization. X'02' A CLU session is being initiated and the CLU is available. X'03' An application program has become available. X'04' Two CLUs are to be compared. X'05' An application program is no longer available. X'06' A CLU session is being terminated and the CLU is no longer available. X'07' Exit termination
4 (4)	4	Logon manager vector table user extension control block pointer. This pointer allows the exit to allocate a global user extension control block during function code X'01' processing. The exit can use this area for building control structures that are not associated with any specific CLU. If the address of the control block is passed to the logon manager in this field, the logon manager saves the pointer and passes it back to the exit whenever the exit is called. On input to user code, this address is 0 for function code X'01'. Otherwise, the pointer is passed by the caller on every invocation.

Function code parameter lists

The parameter list data field starts at offset 8 following the parameter list header. The parameters vary, depending on the function code. The amount of storage allocated for each parameter list is the amount required for the function code X'04' parameter list, no matter what the actual length of the parameter list. A parameter list for another function code overlays the parameter list for function code X'04'.

A format for each function's parameter list is described in the topics that follow.

Note: Source code for the parameter lists is within ELMCLUPL in SYS1.SISTMAC1.

Exit initialization and termination (function codes X'01' and X'07')

The following parameters described in [Table 118 on page 185](#) follow the CLU search exit parameter list header for exit initialization and termination.

Table 118. Function codes X'01' and X'07' parameters

Dec (Hex) offset	Size (Bytes)	Description
8 (8)	2	<p>The size, in bytes, of the logon manager vector table user extension control block</p> <p>If the exit allocates a logon manager vector table user extension control block, it must return the size of the control block in this field. If the logon manager shuts down and is unable to call the exit for function code X'07' processing, the logon manager uses this size information to free the control block. With normal function code X'07' processing, if the pointer is not NULL, the exit is responsible for freeing this control block and its pointer.</p>
10 (A)	2	Reserved

CLU session initiation and termination (function codes X'02' and X'06')

The following parameters described in Table 119 on page 185 follow the CLU search exit parameter list header when a CLU becomes available or is no longer available. A CLU becomes available when the logon manager establishes a session with it. A CLU is no longer available after the logon manager terminates the session.

Table 119. Function codes X'02' and X'06' parameters

Dec (Hex) offset	Size (Bytes)	Description
8 (8)	4	Reserved
12 (C)	8	The name of the CLU that has become available or is no longer available
20 (14)	4	<p>CLU user extension control block pointer</p> <p>This pointer allows the exit to allocate a control block associated with the CLU named above. For function code X'02' processing, the logon manager saves the pointer to this control block and passes it to the exit whenever the exit is called with this same CLU name. When called for function code X'06' processing, if the pointer is not NULL, the exit is responsible for freeing this control block and its pointer.</p>
24 (18)	2	The length, in bytes, of the control block user extension area
26 (1A)	2	Reserved

Application program activation and deactivation (function codes X'03' and X'05')

The following parameters described in Table 120 on page 186 follow the CLU search exit parameter list header when a TPF application program becomes available or when the application program is no longer available. A TPF application program becomes available when its MINLINK count is matched or exceeded. A TPF application program is no longer available when its MINLINK count is no longer matched or exceeded.

Table 120. Function codes X'03' and X'05' parameters

Dec (Hex) offset	Size (Bytes)	Description
8 (8)	4	Reserved
12 (C)	8	The name of the application program that has become available or is no longer available
20 (14)	4	Application program user extension control block pointer This pointer allows the exit to allocate a control block associated with the application program named above. For function code X'03' processing, the logon manager saves the pointer to this control block and passes it to the exit whenever the exit is called with this same application program name. When called for function code X'05' processing, if the pointer is not NULL, the exit is responsible for freeing this control block and its pointer.
24 (18)	2	The size, in bytes, of the application program user extension control block
26 (1A)	2	Reserved
28 (1C)	4	The number of links currently supporting the application program
32 (20)	12	Reserved

CLU comparison (function code X'04')

The following parameters follow the CLU search exit parameter list header when the logon manager receives a session request and needs to compare contending CLUs to determine which CLU is best to service the session. See [“Criteria for contending CLUs” on page 177](#) for more information about the selection algorithm the sample exit uses to choose the best CLU.

If there is only one contending CLU, the exit is called with information about only that CLU. If there is more than one contending CLU, the exit compares them a pair at a time. In each iteration, the exit selects the better of the two CLUs to be the current best CLU. In the first iteration, the current best CLU is the first contending CLU the logon manager encounters. In each subsequent iteration, the exit compares the current best CLU from the previous iteration to another contending CLU. The exit continues to compare pairs of CLUs until there are no more contending CLUs. At that point, the exit returns the name of the better CLU from the final iteration to the logon manager as the best CLU. If no best CLU is found after the list of possible CLUs is exhausted, the session request fails.

Within the function code X'04' parameters, the description of the contending CLU is always presented between hex 48 and hex 7F. The description of the current best CLU is always presented between hex 80 and hex B8. If the exit determines the contending CLU is the better of two CLUs, the contender CLU's values are moved to hex 80 and it becomes the current best CLU. If only one CLU is presented to the CLU search exit, the description of that CLU is presented between hex 80 and hex B8. If two CLUs are equal in comparison, there is no move and the CLU search exit chooses the CLU described between hex 80 and hex B8.

Table 121. Function code X'04' parameters

Dec (Hex) offset	Size (Bytes)	Description
8 (8)	4	Reserved
12 (C)	1	Reserved

Table 121. Function code X'04' parameters (continued)

Dec (Hex) offset	Size (Bytes)	Description
13 (D)	1	<p>Retry indicator</p> <p>B'1...' If this indicator is turned on, the session request is being tried again.</p> <p>The remaining bits are reserved.</p>
14 (E)	2	Reserved
16 (10)	8	If the retry indicator is turned on, this is the name of the CLU that was unable to establish a session in the most recent session attempt.
24 (18)	8	Class-of-service name for this session
32 (20)	8	Reserved
40 (28)	8	<p>Session partner name</p> <p>This is the name of the resource that will be in session with this TPF. This is the DLU if the session request was initiated by TPF; otherwise, this is the OLU.</p>
48 (30)	4	<p>Session partner subarea</p> <p>This is the subarea address of the boundary function for the session partner as known in the network where the logon manager resides.</p>
52 (34)	8	Generic TPF application program name
60 (3C)	4	Reserved
64 (40)	4	Application program user extension control block pointer
68 (44)	4	The number of links currently supporting the application program
72 (48)	8	The name of the new contending CLU
80 (50)	8	Reserved
88 (58)	8	The name of the application program LU
96 (60)	4	<p>Parallel session count</p> <p>This is the number of active parallel sessions currently serviced by the contending CLU.</p>
100 (64)	4	<p>CLU user extension control block pointer</p> <p>This pointer provides access to the control block for the contending CLU.</p>
104 (68)	4	<p>Session count</p> <p>This is the number of active sessions currently serviced by the contending CLU.</p>

Table 121. Function code X'04' parameters (continued)

Dec (Hex) offset	Size (Bytes)	Description
108 (6C)	4	Sessions pending This is the number of pending sessions to be serviced by the contending CLU.
112 (70)	4	Session limit This is the maximum number of sessions the contending CLU can support.
116 (74)	4	Hop count This is the number of hops from the OLU to TPF.
120 (78)	4	Subarea address of the contending CLU's boundary function
124 (7C)	1	Host of the contending CLU
125 (7D)	3	Reserved
128 (80)	8	The name of the current best CLU
136 (88)	8	Reserved
144 (90)	8	The name of the application program LU
152 (98)	4	Parallel session count This is the number of active parallel sessions currently serviced by the current best CLU.
156 (9C)	4	CLU user extension control block pointer This pointer provides access to the control block for the current best CLU.
160 (A0)	4	Session count This is the number of active sessions currently serviced by the current best CLU.
164 (A4)	4	Sessions pending This is the number of pending sessions to be serviced by the current best CLU.
168 (A8)	4	Session limit This is the maximum number of sessions the current best CLU can support.
172 (AC)	4	Hop count This is the number of hops from the OLU to TPF.
176 (B0)	4	Subarea address of the current best CLU's boundary function
180 (B4)	1	Host of the current best CLU
181 (B5)	3	Reserved

Note: If you create any of the user extension control blocks (logon manager vector table user extension, CLU user extension, or application program user extension), begin the control block with a 4-byte reserved field followed by a 2-byte length field.

REQTAIL macroinstruction

To invoke the logon manager and obtain the name of the best CLU for a PLU-initiated session specifying a volatile USERVAR, the USERVAR exit routine invokes the REQTAIL macroinstruction with function code X'04'. The USERVAR exit is responsible for allocating pageable common storage for the REQTAIL macroinstruction parameter list before invoking the macroinstruction. If the storage allocation fails, the USERVAR exit turns off the translation flag and sets the return code to indicate the resource was not found. The return code should not request that VTAM deactivate the exit or delete the USERVAR. See “Final register contents” on page 111 for a description of return codes from the exit.

The USERVAR exit routine is also responsible for freeing storage used by the REQTAIL macro. If the exit is unable to free the storage, the exit sets the return code to 0 and returns control.

Note: Source code for this parameter list is within ELMCUVPL in SYS1.SISTMAC1.

Table 122. REQTAIL macro parameter list for function code X'04'

Dec (Hex) offset	Size (Bytes)	Description
0 (0)	4	Function code
4 (4)	4	Reserved
8 (8)	8	Reserved
16 (10)	8	Session partner name
24 (18)	4	Session partner subarea address
28 (1C)	8	Name of the requesting CLU
36 (24)	4	Session partners list pointer (see Table 123 on page 191)
40 (28)	8	Reserved
48 (30)	8	Target application program LU
56 (38)	8	Class of service name
64 (40)	8	Generic TPF application program name
72 (48)	1	Host of the requesting CLU

Table 122. REQTAIL macro parameter list for function code X'04' (continued)

Dec (Hex) offset	Size (Bytes)	Description
73 (49)	1	<p>Flags</p> <p>B'10..' The OLU is the SLU.</p> <p>B'01..' The OLU is the PLU.</p> <p>B'..0.' The CLU was not found and the DLU name was not translated.</p> <p>B'..1.' The CLU was found and the DLU name was translated.</p> <p>B'...0' The session was established through the logon manager.</p> <p>B'...1' The session was established through the USERVAR exit.</p> <p>The remaining bits are reserved.</p>
74 (4A)	2	Reserved
76 (4C)	4	REQTAIL RU pointer, if any
80 (50)	4	Pointer to the control block for the best CLU
84 (54)	4	Pointer to the USERVAR exit control block (EXCB)

If there is a successful return from the REQTAIL macro, the REQTAIL macro parameter list returns the application program LU name associated with the best CLU to the USERVAR exit. The application program LU name is in the control block for the best CLU. The USERVAR exit then replaces the value VTAM passed in the USERVAR translation parameters with the application program LU name, sets the return code to 0, and returns control to VTAM.

The REQTAIL invocation fails if any of the following conditions occur:

- The CLU search exit is not active.
- The logon manager is not available.
- The cross-memory initialization failed.
- The CLU was not found.
- The application program was not found.

If the REQTAIL invocation fails, the USERVAR return code indicates the resource was not found and the translation flag indicates the USERVAR was not translated.

Session partners list

The REQTAIL macro parameter list contains a pointer to the session partners list. There is an entry in the list for each session partner with which the OLU is currently in session. The format of the list is shown in the following table.

Table 123. Session partners list

Dec (Hex) offset	Size (Bytes)	Description
0 (0)	2	Number of parallel sessions between this session partner and the OLU
2 (2)	2	Reserved
4 (4)	8	Reserved
12 (C)	8	Name of this session partner
20 (14)	4	Pointer to the next entry in the session partner list. This pointer is 0 if there are no more entries.

Chapter 3. Writing TSO/VTAM installation-wide exit routines

This topic contains information you need to write exit routines for TSO/VTAM. You can write routines to do the following:

- Perform input and output editing that replaces or supplements IBM-supplied editing.
- Perform attention handling that replaces IBM-supplied attention handling.
- Provide support for terminals not supported by TSO/VTAM.

Note: To support terminals not supported by TSO/VTAM, write your own terminal input manager and terminal output manager, in addition to exit routines IKTGETXT, IKTINX2, IKTCASX1, and IKTINX1.

- Verify that a specified terminal type can receive TSO/VTAM user messages in the language requested by the terminal user.

Note: TSO/VTAM exit routines are optional. However, if you do not write these exit routines, you will get unresolved external reference messages during link-editing of the load modules that call them. To determine whether you need any of the input and output editing exit routines, see the topics on the TPUT and TGET options in [z/OS TSO/E Programming Services](#).

TSO/VTAM exit routines

At appropriate points during VTAM terminal I/O coordinator (VTIOC) and terminal control address space (TCAS) processing, VTAM checks to determine whether a particular exit routine exists. If the exit routine exists, VTAM calls the routine. If the exit routine does not exist, normal processing continues.

This topic describes the routines in alphabetical sequence and states the required input and output for each routine. Before you use one of the routines, link-edit it with the object module that calls it by performing the following steps:

1. Edit JCLIN (with the current FMID) and find the load module listed in [Table 128 on page 203](#).
2. Find the last INCLUDE card before the NAME card found in your search for that particular load module.
3. Duplicate the INCLUDE card and change the name to the exit routine.
4. Assemble your exit and place the object deck into the distribution library specified on the INCLUDE card.

For example, in the case of IKTRTX1, you should be link-editing into IKTIOM01. To code the link-edit control statements to activate the IKTRTX1 exit, you should edit JCLIN (with the current FMID) and add the following after the INCLUDE AOST3(IKTWTTYO) card: INCLUDE AOST3(IKTRTX1).

Notes:

- Be sure the INCLUDE begins in column 2. Then add the assembled version of the IKTRTXI exit to the AOST3 distribution library.
- All TSO/VTAM exit routines run in 24-bit addressing mode.

The exit routines are summarized in [Table 124 on page 193](#), [Table 125 on page 194](#), and [Table 126 on page 194](#).

Table 124. Summary of the TSO/VTAM exit routines for VTIOC processing

Name	Purpose	Terminal type	Caller
IKTGETXT	Edit input data	Unsupported	IKTVTGET

Table 124. Summary of the TSO/VTAM exit routines for VTIOC processing (continued)

Name	Purpose	Terminal type	Caller
IKTIDSX1	Replace or supplement IBM-supplied output editing	3270	IKT3270O
IKTIDSX2	Supplement IBM-supplied input editing	3270	IKT3270I
IKTIDSX3	Supplement IBM-supplied attention handling	3270	IKT3270I
IKTIDSX4	Replace or supplement IBM-supplied input editing	3270	IKTVTGET
IKTINX2	Initialize user-written I/O managers	Unsupported	IKTXINIT
IKTRTX1	Replace or supplement IBM-supplied output editing	3767/3770 2741	IKT3767O
IKTRTX2	Supplement IBM-supplied input editing	3767/3770 2741 WTTY TWX	IKT3767I
IKTRTX3	Replace IBM-supplied attention handling	3767/3770 (LU1)	IKTIMLU1
IKTRTX4	Replace or supplement IBM-supplied input editing	3767/3770 (LU1)	IKTVTGET
IKTWTX1	Replace or supplement IBM-supplied output editing	TWX WTTY	IKTWTTYO

Table 125. Summary of the TSO/VTAM exit routines for TCAS processing

Name	Purpose	Terminal type	Caller
IKTCASX1	Replace or supplement IBM-supplied logon error messages	Unsupported	IKTCAS31 IKTCASOY
IKTINX1	Set terminal type and buffer size	Unsupported	IKTCAS23

Table 126. Summary of the TSO/VTAM exit routines for VTIOC and TCAS processing

Name	Purpose	Terminal type	Caller
IKTCASX2	Verify language of a message for a terminal	All	IKTMSIFR

IKTCASX1: Error handling for unsupported terminals

Write this routine if you want to build a logon failure message (messages IKT00200I through IKT00204I) for a terminal that is not supported by TSO/VTAM. IKTCAS31 or IKTCASOY calls IKTCASX1 before TSO/VTAM sends the error message to the terminal. This gives the exit an opportunity to build its own message, including both text and device control characters, up to 160 characters. The message text must be located in the message buffer provided in Register 0 on entry.

Input to IKTCASX1

Register 0:

Address of the buffer where the message will be built

The buffer size is equal to the total size of the following, up to a maximum of 160 characters:

- Size of the message text
- A 4-byte length field for each line of text in the message
- Eight bytes for control characters

Register 1:

Address of the message definition

This message can be the default IBM-supplied TSO/VTAM message or a TSO/VTAM message that is defined to the MVS Message Service (MMS) for a particular language.

Each line of message text is preceded by a 4-byte field. The end of the message is indicated by a final 4-byte field set to 0. The format of the message is shown in the following figure:

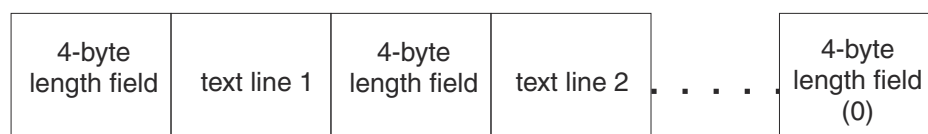


Figure 3. Format of the IKTCASX1 message

Register 10:

Address of the IKTWESTD control block (see [“IKTWESTD” on page 205](#)). (The terminal type is defined in the WETMTP field of the control block.)

Locks:

None

Processing mode:

TCB

Output from IKTCASX1

Register 1:

Length of new message (including device control characters and text). There is a maximum of 160 characters in the message buffer.

Register 15:

Return code:

0

TSO/VTAM constructs the message.

Nonzero

TSO/VTAM uses the message constructed by the exit. This message is located in the message buffer pointed to by register 0 at input.

IKTCASX2: User message language-hardware verification

This routine allows you to verify that a terminal can receive a TSO/VTAM terminal user message in the language requested by the terminal user. The requested language is specified in the PLANG operand on the PROFILE command, or is passed on the CINIT during session initiation. For example, if a terminal user wants messages in French, but the terminal cannot display the French characters that define the message in the MVS Message Service (MMS) file, the exit rejects the requested language and chooses a valid substitute. For a complete list of valid languages, DEVLANG values, and language code settings, see [z/OS Communications Server: SNA Programming](#).

Input to IKTCASX2

Register 1 contains the address of the following parameter list:

Table 127. IKTCASX2 input and output parameter list

Dec (Hex) offset	Size (Bytes)	Description
0 (0)	3	Input: The requested language from the TSO user IDs User Profile Table (UPT). The TSO user can specify the language by using the PLANG operand on the PROFILE command. If this field is blank, the requested language is specified in the field below (this is true for error messages IKT00200I through IKT00204I).
3 (3)	1	Input: The language code passed on the CINIT during session initiation. This value comes from the terminal user's logon request. For the logon request, the value can be retrieved from the USS LANG operand, the USS LANGTAB second suboperand, or from the MODEENT LANG value specified by the USS LOGMODE operand. The values are retrieved and verified in a hierarchical order. See z/OS Communications Server: SNA Resource Definition Reference for a discussion about this hierarchy.
4 (4)	4	Input: The CGCSGID for the single-byte character set (SBCS) for the terminal to which TSO/VTAM is sending the message. This is obtained from the Query Reply (character sets). See <i>3174 Establishment Controller Network Guide</i> for more information about this value. This field is 0 if the information is not available. For example, if the device was not queried, this field is 0.
8 (8)	4	Input: The CGCSGID for the double-byte character set (DBCS) for the terminal to which TSO/VTAM is sending the message. This is obtained from the Query Reply (character sets). See <i>3174 Establishment Controller Network Guide</i> for more information about this value. This field is 0 if the information is not available. For example, if the device was not queried, this field is 0.
12 (C)	1	Input: The TSO/VTAM terminal type: 1 = LU type 0 2 = LU type 1 4 = LU type 2
13 (D)	3	Output: If this exit sets the return code to 0, the language code specified in this field is used in place of the language requested by the terminal user.

Locks:

None

Processing mode:

TCB or SRB

Output from IKTCASX2

Return code:

0

The exit overrides the language requested by the terminal user. VTAM issues the message in the language returned in the IKTCASX2 input/output parameter list.

Nonzero

The exit approves the language requested by the terminal user. VTAM issues the message in the language requested by the terminal user.

IKTGETXT: Editing on unsupported terminals

Write this routine if you want to use a terminal not supported by TSO/VTAM. IKTVTGET calls IKTGETXT instead of using the IBM-supplied code at statement label EDIT3270 (for 3270 terminals) or EDIT3767 (for 3767 and 3770 terminals) in IKTVTGET. IKTGETXT must scan the input data, edit it, and move the edited data from the input queue to the TGET requester's data area.

Input to IKTGETXT

Register 0:

Address of IKTXSA (see [“IKTXSA” on page 208](#))

Register 1:

Address of IKTIARM (see [“IKTIARM” on page 203](#))

Locks:

Local memory lock

Processing mode:

TCB

Output from IKTGETXT

None

IKTIDSX1: Output editing for IBM 3270 terminals

You can write this routine to perform 3270 output editing in place of or in addition to that performed by the IBM-supplied routine IKT32700. This routine's editing occurs before IKT32700 moves the data from the output queue into the output request unit.

Input to IKTIDSX1

Register 1:

Address of IKTOPARM (see [“IKTOPARM” on page 204](#))

OPACBUFA:

Current buffer address

OPACBUFL:

Current buffer length

Locks:

Local memory lock

Processing mode:

SRB

Output from IKTIDSX1

Register 15:

Return code:

0

IBM-supplied routine IKT32700 should perform editing; data is still on the output queue.

Nonzero

Exit routine performed all editing; portions of IKT32700 are bypassed.

IKTIDSX2: Input editing for IBM 3270 terminals

You can write this routine to perform input scanning and editing in addition to that performed by the IBM-supplied routine IKT3270I. If provided, IKTIDSX2 is called after the data is translated (if necessary) from ASCII code to EBCDIC, but before it is scanned for input line delimiters. It is then broken into line segments and placed on the input queue.

Input to IKTIDSX2

Register 1:

Address of IKTMPL (see [“IKTMPL” on page 204](#))

Locks:

Local memory lock

Processing mode:

SRB

Output from IKTIDSX2

MPLXTA:

Address of available input data

MPLXTL:

Length of available input data

IKTIDSX3: Attention handler for IBM 3270 terminals

You can write this routine to handle attention interruptions from 3270 terminals during input editing instead of using the IBM-supplied routine IKTATTN. One use of a user-coded attention handler routine is to clear the queues conditionally (rather than unconditionally) when an attention interruption is received.

Input to IKTIDSX3

PSAAOLD:

Address of ASCB

ASCBTSB:

Address of TSB

TSBEXTNT:

Address of TSB extension

TSBXTVWA:

Address of TVWA

Locks:

Local memory lock

Processing mode:

SRB

Output from IKTIDSX3

None

IKTIDSX4: TGET edit for IBM 3270 terminals

You can write this routine to perform 3270 editing in place of or in addition to that performed by the IBM-supplied routine IKTVTGET. IKTVTGET scans for data that is not valid and 3270 control characters,

and moves the data from the input queue to the TGET requester's data area. You might write an edit exit routine to change TGET EDIT editing criteria.

Input to IKTIDSX4

Register 0:

Address of IKTXSA (see [“IKTXSA” on page 208](#))

Register 1:

Address of IKTIPARM (see [“IKTIPARM” on page 203](#))

Locks:

Local memory lock

Processing mode:

TCB

Output from IKTIDSX4

Register 15:

Return code:

X'00'

Exit routine performed the entire edit operation and moved the input data to the TGET data area.

X'04'

Exit routine performed only a data scan; the IBM-supplied code (EDIT3270) should perform editing.

IKTINX1: Logon edit

Write this routine if you want to use a terminal not supported by TSO/VTAM. IKTCAS23 calls IKTINX1, if provided, when a logon request is encountered from a terminal other than an IBM 3270, 3767, or 3770. (If TSO/VTAM is used with NTO, the 2741, WTTY, and TWX Models 33 and 35 are also supported.) IKTINX1 must verify that the terminal is supported by user-written routines (that is, a terminal input manager, a terminal output manager, and edit routine IKTGETXT), and it must provide the terminal type of X'03', the buffer size, and the device bind image.

Input to IKTINX1

Register 1:

Address of a parameter list containing:

- Address of the RPL
- Address of WETMTP (terminal type) (see [“IKTWESTD” on page 205](#))
- Address of WETMBF (terminal buffer size) (see [“IKTWESTD” on page 205](#))
- Address of WEBIND (bind image) (see [“IKTWESTD” on page 205](#))

Locks:

None

Processing mode:

TCB

Output from IKTINX1

Register 15:

Return code:

X'00'

Recognized terminal type; logon processing continues

X'04'

Unrecognized terminal type; logon processing terminates

IKTINX2: I/O manager initialization

Write this routine if you want to use terminal input managers and terminal output managers you have written. IKTINX2 should perform the same function for user-coded input and output managers that module IKTIOM performs for the IBM-supplied I/O managers. IKTINX2 allocates storage for and initializes the I/O manager SRBs.

Input to IKTINX2

Locks:

Local memory lock

Processing mode:

TCB

Output from IKTINX2

Register 15:

Return code:

0

Successful initialization

Nonzero

Unsuccessful initialization; logon processing terminates

IKTRTX1: Output edit for IBM 3767, 3770, and 2741 terminals

You can write this routine to perform 3767, 2741, or 3770 output editing in place of or in addition to that performed by the IBM-supplied routine IKT3767O. This routine's editing occurs before IKT3767O moves the data from the output queue to the output request unit. The IBM-supplied code scans data, edits it according to specified TPUT operands (EDIT, ASIS, or CONTROL), and provides user-specified character translation.

Input to IKTRTX1

Register 1:

Address of IKTOPARM (see [“IKTOPARM” on page 204](#))

OPACBUFA:

Current buffer address

OPACBUFL:

Current buffer length

Locks:

Local memory lock

Processing mode:

SRB

Output from IKTRTX1

Register 15:

Return code:

0

IBM-supplied code should perform editing; data is still on the output queue.

Nonzero

Exit routine performed all editing; the IBM-supplied code is bypassed.

IKTRTX2: Input edit for IBM 3767 and 3770 terminals

You can write this routine to perform input scanning and editing in addition to that performed by the IBM-supplied routine IKT3767I. If provided, IKTRTX2 is called after the data is translated (if necessary) from ASCII code to EBCDIC and the user-supplied character translation is performed on the data, but before the data is scanned for input line delimiters, broken into single lines, and placed on the input queue.

Input to IKTRTX2

Register 1:

Address of IKTMPL (see [“IKTMPL” on page 204](#))

MPLTXTA:

Address of buffer

MPLTXTL:

Length of buffer

Locks:

Local memory lock

Processing mode:

SRB

Output from IKTRTX2

MPLTXTA:

Address of buffer

MPLTXTL:

Length of buffer

IKTRTX3: Attention handler for IBM 3767 and 3770 terminals

You can write this routine to handle attention interruptions from 3767 or 3770 terminals instead of using the IBM-supplied routine IKTATTN. One use of a user-coded attention handler routine is to clear the queues conditionally (rather than unconditionally) when an attention interruption is received.

Input to IKTRTX3

Register 13:

Standard 72-byte save area

Locks:

Local memory lock

Processing mode:

SRB

Output from IKTRTX3

None

IKTRTX4: Edit for IBM 3767, 3770, and 2741 terminals

You can write this routine to perform 3767 or 3770 editing in place of or in addition to that performed by the IBM-supplied routine IKTVTGET. IKTVTGET scans for data that is not valid and moves the data from the input queue to the TGET requester's data area. You might write an edit exit routine to change TGET EDIT editing criteria.

Input to IKTRTX4

Register 0:

Address of IKTXSA (see [“IKTXSA” on page 208](#))

Register 1:

Address of IKTIPARM (see [“IKTIPARM” on page 203](#))

Locks:

Local memory lock

Processing mode:

TCB

Output from IKTRTX4

Register 15:

Return code:

X'00'

Exit routine performed the entire edit operation and moved the input data to the TGET data area.

X'04'

Exit routine performed only a data scan; the IBM-supplied code (EDIT3767) should perform editing.

IKTWTX1: Output edit for WTTY and TWX terminals

You can write this routine to perform TWX or WTTY output editing in place of or in addition to that performed by the IBM-supplied routine IKTWTYIO. The IBM-supplied code scans data, edits it according to the TPUT operands specified (EDIT, ASIS, or CONTROL), and provides user-specified character translation.

Input to IKTWTX1

Register 1:

Address of IKTOPARM (see [“IKTOPARM” on page 204](#))

OPACBUFA:

Current buffer address

OPACBUFL:

Current buffer length

Locks:

Local memory lock

Processing mode:

SRB

Output from IKTWTX1

Register 15:

Return code:

0

Continue with normal editing.

Nonzero

No further editing.

Installing TSO/VTAM exit routines

Link-edit exit routines you have coded with the object modules that call them. [Table 128 on page 203](#) shows the calling module and load module for each exit routine.

Table 128. Routine-module cross-reference

Name	Calling module	Load module
IKTCASX1	IKTCAS31	IKTCAS30
IKTCASX2	IKTMSIFR	IKTMSIFR
IKTGETXT	IKTVTGET	IGC0009C
IKTIDSX1	IKT32700	IKTIOM02
IKTIDSX2	IKT3270I	IKTIOM02
IKTIDSX3	IKT3270I	IKTIOM02
IKTIDSX4	IKTVTGET	IGC0009C
IKTINX1	IKTCAS23	IKTCAS20
IKTINX2	IKTXINIT	IKJEFLA
IKTRTX1	IKT37670	IKTIOM01
IKTRTX2	IKT3767I	IKTIOM01
IKTRTX3	IKTIMLU1	IKTIOM01
IKTRTX4	IKTVTGET	IGC0009C
IKTWTX1	IKTWTTYO	IKTIOM01

Control blocks

The following control blocks are required input for some of the exit routines. The descriptions of the exits reference the applicable control blocks.

IKTIPARM

Table 129. IKTIPARM control block

Offsets					
Dec	Hex	Type	Len	Name (Dim)	Description
0	(0)	STRUCTURE	*	IKTIPARM	QUEUE MANAGER INPUT PARAMETER LIST
0	(0)	CHARACTER	10	FIXED_PART	
0	(0)	ADDRESS	4	IPACBUFA	ADDRESS OF CURRENT BUFFER
4	(4)	SIGNED	2	IPACBUFL	LENGTH OF CURRENT BUFFER
6	(6)	SIGNED	2	IPACHDRL	LENGTH OF HEADING CHARACTERS
8	(8)	CHARACTER	2	IPACFLAG	CURRENT BUFFER FLAGS
8	(8)	CHARACTER	1	*	FLAG BYTE
		1...		IPACGETM	BUFFER HAS BEEN GETMAINED
		.111 1111		*	RESERVED
9	(9)	CHARACTER	1	*	FLAG BYTE
		1...		IPACMSGGA	MESSAGE IS AVAILABLE
		.1...		IPACMSGP	MESSAGE PROMPTED FOR

Table 129. IKTIPARM control block (continued)					
Offsets					
Dec	Hex	Type	Len	Name (Dim)	Description
		..1.		IPACPRTL	PARTIAL MESSAGE
		...1		IPACPASS	PASS THROUGH MESSAGE
	 1...		IPACTGET	TGET WAIT
	111		*	RESERVED
10	(A)	CHARACTER	*	IPACHDRT	HEADING CHARACTERS (PASSED ONLY WHEN ADDING TO QUEUE)

IKTMPL

Table 130. IKTMPL control block					
Offsets					
Dec	Hex	Type	Len	Name (Dim)	Description
0	(0)	STRUCTURE	12	IKTMPL	DEVICE MAPPING MANAGER PARAMETER LIST
0	(0)	CHARACTER	1	MPLFC	FUNCTION CODE
1	(1)	CHARACTER	1	MPLFM	FUNCTION MODIFIER
2	(2)	CHARACTER	2	MPLFLAGS	FLAGS
2	(2)	CHARACTER	1	MPLFLAG0	FLAG BYTE 0
3	(3)	CHARACTER	1	MPLFLAG1	FLAG BYTE 1
		1111 11..		*	RESERVED
	11		MPLCHAIN	CHAINING FLAGS
	1.		MPLCHBEG	+ BEGINNING OF CHAIN
	1		MPLCHEND	+ END OF CHAIN
4	(4)	ADDRESS	4	*	RESERVED
8	(8)	SIGNED	4	*	RESERVED

IKTOPARM

Table 131. IKTOPARM control block					
Offsets					
Dec	Hex	Type	Len	Name (Dim)	Description
0	(0)	STRUCTURE	10	IKTOPARM	QUEUE MANAGER OUTPUT PARM LIST
0	(0)	ADDRESS	4	OPACBUFA	A(CURRENT BUFFER)
4	(4)	SIGNED	2	OPACBUFL	L(CURRENT BUFFER)
6	(6)	SIGNED	2	*	RESERVED

Table 131. IKTOPARM control block (continued)					
Offsets					
Dec	Hex	Type	Len	Name (Dim)	Description
8	(8)	CHARACTER	2	OPACFLAG	CURRENT BUFFER FLAGS
8	(8)	CHARACTER	1	*	FLAG BYTE
		1... ..		OPACGETM	BUFFER HAS BEEN GETMAINED
		.1.. ..		OPACTOPQ	ELEMENT ADDED TO TOP OF QUEUE
		..1.		OPACPASS	PASS THROUGH MESSAGE
		...1		OPACREAD	DATA CAUSES DEVICE INPUT
	 1...		OPACFLSH	MESSAGE IS FLASHBACK DATA
	1..		OPACUSEG	USING USER'S GETMAINED BUFFER AREA
	11		*	RESERVED
9	(9)	CHARACTER	1	*	FLAG BYTE
		1... ..		OPACHOLD	TPUT HOLD INDICATOR
		.1.. ..		OPACBKIN	TPUT BREAKIN INDICATOR
		..11		OPACOPTN	OPTION FLAG BYTE
	 1...		OPACASID	TPUT ASID INDICATOR
	111		*	RESERVED

IKTWESTD

Table 132. IKTWESTD control block					
Offsets					
Dec	Hex	Type	Len	Name (Dim)	Description
0	(0)	STRUCTURE	*	WESTD	STANDARD WORK ELEMENT
0	(0)	CHARACTER	252	FIXED_PART	
0	(0)	BITSTRING	2	WECODE1	PRIMARY CODE
0	(0)	BITSTRING	1	WECODE1T	TASK CODE
1	(1)	BITSTRING	1	WECODE1F	FUNCTION CODE
		1111		WECDMAF	MAJOR FUNCTION CODE
	 1111		WECDMIF	MINOR FUNCTION CODE
2	(2)	BITSTRING	2	WECODE2	SECONDARY CODE
2	(2)	BITSTRING	1	WECODE2T	SECONDARY TASK CODE
3	(3)	BITSTRING	1	WECODE2F	FUNCTION CODE
		1111		WEC2MAF	MAJOR
	 1111		WEC2MIF	MINOR
4	(4)	ADDRESS	4	WENEXT	NEXT WORK ELEMENT POINTER

Table 132. IKTWESTD control block (continued)

Offsets					
Dec	Hex	Type	Len	Name (Dim)	Description
8	(8)	BITSTRING	2	WERC	FAILURE REASON CODE FOR MESSAGES IKT0002I, IKT0003D
8	(8)	BITSTRING	1	WERC1	
9	(9)	BITSTRING	1	WERC2	TCAS TERMINATION REQUEST CODE 04 - DETACH SUBTASKS 02 - FREE STORAGE 01 - ISSUE DUMP
10	(A)	SIGNED	2	WELEN	WORK ELEMENT LENGTH IF VAR
12	(C)	ADDRESS	4	WEUASCB	USER ASCB ADDRESS
16	(10)	ADDRESS	4	WEVNIB	VTAM NIB ADDRESS
20	(14)	ADDRESS	4	WEUECB	USER TERMINATE ECB ADDRESS
24	(18)	ADDRESS	4	WEVRPL	VTAM RPL ADDRESS
28	(1C)	ADDRESS	4	WELBUF	LOGON BUFFER ADDRESS
32	(20)	CHARACTER	4	WETERM	TERMINAL CHARACTERISTICS
32	(20)	UNSIGNED	1	WETMTP	TERMINAL TYPE BYTE
33	(21)	UNSIGNED	1	WESUBT	TERMINAL SUBTYPE FIELD
34	(22)	SIGNED	2	WETMBF	TERMINAL BUFFER SIZE
36	(24)	BITSTRING	1	WETMFLG	TERMINAL FLAGS
		1...		WEFLG1	ASCII INDICATOR FOR TSB INIT
		.1..		WENOSW	NO SCREEN SWITCHING
		..11		WEASCC	ASCII CODE IDENTIFIER
	 1...		WEQSCR	QUERY FOR SCREEN SIZE
	111		*	RESERVED
37	(25)	UNSIGNED	1	WERUSZI	TERMINAL INPUT RU SIZE
38	(26)	UNSIGNED	1	WERUSZO	TERMINAL OUTPUT RU SIZE
39	(27)	UNSIGNED	1	*	RESERVED
40	(28)	BITSTRING	36	WEBIND	BIND IMAGE
76	(4C)	UNSIGNED	1	WEPRMR	PRIMARY ROW VALUE
77	(4D)	UNSIGNED	1	WEPRMC	PRIMARY COLUMN VALUE
78	(4E)	UNSIGNED	1	WEALTR	ALTERNATE ROW VALUE
79	(4F)	UNSIGNED	1	WEALTC	ALTERNATE COLUMN VALUE
80	(50)	CHARACTER	8	WELMODE	LOGMODE NAME
88	(58)	UNSIGNED	1	WELANG	LANGUAGE FIELD
		1...		WEQRYRQD	QUERY REQUIRED TO DETERMINE TERMINAL LANGUAGE CHARACTERISTICS

Table 132. IKTWESTD control block (continued)

Offsets					
Dec	Hex	Type	Len	Name (Dim)	Description
		.111 1111		WELNGID	LANGUAGE INDICATOR
89	(59)	UNSIGNED	1	*	
		1...		WEDISP	DISPLAY CAPABLE FLAG
		.1..		WE3FALTP	SLU CAPABILITIES BIT
		..1.		WERECONN	CLSDST PASS FOR RECONNECT OF A DISCONNECTED TERMINAL
		...1		WE_CV64_PRESENT	CV64 IS PRESENT ON CINIT
	 1...		WE_SV6485_PRESENT	SV85 PRESENT ON CV64
	 1..		WE_DNS_NAME_TRUNC	DNS NAME TRUNCATION INDICATOR
	11		*	AVAILABLE
90	(5A)	CHARACTER	6	*	AVAILABLE
90	(5A)	CHARACTER	4	WEURC	URC DATA
90	(5A)	SIGNED	4	WEMSG	MESSAGE TO BE SENT IF A FAILURE IN IKTCAS31 OCCURS AND IKTCASOY GETS CONTROL. IF NOT FAILING, BYTES AVAILABLE FOR USE IN LATER PROCESSING.
96	(60)	CHARACTER	8	WETRMID	SYMBOLIC TERMINAL NAME (THIS COULD BE AN ALIAS)
104	(68)	CHARACTER	8	WETRMR	REAL TERMINAL NAME
112	(70)	CHARACTER	4	WE_IP_ADDR	IPv4 ADDRESS
116	(74)	UNSIGNED	2	WE_PORT_NUM	IP PORT NUMBER
118	(76)	UNSIGNED	2	WE_DNS_NAME_LEN	LENGTH OF DNS NAME
120	(78)	CHARACTER	131	WE_DNS_NAME	DNS NAME
251	(FB)	CHARACTER	1	*	FOR ALIGNMENT
252	(FC)	ADDRESS	4	WE_CV64_PTR	Address of CV64
256	(100)	UNSIGNED	1	WE_DISP_IPAD_LEN	Printable IP address length
257	(101)	CHARACTER	39	WE_DISP_IPAD	Printable IP address
296	(128)	CHARACTER	*	WEEND	Ending alignment

Offsets					
Dec	Hex	Type	Len	Name (Dim)	Description
0	(0)	STRUCTURE	177	WESTD_CV64_EXT	
0	(0)	CHARACTER	177	WESTD_CV64	Copy of the CV64 when it exists

TCP/IP information control vector

This is defined by the control vector X'64' (CV64) and contains the IP characteristics of a TN3270E LU. These characteristics include the IP address of the LU, the port name associated with the LU, and the name of the LU's domain name server, if any. The WE_IP_ADDR field has been deprecated because it can be used for a client represented by only an IPv4 address. WE_PORT_NUM, WE_DNS_NAME_LEN, and WE_DNS_NAME have also been deprecated. To support both IPv4 and IPv6 clients, the WE_CV64_PTR is a pointer to a CV64. Exits should change to process the CV64 information. See *SNA Formats* for information about how to process the CV64 and its subvectors.

IKTXSA

Table 133. IKTXSA control block

Offsets					
Dec	Hex	Type	Len	Name (Dim)	Description
0	(0)	STRUCTURE	48	IKTXSA	SVC 93 SVRB EXTENDED SAVE AREA
0	(0)	SIGNED	4	XSAPRMO	REGISTER 0 AT SVC ENTRY
0	(0)	SIGNED	2	XSAASID	ASID PASSED
		1... ..		XSANEDIT	TPUT NOEDIT =1 (PASS-THROUGH)
0	(0)	BITSTRING	1	*	ASID PASSED
2	(2)	SIGNED	2	XSAPRMSZ	BUFFER SIZE PASSED
4	(4)	SIGNED	4	XSAPRM1	REGISTER 1 AT SVC ENTRY
4	(4)	UNSIGNED	1	XSAOPTNS	USER OPTIONS
		1... ..		XSAPTGT	TPUT =0, TGET =1
		.1.. ..		XSAUSRID	USER ID IN REG 15 IF=1
		..1.		XSAPRIOR	HIGHP=0, LOWP=1
		...1		XSANOWT	WAIT =0, NOWAIT=1
	 1...		XSAHOLD	NOHOLD=0, HOLD =1
	1..		XSABRK	NOBREAK=0, BREAK = 1
	11		XSAEDITO	EDIT=00,ASIS=01,CNTL=10, FULLSCRN=11
5	(5)	ADDRESS	3	XSABFRAD	ADDRESS OF USER'S BUFFER
8	(8)	SIGNED	4	XSAWD3	
8	(8)	UNSIGNED	1	XSAFLAG	FLAG BYTE
		1... ..		XSAIDENQ	IKTASTPT HAS ENQ ED THE ASID
		.1.. ..		XSADMOVE	DATA MOVE INDICATOR
		..1.		XSATCBFX	ASYN EXIT INDICATOR. SET AT SVC ENTRY
		...1		XSADOSWT	SWITCH FOR 'DO' FALLOUT
		...1		XSATOKF	TPUT WITH TOKEN PASSED
	 1...		XSATATVF	'DATA VERIFICATION' INDICATOR
	1..		XSALOCKS	ROUTINE HAS LOCKS OBTAINED
	1.		XSAAUTH	ASID CALLER IS AUTHORIZED

Table 133. IKTXSA control block (continued)

Offsets					
Dec	Hex	Type	Len	Name (Dim)	Description
	1		XSARETY	FLAG USED BY IKTASTPT
9	(9)	ADDRESS	3	XSATCBA	CALLER'S TCB ADDRESS
12	(C)	SIGNED	4	XSAWD4	
12	(C)	SIGNED	4	XSASAVEA	USED FOR SAVING REG 14
16	(10)	CHARACTER	12	XSAENQAD	WORK AREA FOR ENQ/DEQ
16	(10)	SIGNED	4	XSAWD5	3 WORD LIST FORM OF ENQ AND
20	(14)	SIGNED	4	XSAWD6	DEQ. USED IN IKTASTPT
24	(18)	SIGNED	4	XSAWD7	AND IKT93EST
28	(1C)	SIGNED	4	XSAWD8	
28	(1C)	SIGNED	4	XSASAVEB	USED FOR SAVING REG 14
32	(20)	ADDRESS	4	XSAUSERP	PTR TO USER ID, IF PRESENT. SET AT SVC ENTRY(REG 15)
32	(20)	SIGNED	4	XSAWD9	
36	(24)	SIGNED	4	XSARETG	SAVE RETURN ADDRESS. SET AT SVC ENTRY.
36	(24)	SIGNED	4	XSAWD10	
40	(28)	SIGNED	4	XSAWD11	
40	(28)	CHARACTER	1	XSARC	SAVE AREA FOR RETURN CODE
41	(29)	CHARACTER	1	*	KEY OF SVC 93 CALLER
42	(2A)	SIGNED	2	XSAIDAS	CALLER'S ASID
44	(2C)	SIGNED	4	XSAWD12	
44	(2C)	ADDRESS	4	XSABUFAD	TPUT/TPG BUFFER ADDRESS

Chapter 4. Defining user modules and tables

This information describes the following topics:

- HSRTSIZE and OSRTSIZE start options
- Communication network management (CNM) routing table
- Logon-interpret routine requirements

Directory size of symbol resolution table for the host network

The symbol resolution table (SRT) consists of an internal VTAM table used primarily to find information about control blocks. For a VTAM started as gateway-capable and interconnected with other networks, the SRT is actually several tables: one for the host network and one for each network identified to VTAM. Each symbolic name has two parts. The first is a network ID (one assigned to each network) that finds an SRT directory. After the SRT directory is found, the second part of the symbolic name points to the address of a queue of SRT entries.

The HSRTSIZE constant specifies the number of queue pointers in the SRT directory for the network containing the VTAM host node. The IBM-supplied default value is 9973. You can change the number of SRT entry queues by specifying a number in the range 0–32767 (X'7FFF'). If you specify 0, VTAM uses the default value of 9973. Using a prime number of queue pointers results in a fairly even distribution of SRT entries to the queues. It is advisable that you choose a prime number.

Note: For networks with a large number of LUs, increasing the number shortens the length of the queues, thereby decreasing the logon time.

See the following instructions and information before changing the default directory size.

In order to estimate the number of entries in the host network's symbol resolution table (SRT), add the following:

- Network resources

Add one SRT entry for each network resource identified to VTAM. This includes applications, PUs, LUs, cross-domain resources, groups, and lines.

- Network ID

Add one SRT entry for each network identified to VTAM.

- Class of service table

- Add one SRT entry for each COS table associated with this host's network by COSTAB operands in active gateway NCPs. That is, add one to your count of SRT entries for every COSTAB operand specified on the BUILD and NETWORK macroinstruction with this host's network ID.

- Add one SRT entry for this host's COS table.

- Adjacent SSCP table

- Add one SRT entry for each destination SSCP in this host's network that is specified in an active adjacent SSCP table.

- Add one SRT entry for this network's SSCP list.

- Add one SRT entry for the default SSCP list.

- Alias name

Add one SRT entry for each alias LU name identified to the VTAM host.

- Autologon

Add one SRT entry for each SSCP that controls at least one LU that is designated as the PLU for a PLU-SLU automatic logon.

- Network address
 - Add one SRT entry for each network address assigned to nodes in this VTAM domain.
 - Add one SRT entry for the real network address of each cross-domain resource that establishes a session through this VTAM SSCP.
 - Add one SRT entry for the alias network address of each cross-network resource that establishes a session through this VTAM SSCP.
 - Add one SRT entry for each alternate gateway path to other-network SSCPs, if the SUBAREA operand is specified on the GWPATH statement.

- NetView trace requests

Add one SRT entry for each resource for which there might be pending NetView trace requests outstanding at any one time.

This SRT entry represents a NetView trace request that is pending for a resource that is not yet defined to VTAM. An example of this kind of undefined resource is a dynamically defined cross-domain resource for which no session is active at the time of the trace request. To determine how many entries might be needed for these pending traces, you should know how the NetView program is used in your installation, and be aware of the kinds of resources in your configuration that might be undefined to VTAM at the time of a NetView trace request.

Note: This value should represent the maximum number of resources for which there might be pending NetView trace requests outstanding at any one time, not the total number of resources for which a NetView trace might be requested.

- PCID

Add one SRT entry for each LU-LU session.

- Physical unit services control block

For each active connection to a physical unit channel-attached to this VTAM host:

- Add one SRT entry for the link.
- Add one SRT entry for the station.

In addition, if the PU is a subarea node (for example, a communication controller or a channel-to-channel attachment to a host):

Add one SRT entry for that subarea node.

Divide the number of entries in the host network's symbol resolution table by two times the size of the symbol resolution table (HSRTSIZE) to obtain an average queue depth. Performance increases as the SRT queue depth decreases. Therefore the larger the size of the SRT table, the better the performance.

Making the table larger can improve performance in two ways:

- The search is shorter and saves on CPU cycles.
- The page each element references is easily accessed, so there is less paging with the larger table size.

Directory size of symbol resolution tables for other networks

This constant specifies the number of queue pointers in every SRT directory for networks other than this VTAM network (that is, all networks with a network ID other than the network ID of this VTAM specified on the BUILD and NETWORK macroinstructions that are included in the NCP definition program). The IBM-supplied default value is 43. You can change the number of SRT entry queues in each directory by specifying a number in the range 0–32767 (X'7FFF'). If you specify 0, VTAM uses the default value of 43. Using a prime number of queue pointers results in a fairly even distribution of SRT entries to the queues. It is advisable that you choose a prime number.

Note: For networks with a large number of LUs, increasing this number shortens the length of the queues, thereby decreasing the logon time.

See the following instructions and information before changing the default directory size. To estimate the number of entries in the other network's symbol resolution table (SRT), add the following:

- Class-of-service table
 - Add one SRT entry for each COS table associated with another network by COSTA operands in active gateway NCPs. That is, add one to your count of SRT entries for every COSTAB operand specified on the BUILD and NETWORK macroinstructions with network IDs other than that of this host.
- Adjacent SSCP table
 - Add one SRT entry for each destination SSCP in another network that is specified in an active adjacent SSCP table.
 - Add one SRT entry for each default adjacent SSCP list for the other networks.
- Network address
 - Add one SRT entry for the real network address of each cross-network resource that establishes a session through this VTAM SSCP.
 - Add one SRT entry for the alias network address of each host-network resource that establishes a session through the VTAM SSCP.
 - Add one SRT entry for each alternate gateway path to other-network SSCPs, if the ADJNETSA operand is specified on the GWPATH statement.
- SSCPID
 - Add one SRT entry for each other-network SSCP that has a session with this VTAM SSCP.
- PCID
 - Add one SRT entry for each LU-LU session.
- Nodes
 - Add one SRT entry for each active CDRSC in another network.

Divide the number of entries in the other network's symbol resolution table by two times the size of the other network symbol resolution table, OSRTSIZE, to obtain an average SRT queue depth. Performance increases as the SRT queue depth decreases. Therefore the larger the size of the SRT table, the better the performance.

Making the table larger can improve in two ways:

- The search is shorter and saves on CPU cycles.
- The page each element references is easily accessed, so there is less paging with the larger table size.

CNM routing table

VTAM uses a communication network management (CNM) routing table to determine which CNM application program is to receive an unsolicited network-services request unit that requires further processing. An application program can embed its own procedure-related identifier (PRID) in each request sent to VTAM. When a reply to the request is returned, VTAM uses the PRID to route the reply to the application program. Unsolicited RUs contain network information but do not contain a PRID. VTAM uses the CNM routing table to determine which application program is to receive the unsolicited RU.

IBM supplies default routing information for the following CNM application programs:

- The NetView program
- Alias name translation facility (supplied by IBM as part of the NetView program)
- Downstream load utility

The IBM-supplied CNM routing table is named ISTMGC01. See [Appendix A, “IBM-supplied CNM routing table,” on page 219](#) for a listing of the IBM-supplied CNM routing table.

For any application other than the default CNMTABLE to route unsolicited RUs to the application program, modify the default and give it a new name. You can name your supplemental table any 1- through

8-character name, but do not use ISTMGC00 as a table name if you are using multiple CNM routing tables because VTAM will always look for ISTMGC001 first and use that when VTAM is starting with a particular set of libraries. You might choose instead to select a 1- through 8-character name and specify the CNMTAB= start option. This allows you to have different CNM tables stored in the same set of VTAM libraries, which results in the following benefits:

- You can start VTAM with different CNM tables.
- You can run VTAM with different characteristics at different times.
- You can run two VTAMs that are different at the same time.
- You will have an old copy and a backup copy.

If you are starting multiple VTAMs that use the same VTAM library, you can have multiple supplemental CNM routing tables. Use the CNMTAB start option to associate a specific CNM routing table with the VTAM being started.

Whenever VTAM receives an unsolicited RU, VTAM uses the value specified on the CNMTAB start option to route the request.

- If you do not specify the CNMTAB start option or specify CNMTAB=ISTMGC00, VTAM will use ISTMGC00, if available. No message is issued if ISTMGC00 is not found. In this case, VTAM uses the IBM-supplied table, ISTMGC01.
- If you specify a name other than ISTMGC00 on the CNMTAB start option and it is not found by VTAM, message IST116I is issued.

Any user-supplied table used by VTAM overrides the IBM supplied table, ISTMGC01.

Installing the CNM routing table

You must link-edit before starting VTAM. The name of the replacement module must be the same as the IBM-supplied module it replaces.

Procedure

Follow these steps to install a CNM routing table:

1. Assemble the module.
2. Link-edit it to the appropriate VTAM library: SYS1.VTAMLIB.

Structure of the CNM routing table

A CNM routing table consists of a 12-byte header entry and routing table entries. The 12-byte header entry contains the size and number of routing table entries that follow it. Each routing table entry contains the network services RU type to be routed, followed by the application program name to which the network services RU is to be routed. The header format is described in [Table 134 on page 214](#).

Table 134. Format of CNM routing table header

Dec (Hex) offset	Size (Bytes)	Description
0(0)	2	Number of entries
2(2)	2	Entry length (X'000C')
4(4)	8	Reserved

A CNM routing table entry consists of 12 bytes as described in [Table 135 on page 215](#).

Table 135. Format of CNM routing table

Dec (Hex) offset	Size (Bytes)	Description
0(0)	1	Flag byte: Bit 0 0 = Do not send to VTAM operator 1 = Send to VTAM operator also Bit 1 0 = Embed in DELIVER RU 1 = Do not embed in DELIVER RU Bits 2–7 Reserved
1(1)	3	Network services RU type
4(4)	8	Application program name in EBCDIC

Bit 0 of the flag byte specifies whether the RU is to be sent to the VTAM operator designated to receive unsolicited messages as well as to the CNM application program named in the table entry. This flag bit is supported only for requests for which VTAM has operator message support (that is, ROUTE-INOP).

Bit 1 of the flag byte allows you to specify that an RU is to be sent to a user-written CNM application program without being embedded in a DELIVER RU. If this flag bit is set to 1, VTAM sends the request unit to the application program without embedding it in a DELIVER RU. If the flag bit is set to 0, the request unit is embedded in the DELIVER RU.

The following types of requests can be received by an application program that uses the CNM interface. You should provide an entry in the table for each type of unsolicited network services RU. The network services header value listed below must be coded in the respective entry for that network services RU type (bytes 1–3).

Request	Header value
RECMS	X'010381'
RECFMS	X'410384'
INIT-LOAD	X'3F0233'
TR-INQ	X'3F0814'
ROUTE-INOP	X'410289'
ER-TESTED	X'410386'
CNM	X'810814'
NMVT	X'41038D'

You can code more than one entry associating a single type of RU with more than one application program; however, no more than one program associated with that type of RU can be active at the same time. For example, if an application program associated with the RECMS RU is already active, another application program associated with the RECMS RU is unable to open its ACB.

For a CNM routing table that permits either of two CNM application programs (CNMAPPL1 or CNMALT) to receive unsolicited requests, you might code the following:

HDR	CSECT DC	X'0004'	NUMBER OF ENTRIES
-----	-------------	---------	-------------------

	DC	X'000C'	ENTRY LENGTH
	DC	8X'00'	RESERVED
ENT1	DC	X'00'	RESERVED
	DC	X'010381'	RECMS
	DC	CL8'CNMAPPL1'	APPLNAME
ENT2	DC	X'00'	RESERVED
	DC	X'410384'	RECFMS
	DC	CL8'CNMAPPL1'	APPLNAME
ENT3	DC	X'00'	RESERVED
	DC	X'010381'	RECMS
	DC	CL8'CNMALT'	APPLNAME
ENT4	DC	X'00'	RESERVED
	DC	X'410384'	RECFMS
	DC	CL8'CNMALT'	APPLNAME
	END		

Note that in the above example, CNMAPPL1 and CNMALT have each been defined to receive the same type of unsolicited requests (RECMS and RECFMS). In this case, both application programs cannot be active (that is, have open ACBs for application programs whose network names are CNMALT and CNMAPPL1) at the same time.

Notes:

- If you are using a user-written alias name translation facility, include it in your CNM routing table as the receiver of the TR-INQ and ROUTE-INOP request units. If you are using the alias name translation facility supplied as a function of the NetView program, the default CNM routing table already contains the information needed for routing.
- If you add an entry for ER-TESTED to ISTMG00 (or whatever you have chosen to name it) and specify an application other than ISTNOP as the receiver of that request unit, D NET,ROUTE,TEST=YES will not display the expected response messages. This is because the ROUTE TEST=YES responses are returned as ER-TESTED RUs and routed using the CNM routing tables.

Logon-interpret routine requirements

When a session-establishment request is received, VTAM uses the interpret table to determine which application program is to be notified. This topic discusses the logon-interpret routine requirements.

If you are not using network-qualified names, you do not need to change your logon-interpret routine; continue to use the interface you have been using.

Entry from:

VTAM to entry point *routinename*

Initial register contents

The initial contents of register 4 point to a parameter list that your routine can fill with the network identifier and resource name.

Register 0:

Length of logon message (any length from 1 to 80 bytes)

Register 1:

Address of first byte of logon message (see note in [“Operation” on page 218](#))

Register 2

Address of an 8-byte LU name

Register 4:

Address of parameter list for network identifier and resource name

Offset

Description

0–27

Information about fixed or interpreted name

28

Uninterpreted name

Register 13:

Address of a 72-byte save area provided by VTAM

Register 14:

Return address

Register 15:

Address of entry point of this routine

Final register contents

Registers 0 and 1 contain the name of the application program (in EBCDIC characters) with which the LU is to establish a session:

Register 0:

First 4 characters of name (left-adjusted)

Register 1:

Last 4 characters of name (left-adjusted)

Registers 2–14:

Restored to entry contents

Register 15:

Return code:

0:

Application program was found and the name was placed in registers 0 and 1.

Nonzero:

Application program was not found and the name was not placed in registers 0 and 1.

If the name of the application program contains fewer than 8 characters, use blanks to provide a name with 8 characters.

Logon-interpret routine parameter list

When the exit gets control, the address of the following parameter list is in register 4. Offsets 0 through 27 include information about the fixed or interpreted name. Offset 28 includes the uninterpreted name.

Table 136. Logon-interpret routine parameter list

Dec offset	Size (Bytes)	Description	Input or output
0	2	Length of parameter list	Input
2	8	Name of requesting LU	Input
10	17	Interpreted name (In the form of either name or netid.name)	Output
27	1	Length of uninterpreted name	Input
28	n	Uninterpreted name	Input

Operation

The logon-interpret routine is run synchronously in pageable storage under the control of VTAM, not under the control of an application program. For the application program to receive the logon, this routine must validate the logon, obtain the symbolic name of the application program to receive control, and provide this name to VTAM. Otherwise, the routine specifies that the logon is not valid or that the name of the application program was not found.

Because the logon-interpret routine operates at VTAM's main task dispatching priority, there is a possibility of lockout if a wait requires another task action. The routine gets control in supervisor state with a VTAM storage key, so errors within the routine could cause damage to VTAM or to system control blocks and modules.

The logon-interpret routine must also:

- Save and restore the contents of registers 2–14 when receiving and passing control.
- Use reenterable code (the routine must not store anything within itself or modify itself while it has control).
- Perform no I/O operations; an I/O request causes the routine to end abnormally.

Notes:

- The logon message that is passed to the interpret routine is read-only, and cannot be modified.
- For LOGON requests, VTAM again searches the interpret table—after USS translation—looking just for the specified APPLID. After USS translation, register 1 contains the address of the first byte of the APPLID.

Appendix A. IBM-supplied CNM routing table

Following is the CNM routing table for IBM-supplied CNM applications. Each entry in the table represents an application.

ISTMGC01	CSECT		
	DS	0F	
	DC	S(ENTRIES)	NUMBER OF ENTRIES
	DC	X'000C'	LENGTH OF EACH ENTRY
	DC	XL4'00000000'	RESERVED
	DC	XL4'00000000'	RESERVED
ENTRIES	EQU	22	NUMBER OF ENTRIES
*			
	DC	XL1'00'	FLAG BYTE
	DC	XL3'410386'	ER_TESTED RU
	DC	CL8'ISTNOP '	NETWORK OPERATOR SERVICES NETWORK NAME
*			
	DC	XL1'00'	FLAG BYTE
	DC	XL3'410289'	ROUTE_INOP RU
	DC	CL8'ISTNOP '	NETWORK OPERATOR SERVICES NETWORK NAME
*			
	DC	XL1'00'	FLAG BYTE
	DC	XL3'010381'	RECMS RU
	DC	CL8'BNHDSERV'	NPDA ACB NETWORK NAME
*			
	DC	XL1'00'	FLAG BYTE
	DC	XL3'010381'	RECMS RU
	DC	CL8'BNJDSERV'	NPDA VERSION 2 ACB NETWORK NAME
*			
	DC	XL1'00'	FLAG BYTE
	DC	XL3'410384'	RECFMS RU
	DC	CL8'BNHDSERV'	NPDA ACB NETWORK NAME
*			
	DC	XL1'00'	FLAG BYTE
	DC	XL3'410384'	RECFMS RU
	DC	CL8'BNJDSERV'	NPDA VERSION 2 ACB NETWORK NAME
*			
	DC	XL1'00'	FLAG BYTE
	DC	XL3'3F0233'	INIT LOAD RU
	DC	CL8'DLUPULP '	DOWNSTREAM LOAD UTILITY ACB NETWORK NAME
*			
	DC	XL1'00'	FLAG BYTE
	DC	XL3'410384'	RECFMS RU
	DC	CL8'AAUTSKLP'	NLDM ACB NETWORK NAME WITHOUT NPDA
*			
	DC	XL1'40'	FLAG - DO NOT IMBED IN DELIVER RU
	DC	XL3'3F0814'	TRANSLATE INQUIRE RU
	DC	CL8'ALIASAPL'	ALIAS ACB NETWORK NAME
*			
*		NLDM R2 UNIQUE SUPPORT	
*			
	DC	XL1'40'	FLAG - DO NOT IMBED IN DELIVER RU
	DC	XL3'810814'	CNM AMRU
	DC	CL8'BNHDSERV'	NPDA ACB NETWORK NAME
*			
	DC	XL1'40'	FLAG - DO NOT IMBED IN DELIVER RU
	DC	XL3'810814'	CNM AMRU
	DC	CL8'BNJDSERV'	NPDA VERSION 2 ACB NETWORK NAME
*			
	DC	XL1'40'	FLAG - DO NOT IMBED IN DELIVER RU
	DC	XL3'810814'	CNM AMRU
	DC	CL8'AAUTSKLP'	NLDM ACB NETWORK NAME WITHOUT NPDA
*			
	DC	XL1'80'	FLAG - ALSO SEND TO NETWORK OPERATOR
	DC	XL3'410289'	ROUTE_INOP RU
	DC	CL8'BNHDSERV'	NPDA ACB NETWORK NAME
*			
	DC	XL1'80'	FLAG - ALSO SEND TO NETWORK OPERATOR
	DC	XL3'410289'	ROUTE_INOP RU
	DC	CL8'BNJDSERV'	NPDA VERSION 2 ACB NETWORK NAME
*			
	DC	XL1'80'	FLAG - ALSO SEND TO NETWORK OPERATOR
	DC	XL3'410289'	ROUTE_INOP RU
	DC	CL8'AAUTSKLP'	NLDM ACB NETWORK NAME WITHOUT NPDA
*			

```

*          NLDM R3 UNIQUE SUPPORT
*
DC      XL1'00'          FLAG BYTE
DC      XL3'41038D'      NMVT RU
DC      CL8'BNJDSERV'    NPDA VERSION 2 ACB NETWORK NAME
*
DC      XL1'00'          FLAG BYTE
DC      XL3'41038D'      NMVT RU
DC      CL8'AAUTSKLP'    NLDM ACB NETWORK NAME WITHOUT NPDA
*
DC      XL1'00'          FLAG BYTE
DC      XL3'010381'      RECMS RU
DC      CL8'DSICRTR '    NETVIEW ROUTER
*
DC      XL1'00'          FLAG BYTE
DC      XL3'410384'      RECFMS RU
DC      CL8'DSICRTR '    NETVIEW ROUTER
*
DC      XL1'40'          FLAG - DO NOT IMBED IN DLV RU
DC      XL3'810814'      CNM RU
DC      CL8'DSICRTR '    NETVIEW ROUTER
*
DC      XL1'80'          FLAG - ALSO SEND TO NOS
DC      XL3'410289'      ROUTE INOP RU
DC      CL8'DSICRTR '    NETVIEW ROUTER
*
DC      XL1'00'          FLAG BYTE
DC      XL3'41038D'      NMVT RU
DC      CL8'DSICRTR '    NETVIEW ROUTER
*
*          END OF REFERENCED TABLE ENTRIES
*
DC      XL1'00'          UNUSED TABLE ENTRY
DC      XL3'000000'      IF REFERENCED THEN ALSO UPDATE
DC      CL8'              NUMBER OF ENTRIES COUNT
END      ISTMGC01

```

Appendix B. VTAM session flows

This topic contains flow information for writing VTAM installation-wide exit routines.

[Figure 4 on page 222](#) and [Figure 5 on page 223](#) depict subarea flows. These session flows are related primarily to the information found throughout [Chapter 1, “Writing VTAM installation-wide exit routines,”](#) on [page 1](#). [Figure 6 on page 224](#) through [Figure 13 on page 231](#) feature alias selection.

[Figure 16 on page 235](#) depicts VR selection for boundary function LUs.

[Figure 17 on page 236](#) through [Figure 21 on page 240](#) depict APPN flows. These session flows are related primarily to APPN-related information found in [Chapter 1, “Writing VTAM installation-wide exit routines,”](#) on [page 1](#).

Session flows for subarea

[Figure 4 on page 222](#) through [Figure 13 on page 231](#) depict existing flows in a subarea environment and remain unchanged with the addition of APPN to your network.

Cross-network session for CDINIT

[Figure 4 on page 222](#) depicts a sample cross-network session between APPLA and APPLB.

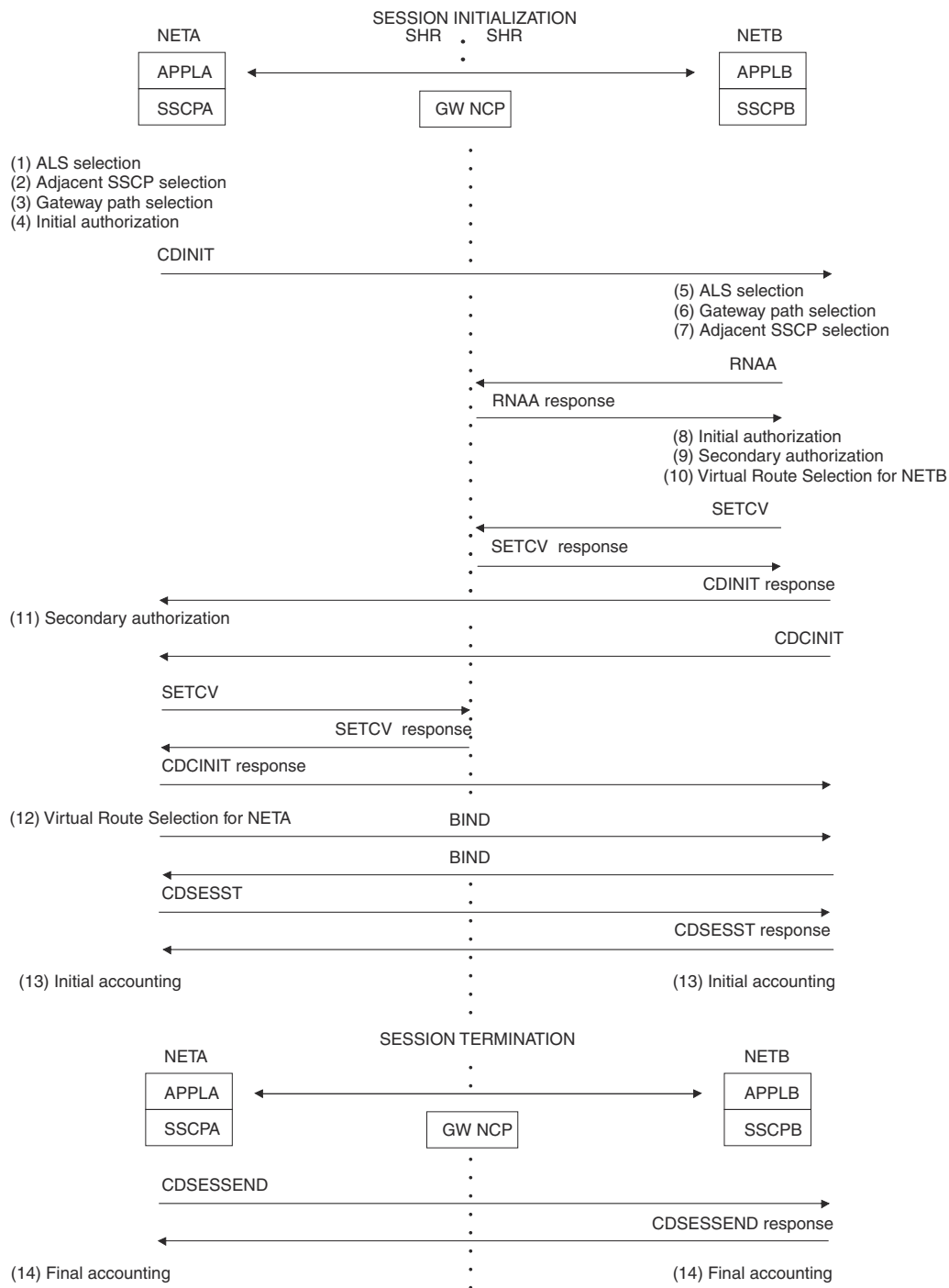


Figure 4. Sample cross-network session for CDINIT

Cross-network for INIT_OTHER_CD (third-party initiated)

Figure 5 on page 223 depicts a sample cross-network session between APPLA and APPLB for INIT_OTHER_CD.

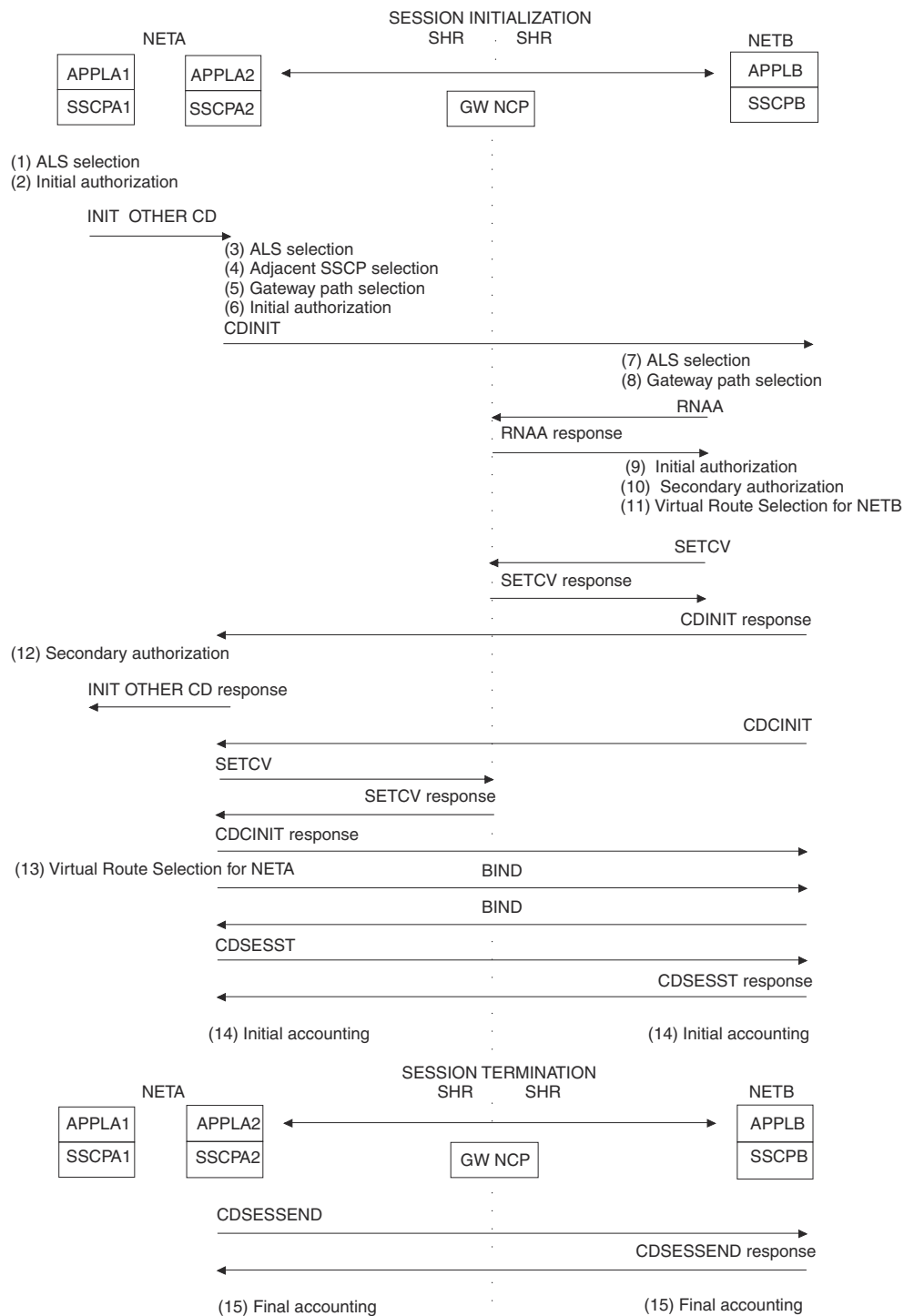


Figure 5. Sample cross-network session for INIT_OTHER_CD (third-party initiated)

Session flows for alias selection

The flows depicted in [Figure 6 on page 224](#) through [Figure 13 on page 231](#) indicate when the alias selection function is invoked and for what reason. The following information applies to the flows:

- Unless otherwise noted, the flows proceed as though the alias selection function does not return any translated names and that no host is running the NetView alias application program.
- If the exit returns a translated name, subsequent hosts do not request translation for that name.

- If the DLU real network ID and real name are already known, either through prior translation or a predefined real CDRSC, VTAM does not pass the DLU name to the exit for translation in a subsequent session.
- Associated LU names do not appear if the SLU is not using an associated LU table.
- All flows proceed as though the gateway control is SHARE/SHARE. If the gateway control is ONLY, alias invocations and RU flows vary slightly.

Tip: In Figure 6 on page 224 through Figure 13 on page 231, the session flows are numbered. See the respective numbered items following the session flows for an explanation of the flows.

SLU-initiated session for alias selection

Figure 6 on page 224 depicts SLU-initiated session flows for the alias selection function; it indicates when the alias selection function is invoked and for what reason. The numbers correspond to the descriptions after the figure.

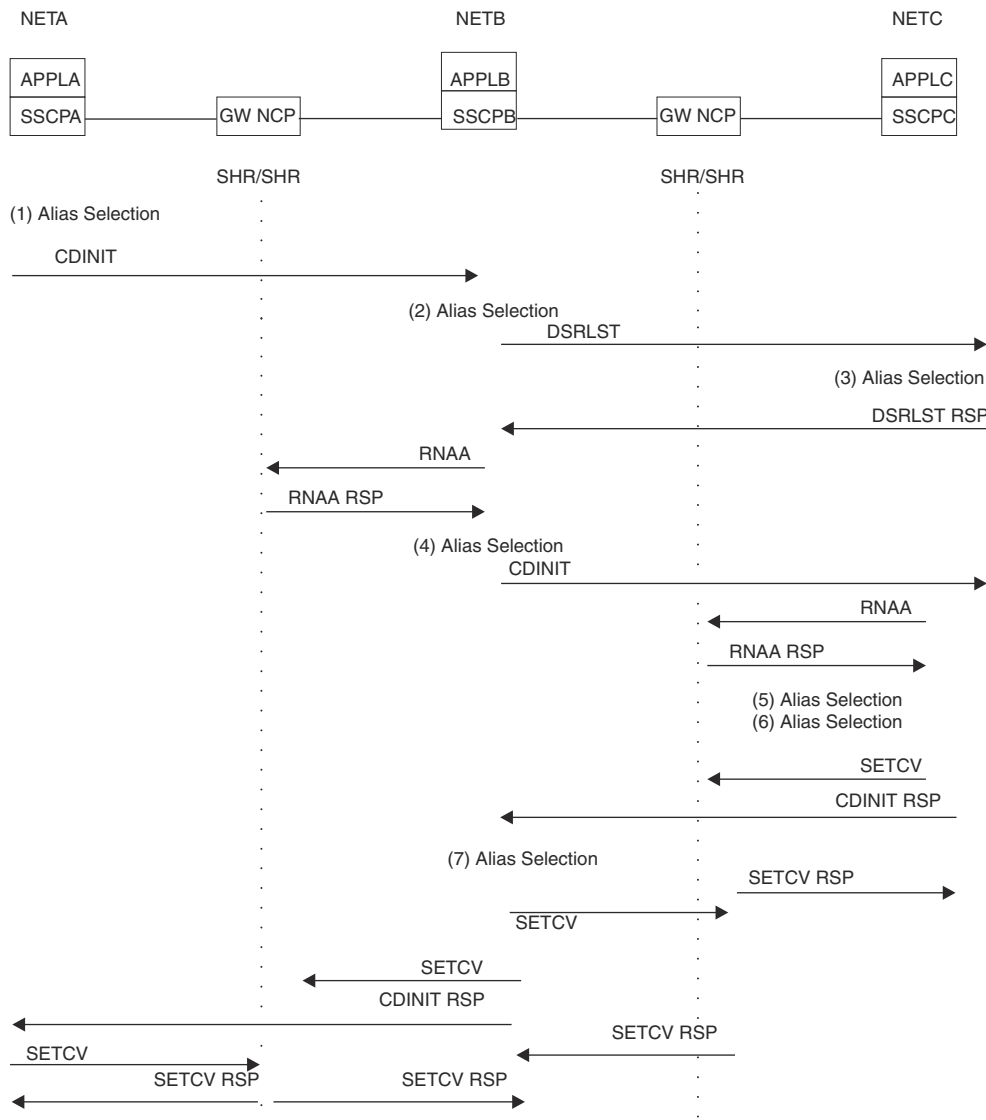


Figure 6. SLU-initiated session for alias selection

1. VTAM passes the DLU alias name, the OLU real name, the associated LU real names, and the logon mode and COS names as known in the OLU's network. The alias selection function has the option of returning the DLU real name, the network ID of the DLU real name, the owning SSCP of the DLU, the OLU and associated LU alias names as known in the DLU network, and the logon mode and COS names as known in the DLU network.

2. Same description as number 1.
3. VTAM passes the DLU alias name. The alias selection function has the option of returning the DLU real name, the network ID of the DLU real name, and the owning SSCP of the DLU.
4. VTAM requests translations from NETA to NETC for the OLU alias name and the logon mode name. VTAM also requests the name of the SSCP that owns the DLU.
5. This invocation is identical to number 3 except that no request is made for the owning SSCP.
6. VTAM drives the session management exit routine to translate the COS name from NETC to NETB.
7. Same description as number 6.

PLU-initiated session for alias selection

Figure 7 on page 226 depicts PLU-initiated session flows for the alias selection function; it indicates when the alias selection function is invoked and for what reason. The numbers correspond to the descriptions after the figure.

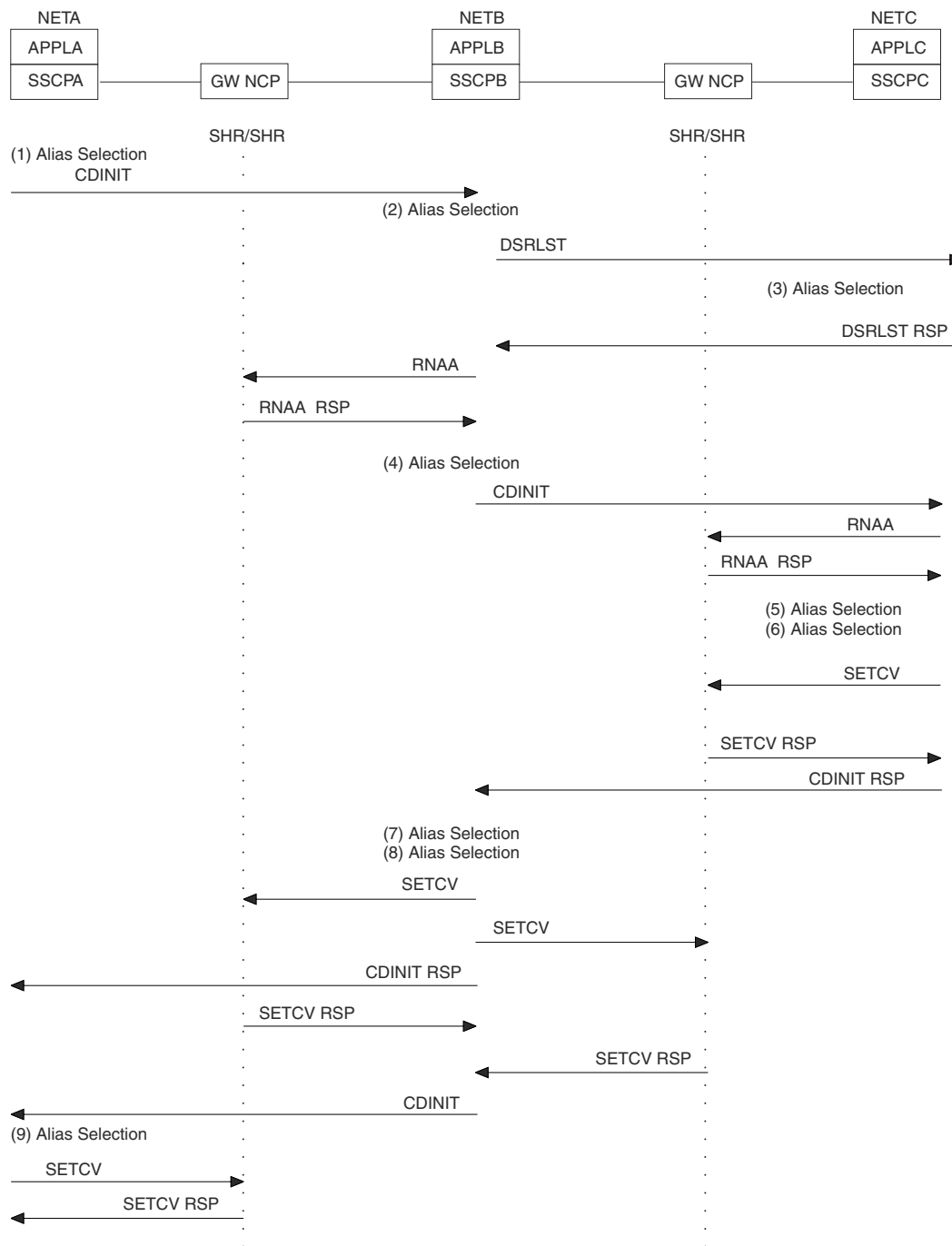


Figure 7. PLU-initiated session for alias selection

1. VTAM passes the DLU alias name, the OLU real name, and the logon mode name as known in the OLU's network. The alias selection function has the option of returning the DLU real name, the network ID of the DLU real name, the name of the SSCP that owns the DLU, the OLU alias name, and the logon mode name as known in the DLU network.
2. Same description as number 1.
3. VTAM passes the DLU alias name. The alias selection function has the option of returning the DLU real name, the network ID of the DLU real name, and the owning SSCP of the DLU.
4. VTAM requests translations from NETA to NETC for the OLU alias name and the logon mode name. VTAM also requests the name of the SSCP that owns the DLU.
5. Identical to number 3 except no request is made for the owning SSCP.

6. VTAM drives the session management exit routine to translate the COS name from NETC to NETB, and to translate the associated LU names from NETC to NETA.
7. Same description as number 6.
8. VTAM drives the exit routine to translate the COS name from NETB to NETA. No attempt is made to translate the associated LU names because this host has already attempted to translate them.
9. VTAM drives the exit routine to translate the COS name from NETB to NETA, and to translate the associated LU names from NETC to NETA.

Session flows for HPR

Figure 8 on page 227 depicts flows for an HPR route across a VR-based TG.

Figure 8. HPR across a VR-based TG

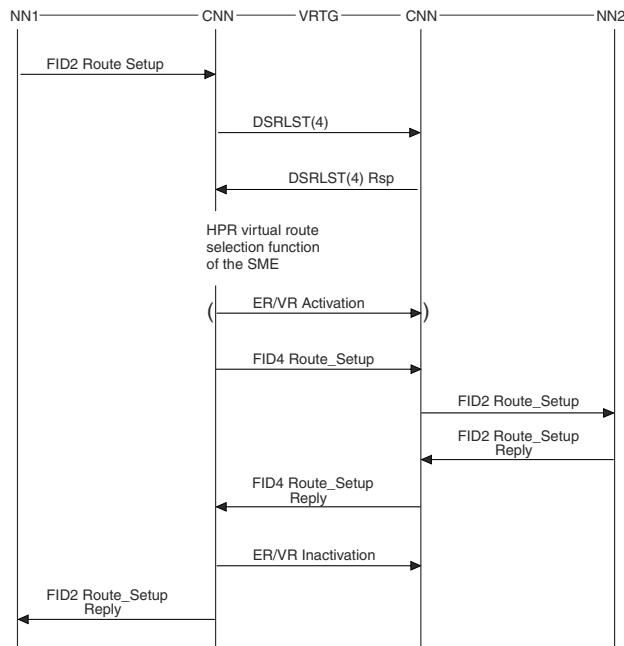


Figure 9. HPR across a VR-based TG

ILU-initiated session flows (third-party initiated)

Figure 10 on page 228 depicts session flows for an ILU-initiated session. LUC is in session with APPLA, which issues CLSDST PASS to pass the SLU(LUC) to a new PLU(APPLB). The numbers correspond to the descriptions after the figure.

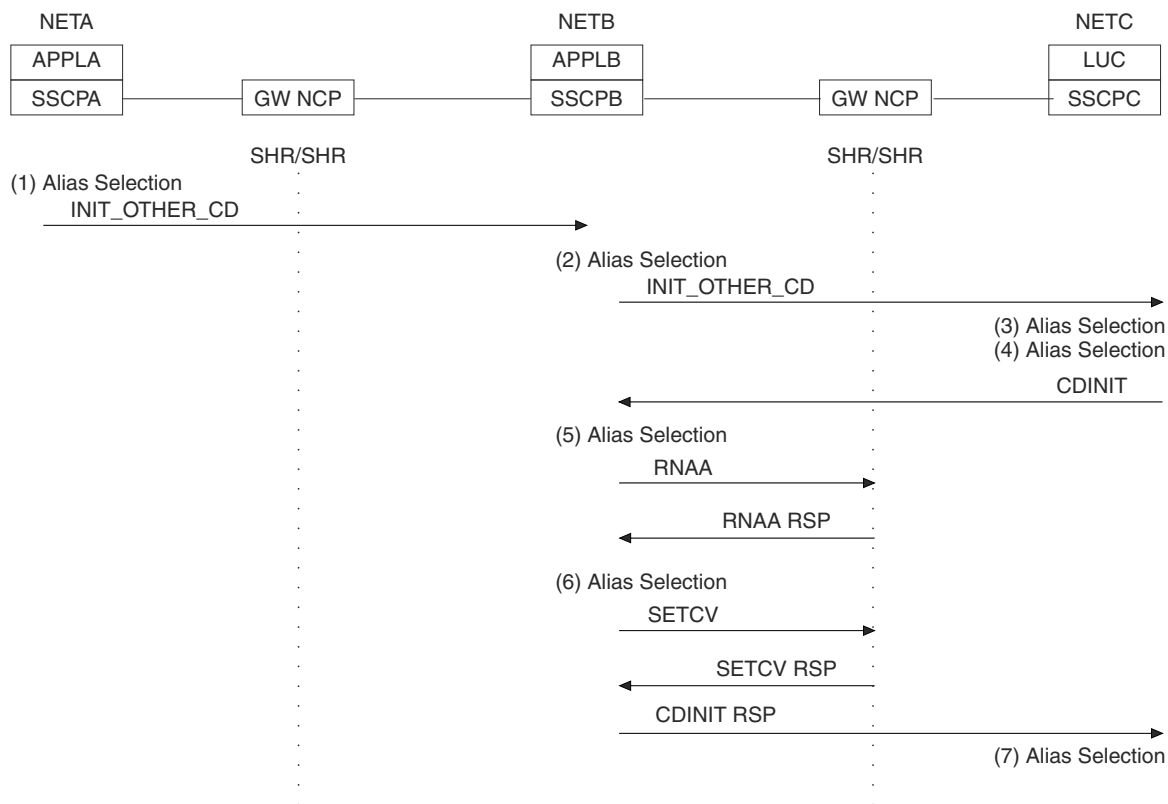


Figure 10. ILU-initiated session flows (third-party initiated)

1. VTAM passes the PLU alias name as known in NETA. The alias selection function has the option of returning the real name of the PLU, the network ID of the PLU, and the name of the SSCP that owns the PLU. The exit routine can also return the logon mode name as known in the SLU's real network.
2. The exit returned the logon mode name.
3. Same description as number 1.
4. VTAM passes the DLU alias name, the OLU real name, the associated LU real names, and the logon mode and COS names as known in NETC. The alias function has the option of returning the DLU real name, the network ID of the DLU real name, the name of the SSCP that owns the DLU, the OLU and associated LU alias names as known in the DLU network, and the logon mode and COS names as known in the DLU network.
5. Same description as number 4.
6. SSCPB assumes the DLU network ID is NETB and finds that it owns the PLU. VTAM requests translations from NETC to NETB for the OLU alias name, the associated LU alias names, the logon mode name, and the COS name.
7. VTAM drives the session management exit routine to translate the OLU alias name, the associated LU alias names, the logon mode name, and the COS name from NETC to NETB.

PLU real name, NETID returned (third-party) ILU-initiated session flows with PLU real name and NETID

Figure 11 on page 229 depicts session flows for an ILU-initiated session with PLU real name and NETID returned. The numbers correspond to the descriptions after the figure.

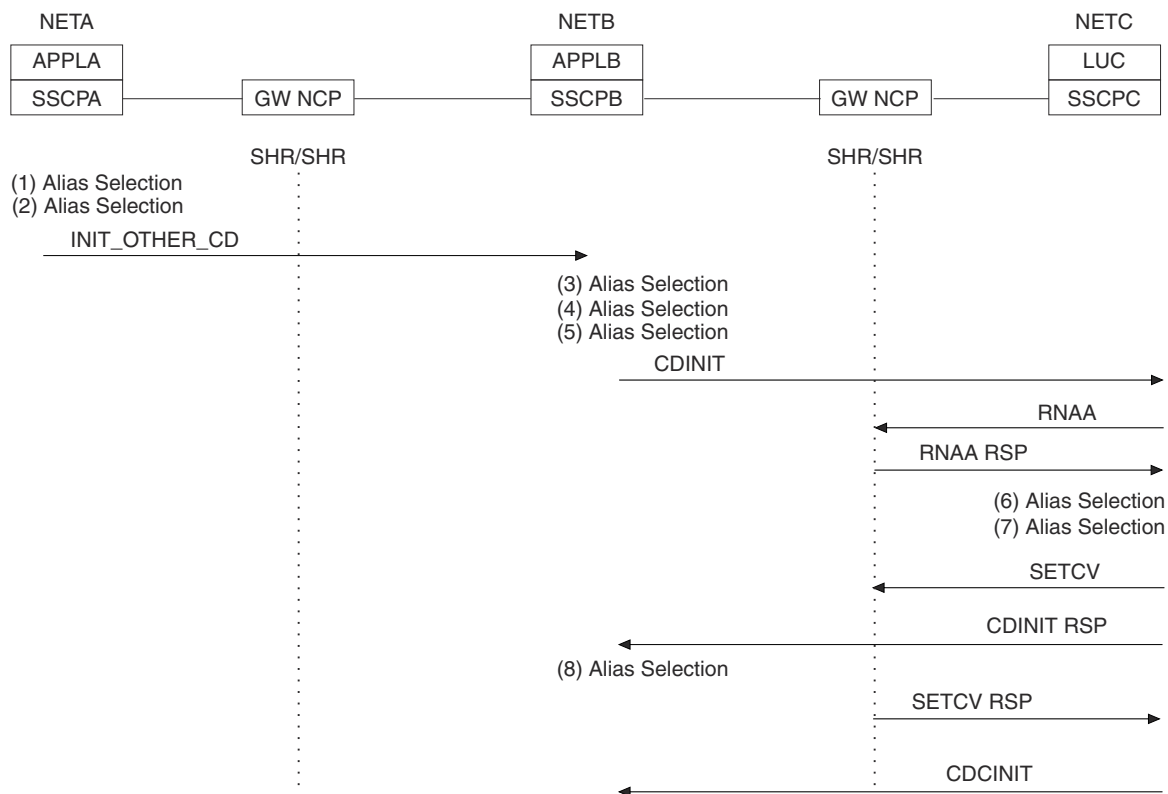


Figure 11. ILU-initiated session flows with PLU real name and NETID returned (third-party initiated)

1. VTAM passes the PLU alias name as known in NETA. The alias selection function has the option of returning the real name of the PLU, the network ID of the PLU, and the name of the SSCP that owns the PLU. The exit can also return the logon mode name as known in the SLU's network. In this case, assume the alias selection function has returned the PLU real name and network ID.
2. Because VTAM has learned the PLU real name and already knows the SLU real name, VTAM drives the session management exit routine again, asking for the PLU and SLU alias names.
3. VTAM drives the session management exit routine for the logon mode name as known in the SLU's network.
4. Same description as number 2.
5. VTAM drives the exit routine to translate the OLU alias and the logon mode name from NETB to NETC.
6. Same description as number 5.
7. VTAM drives the exit routine to translate the COS name and associated LU names from NETC to NETA.
8. Same description as number 7.

SLU-initiated flows with USERVAR name

Figure 12 on page 230 depicts SLU-initiated session flows (with USERVAR name) for the alias selection function; it indicates when the alias selection function is invoked and for what reason. The numbers correspond to the descriptions after the figure.

APPLA initiates a session with an application program named UVNAME. SSCPC recognizes that UVNAME is a USERVAR for the application program named APPLC.

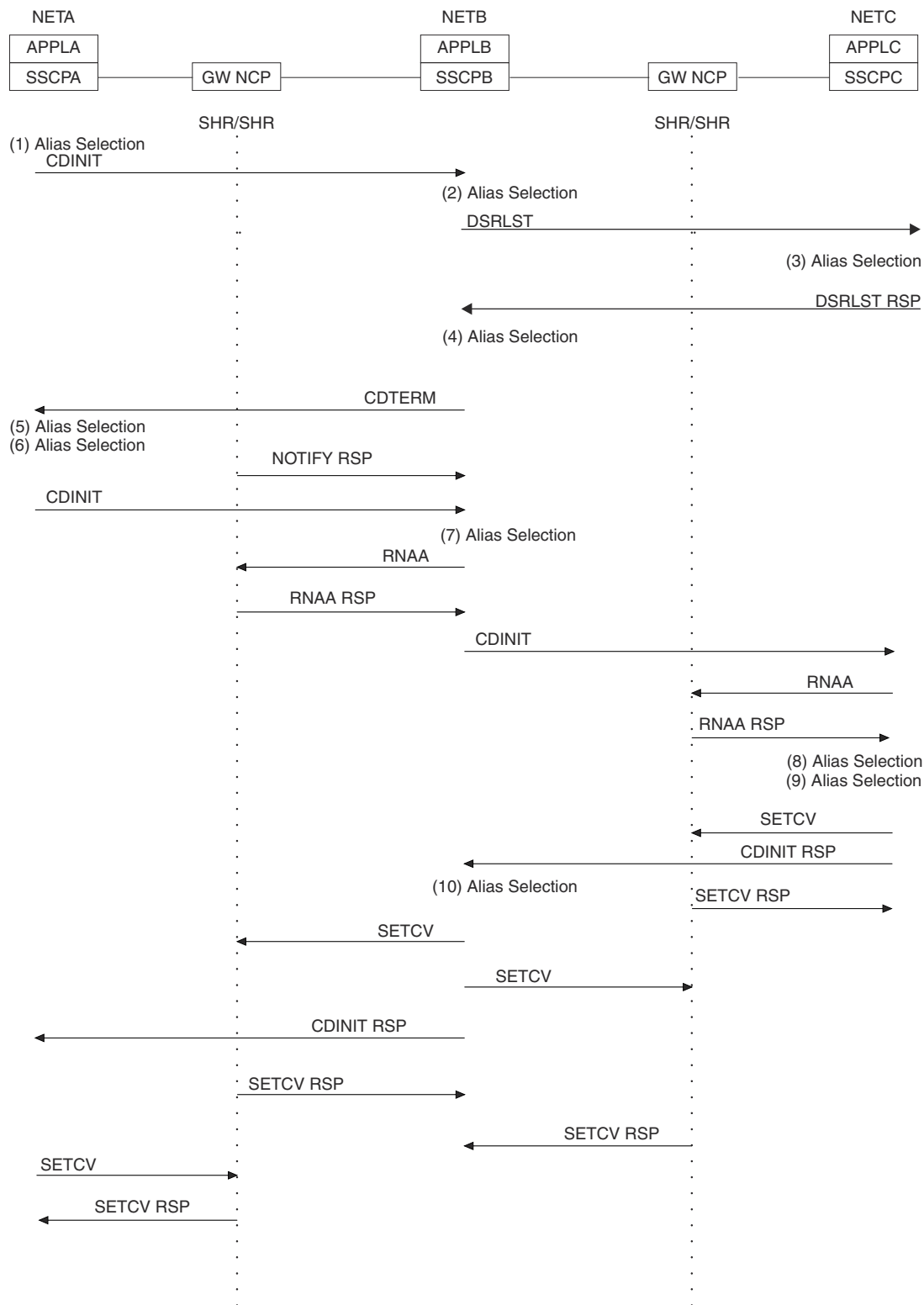


Figure 12. SLU-initiated flows with USERVAR name

1. VTAM passes the DLU alias name, the OLU real name, the logon mode name, the COS name, and associated LU names as known in the OLU's network. The alias selection function has the option of returning the DLU real name, the network ID of the DLU real name, the owning SSCP of the DLU, the OLU and associated LU alias names as known in the DLU network, and the logon mode and COS names as known in the DLU network.
2. Same description as number 1.

3. SSCPC recognizes that UVNAME is a USERVAR for APPLC, returns the real name on the DSRLST RSP, and drives the session management exit routine to translate the DLU real name to the DLU alias name.
4. VTAM drives the exit routine to translate the DLU real name to the DLU alias name.
5. Same description as number 4.
6. SSCPA restarts the session with the new application program name and drives the session management exit routine to translate the OLU alias name, associated LU alias names, the logon mode name, and the COS name from NETA to NETC.
7. VTAM drives the exit routine to translate the OLU alias name, the associated LU alias names, the logon mode name, and the COS name from NETA to NETC.
8. Same description as number 7.
9. VTAM drives the exit routine to translate the COS name from NETC to NETB.
10. Same description as number 9.

DSRLST session flows

Figure 13 on page 231 depicts DSRLST session flows for the alias selection function; it indicates when the alias selection function is invoked and for what reason. The numbers correspond to the descriptions after the figure.

APPLA has issued an INQUIRE OPTCD=APPSTAT for an application program named UVNAME. UVNAME is a USERVAR known in NETC as APPLC.

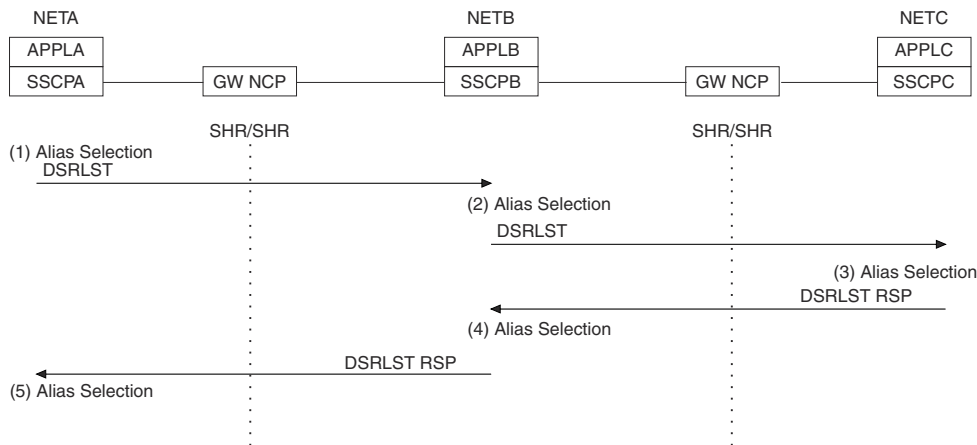


Figure 13. Session flows for DSRLST

1. VTAM passes the DLU alias name. The alias selection function has the option of returning the DLU real name, the network ID of the DLU real name, and the owning SSCP of the DLU.
2. Same description as number 1.
3. SSCPC recognizes that UVNAME is a USERVAR for APPLC, returns the real name on the DSRLST RSP, and drives the session management exit routine to translate the DLU real name to the DLU alias name.
4. VTAM drives the exit routine to translate the DLU real name to the DLU alias name.
5. Same description as number 4.

PLU-initiated session (ONLY-SHR)

Figure 14 on page 232 depicts a PLU-initiated session in an ONLY-SHR gateway configuration. All SME invocations are depicted. The alias selection invocations are described in the details below the figure.

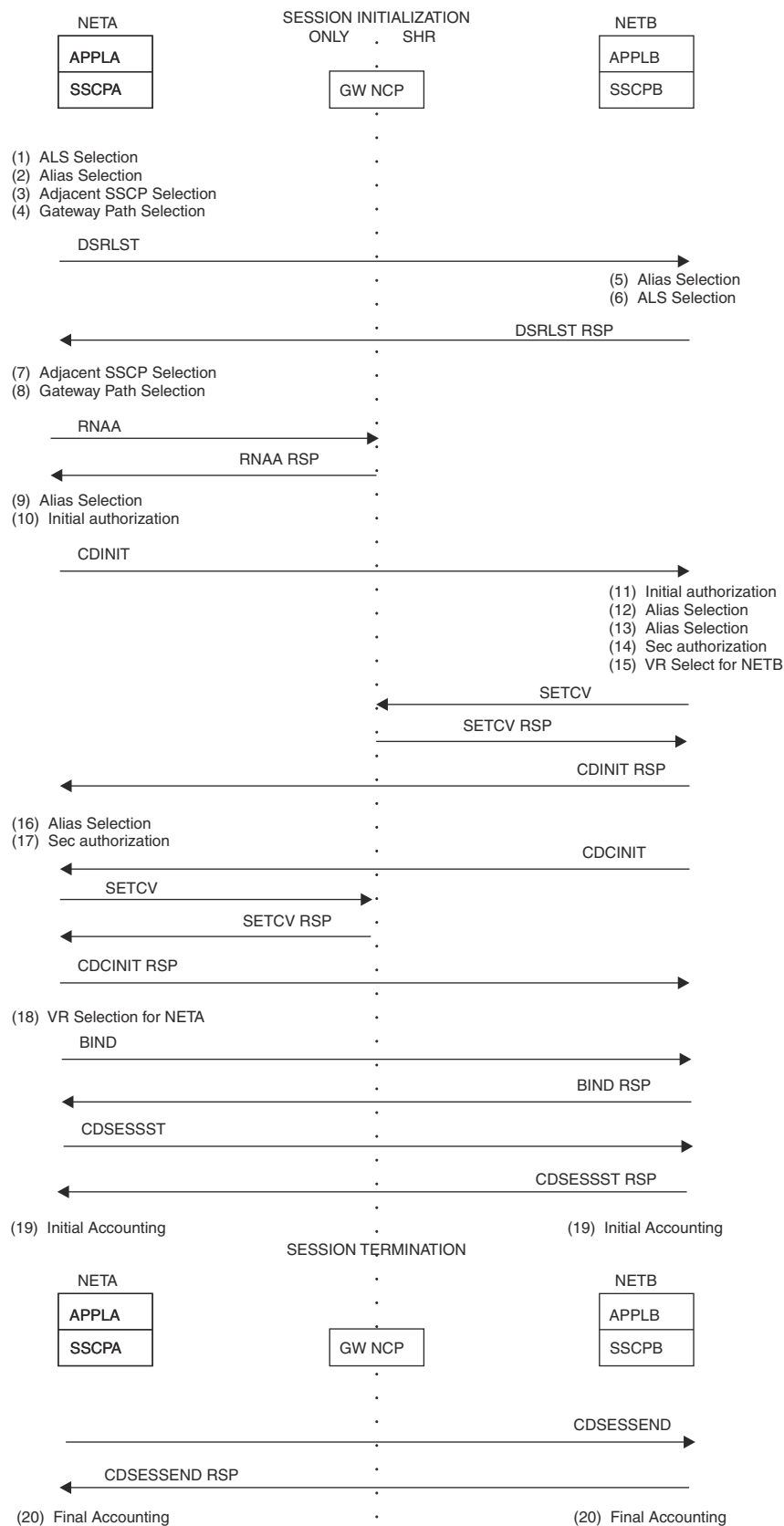


Figure 14. PLU-initiated flows for ONLY-SHR

2

VTAM passes the DLU alias name, the OLU real name, the associated LU real names, and the logon mode name as known in the OLU's network. The alias selection function has the option of returning

the DLU real name, the network ID of the DLU real name, the owning SSCP of the DLU, the OLU and associated LU alias names as known in the DLU network, and the logon mode name as known in the DLU network.

5

VTAM passes the DLU alias name. The alias selection function has the option of returning the DLU real name, the network ID of the DLU real name, and the owning SSCP of the DLU.

9

VTAM requests translations from NETA to NETB for the OLU alias name, the associated LU alias names, and the logon mode name. VTAM also requests the owning SSCP for the DLU.

12

VTAM requests translations from NETA to NETB for the OLU alias name, the associated LU alias names, and the logon mode name.

13

VTAM drives the exit routine to translate the COS name from NETB to NETA. No attempt is made to translate the associated LU names because this host has already attempted to translate them.

16

Same description as number 13.

PLU-initiated session (SHR-SHR)

Figure 15 on page 234 depicts a PLU-initiated session in a SHR-SHR gateway configuration. All SME invocations are depicted. The alias selection invocations are described in the details below the figure.

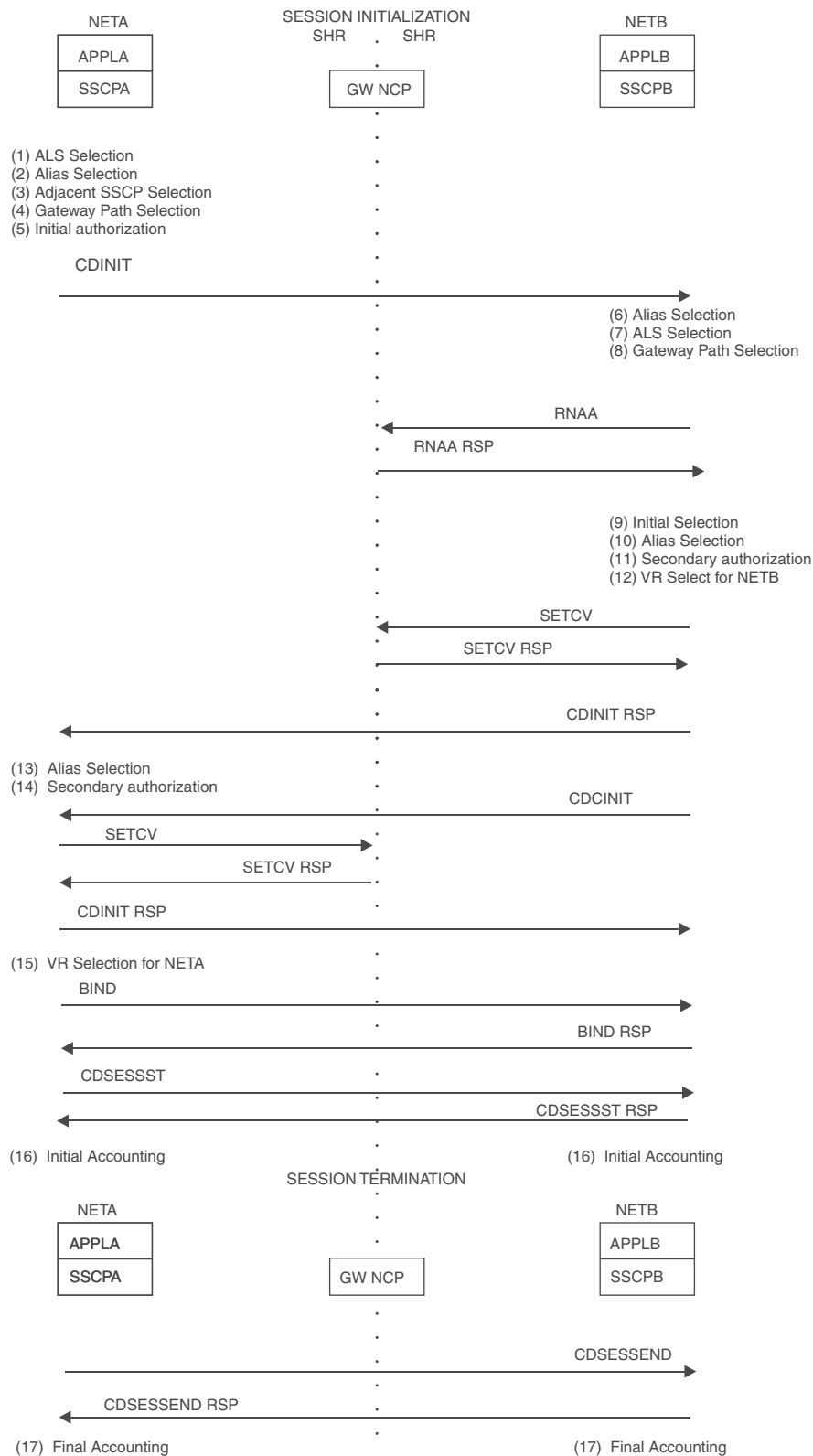


Figure 15. PLU-initiated flows for SHR-SHR

2

VTAM passes the DLU alias name, the OLU real name, the associated LU real names, and the logon mode name as known in the OLU's network. The alias selection function has the option of returning the DLU real name, the network ID of the DLU real name, the owning SSCP of the DLU, the OLU and

associated LU alias names as known in the DLU network, and the logon mode name as known in the DLU network.

6

Same description as number 2.

10

VTAM requests translations from NETA to NETB for the OLU alias name, the associated LU alias names, the logon mode name, the COS name, and to translate the COS name from NETB to NETA.

13

VTAM drives the exit routine to translate the COS name from NETB to NETA. No attempt is made to translate the associated LU names because this host has already attempted to translate them.

Virtual route selection for boundary function LUs

Figure 16 on page 235 depicts flows during virtual route selection for boundary function LUs.

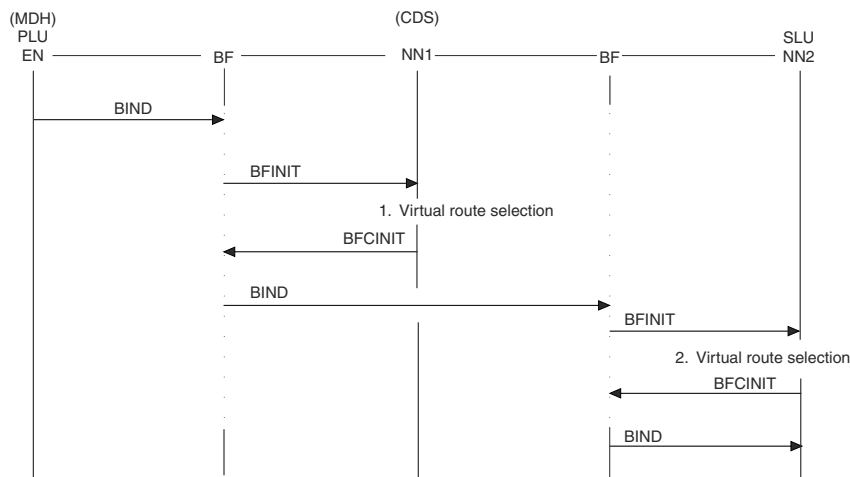


Figure 16. Virtual route selection for boundary function LUs

Session flows for APPN

Figure 17 on page 236 through Figure 21 on page 240 depict session flows in an APPN environment or in a mixed subarea/APPN environment. In these flow diagrams, a broken line connecting NN1 and NN2 (-//-) indicates that there can be multiple network nodes (the exit calls at each intervening network node are identical). The dashed line connecting ICN1 and ICN2 (= = =) indicates SSCP-SSCP sessions.

Note: Bind processing is illustrated in Figure 17 on page 236 and Figure 21 on page 240. For associated flows for virtual route selection for boundary function LUs, see Figure 16 on page 235.

Mixed subarea/APPN ILU-initiated session

Figure 17 on page 236 and Figure 18 on page 237 depict flows for an ILU-initiated session in a mixed subarea and APPN network.

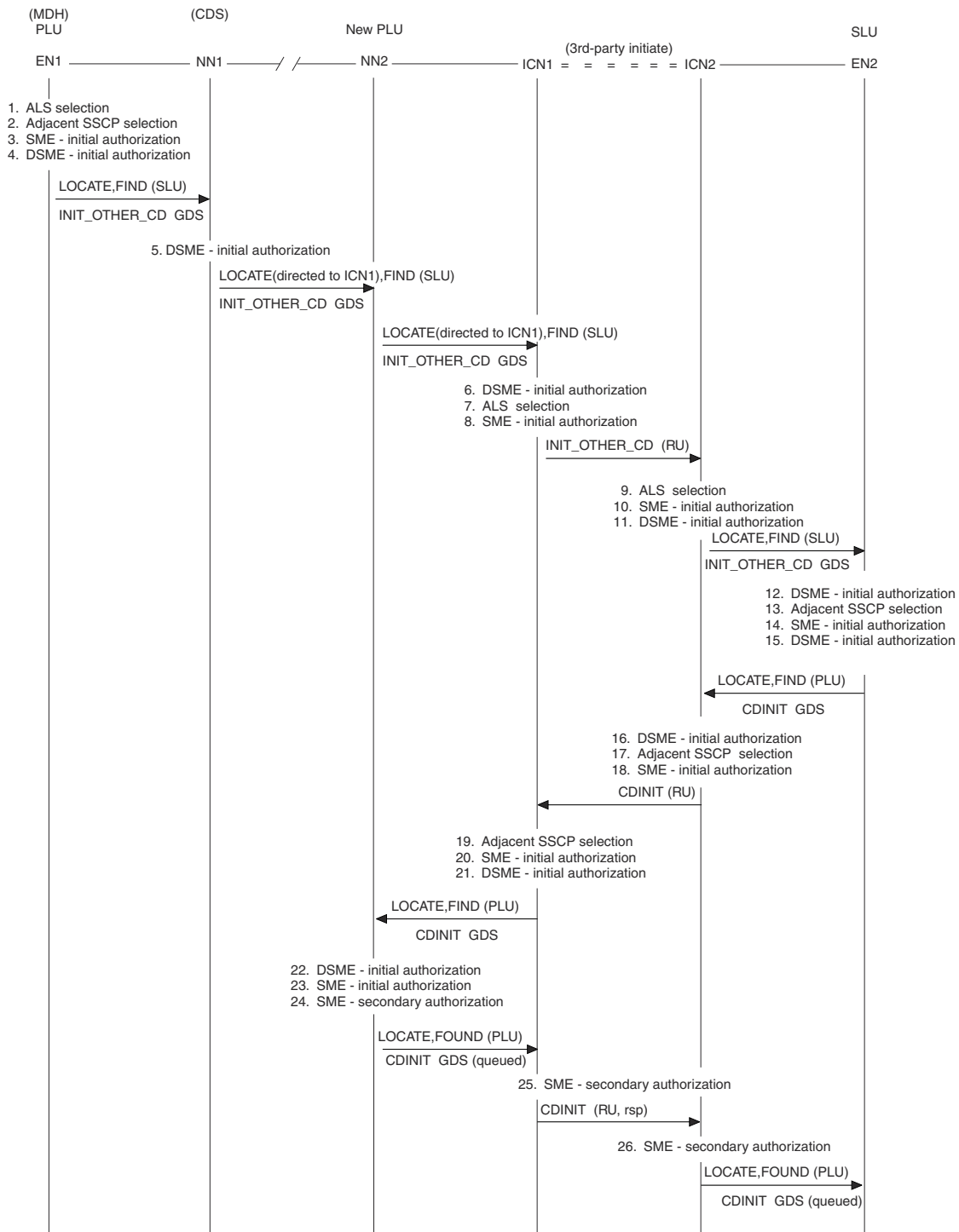


Figure 17. Mixed subarea/APPN ILU-initiated session, part 1

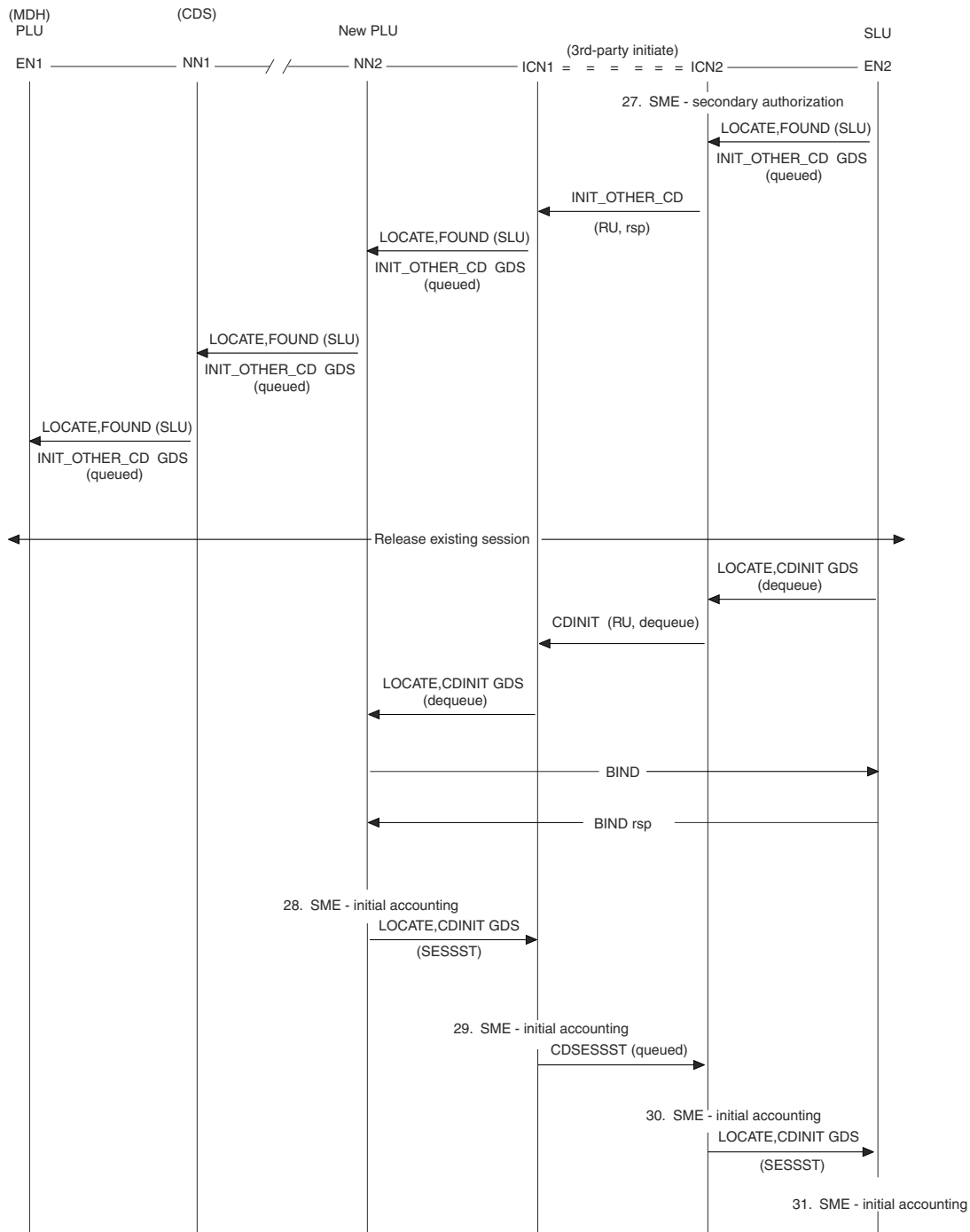


Figure 18. Mixed subarea/APPN ILU-initiated session, part 2

Mixed subarea/APPN PLU-initiated network broadcast

Figure 19 on page 238 and Figure 20 on page 239 depict flows for a PLU-initiated session in a mixed subarea and APPN network; this session is not queued.

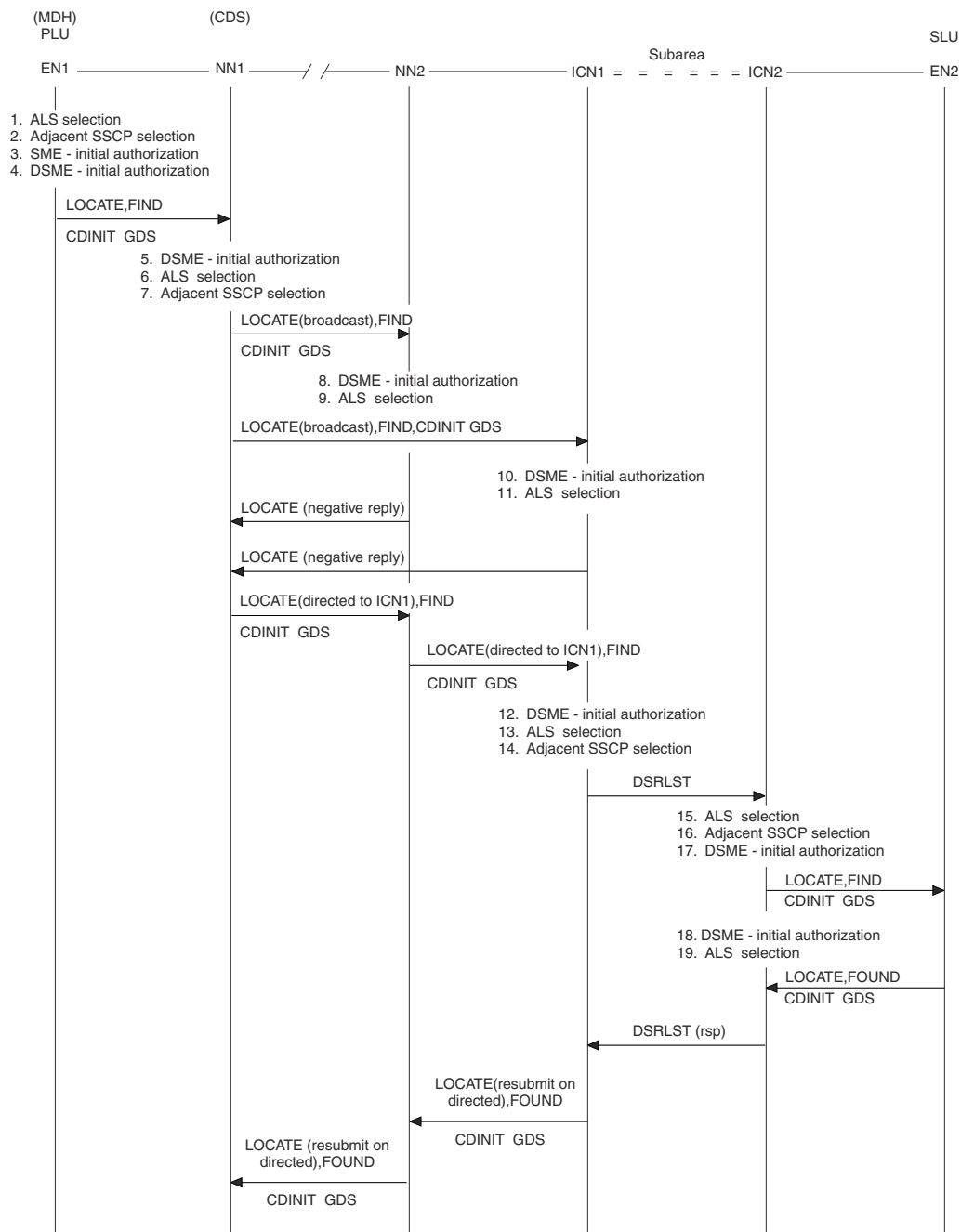


Figure 19. Mixed subarea/APPN PLU-initiated network broadcast, part 1

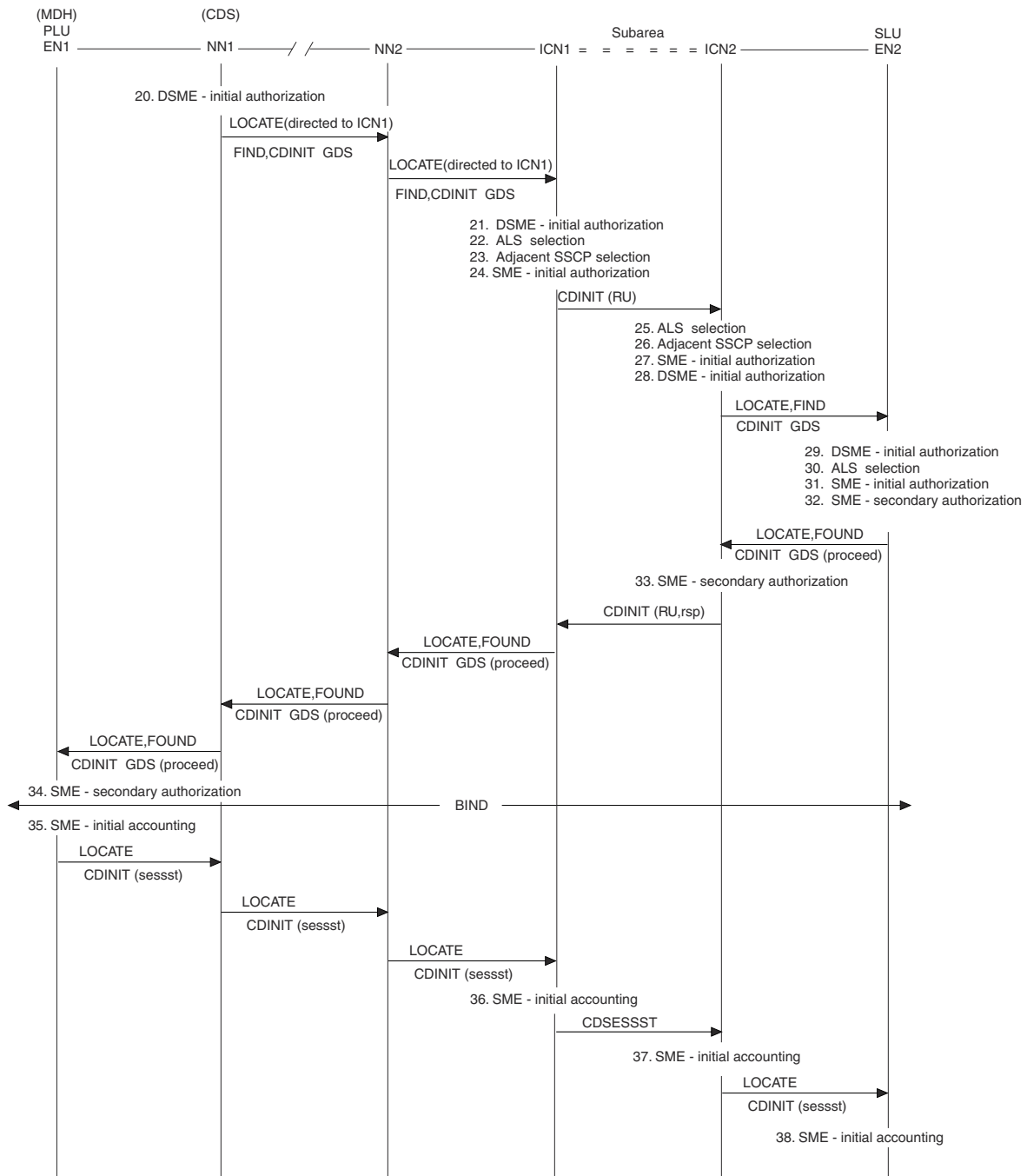


Figure 20. Mixed subarea/APPN PLU-initiated network broadcast, part 2

Mixed subarea/APPN SLU-initiated directed search

Figure 21 on page 240 depicts flows for a SLU-initiated directed search in a mixed subarea and APPN network.

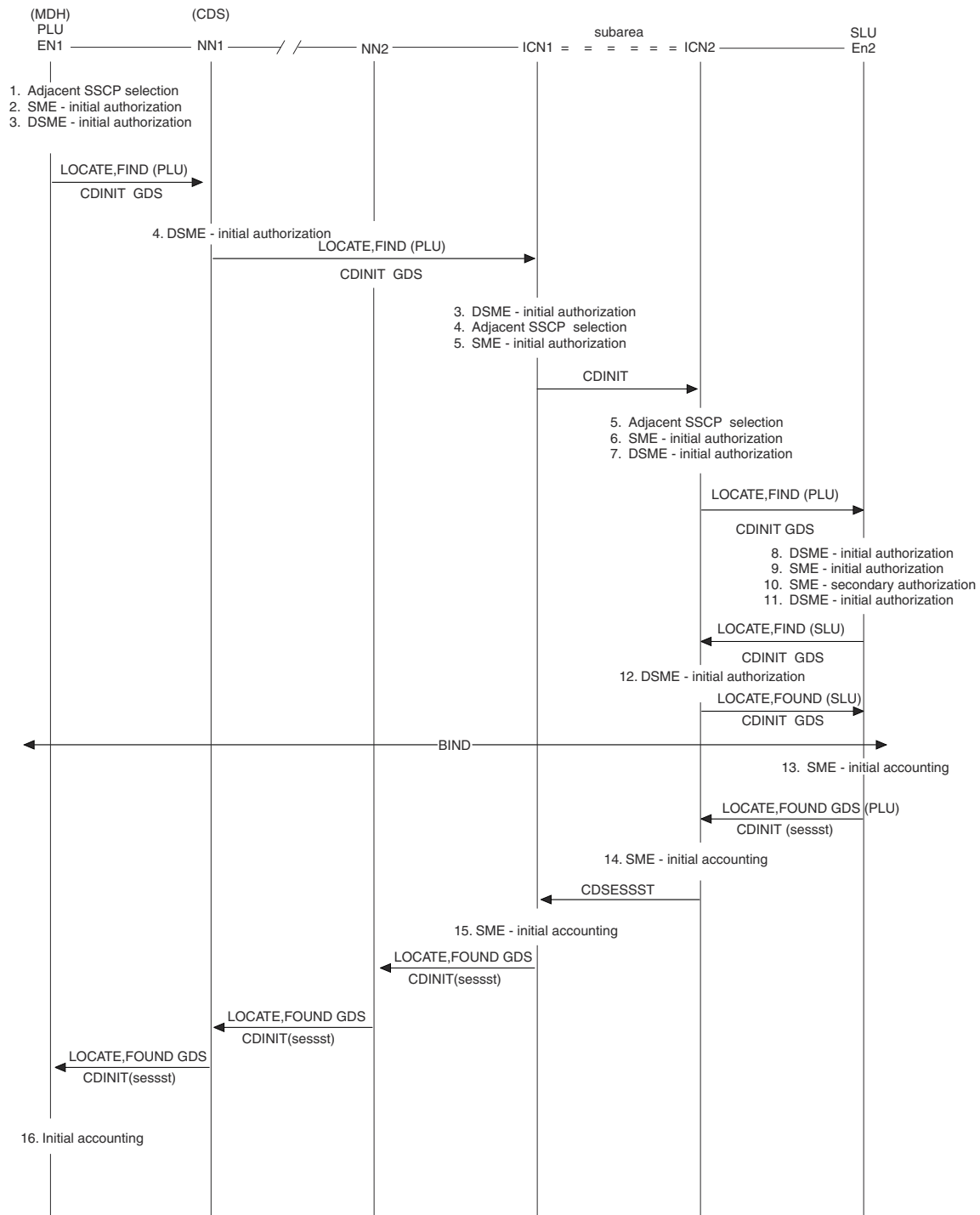


Figure 21. Mixed subarea/APPN SLU-initiated directed search

Appendix C. Sample session management exit

The session management exit routine controls various functions associated with VTAM logical unit session processing. These functions include:

- Initial and secondary session authorization
- Initial and final session accounting
- Gateway path selection
- XRF session switch
- Adjacent SSCP selection
- Alias selection
- ALS selection
- Exit replacement
- Exit replaced

For a complete description of the exit functions, see [“Session management exit routine” on page 1](#). If you write a session management exit routine, you should use the interface information described in that section.

The sample session management exit routine in this topic is only prototype code that can be included in a more complete and comprehensive session management exit routine. The sample code illustrates how the initial session management exit routine environment is established, the type of information that is available in each parameter list for the specific sample environment, and some techniques used to examine the various parameter lists. The sample session management exit routine is written specifically for the SNA network environment illustrated in [Figure 22 on page 242](#). The sample uses only the following session management exit routine functions:

- Begin
- Secondary session authorization
- Gateway path selection
- Alias selection
- End

A portion of a final accounting function is also included. However, the code is related specifically to the alias function. The initial and final accounting functions are normally used to record session connection time.

Note: The examples in this topic are intended as an educational tool. They show how various functions in a session management exit routine can be used to control VTAM functions. The examples shown are coded to a specific network environment and are not compatible with any other installation configuration or processing requirements.

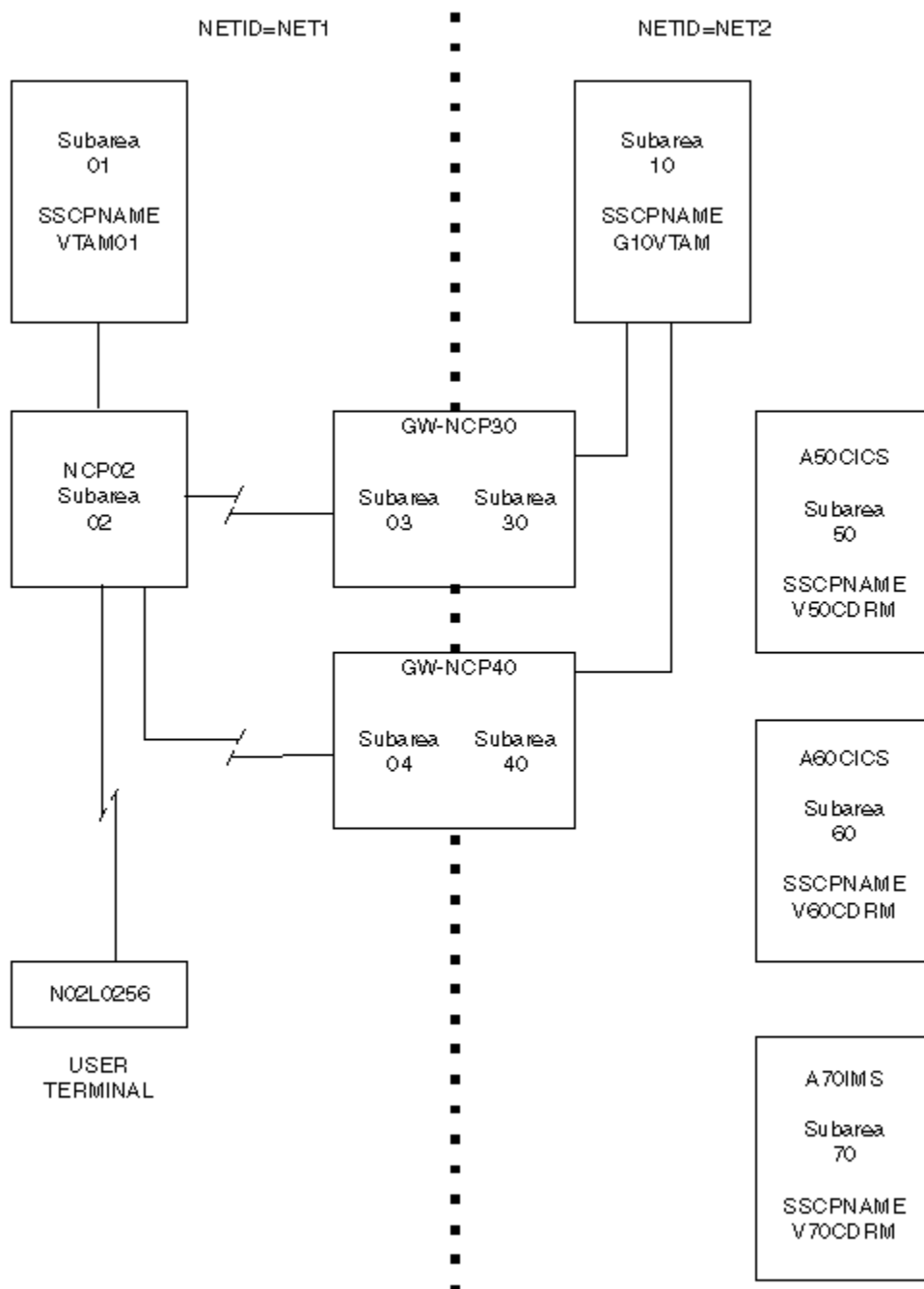


Figure 22. Network environment for sample session management exit routine

Initialization

Each time VTAM invokes the session management exit routine, the exit routine must perform its initial programming functions. For example, the session management exit routine must save the register contents into a save area provided by VTAM. The exit routine entry point address, which establishes the necessary routine addressing, is contained in register 15.

The prologue code that performs these initial tasks follows.

```
*****
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
R11     EQU      11
R12     EQU      12
R13     EQU      13
R14     EQU      14
R15     EQU      15
*****
*          SME ROUTINE PROLOGUE
*****
ISTEXCAA CSECT
*
** SAVE REGISTERS IN VTAM PROVIDED SAVEAREA
*
          STM     R14,R12,12(R13)  SAVE VTAM'S REGISTERS
*
** ESTABLISH SME ROUTINE ADDRESSABILITY
*
INITS1   EQU      *
          LR      R12,R15           ESTABLISH ADDRESSABILITY
          USING   ISTEXCAA,R12     BASE REGISTER USED EQUALS R12
          B       INITS1A         SAVE PARAMETER LIST ADDRESS
*
** MODULE NAME AND ASSEMBLY DATE
*
          DC      CL8'ISTEXCAA'    CSECT NAME
          DC      CL8'&SYSDATE'    ASSEMBLY DATE
**
*
INITS1A   EQU      *
          LR      R11,R1           ADDRESS FUNCTION PARAMETER LIST
*
** ESTABLISH SME ROUTINE REGISTER SAVE AREA
*
INITS1B   EQU      *
          LH      R2,GMSIZESA      SIZE OF REGISTER SAVE AREA
          LH      R3,GMSPNO        SUBPOOL NUMBER FOR REGISTER SAVE AREA
          GETMAIN RC,LV=(R2),SP=(R3) REQUEST REGISTER SAVE AREA STORAGE
          LTR     R15,R15          DETERMINE IF GETMAIN REQUEST SUCCESSFUL
          BNZ     ABEND            NO, STOP PROCESSING
INITS1C   EQU      *
          ST      R13,4(R1)        BACKWARD SAVE AREA POINTER
          ST      R1,8(R13)        FORWARD SAVE AREA POINTER
          LR      R13,R1           SME ROUTINE REGISTER SAVE AREA
          B       INITS2          CONTINUE INITIALIZATION PROCESSING
GMSIZESA  DC      H'72'           SIZE OF REGISTER SAVE AREA
GMSPNO    DC      H'0'            SUBPOOL OF GETMAIN'ED AREA
*
** TERMINATE SME ROUTINE ABNORMALLY DUE TO ERRORS:
** COMPLETION CODE=999, REASON=SMEF
*
          L       R8,=X'E2D4C5C6'
ABEND     ABEND 999,REASON=(R8),DUMP
*****

```

Function selection

VTAM invokes the session management exit routine to perform various LU-session processing functions. Design your session management exit routine to process these functions. You indicate in the begin function which LU-session processes require VTAM to invoke the session management exit routine. After VTAM invokes the session management exit routine, the routine must read the function code in the function parameter list to determine which LU-session processing to perform.

The sample session management exit routine coding compares the exit routine function code to a value associated with the functions selected. The function code values for this sample exit routine follow:

Function

Begin

X'FE'

Secondary session authorization

X'01'

Final session accounting

X'03'

Gateway path selection

X'04'

Alias selection

X'07'

End

X'FF'

The address of the exit routine function code is always the second field in the function parameter list. The session management exit routine examines the code and branches to the appropriate processing within the session management exit routine. The following code illustrates this technique:

```
*****
INITS2 EQU *
*
** LOCATE EXIT ROUTINE FUNCTION CODE
*
      L      R1,4(R11)      EXIT ROUTINE FUNCTION CODE ADDRESS
*
** EXAMINE EXIT ROUTINE FUNCTION CODE TO DETERMINE LU
** SESSION PROCESSING
*
      CLI    0(R1),X'FE'     TEST FOR BEGIN FUNCTION
      BE     BEGIN          YES, BRANCH TO FUNCTION PROCESSING
*
      CLI    0(R1),X'01'     TEST FOR SECONDARY SESSION AUTHORIZATION
      BE     SECAUTH         YES, BRANCH TO FUNCTION PROCESSING
*
      CLI    0(R1),X'03'     TEST FOR FINAL SESSION ACCOUNTING
      BE     ALIASS4         YES, BRANCH TO FUNCTION PROCESSING
*
      CLI    0(R1),X'04'     TEST FOR GATEWAY PATH SELECTION
      BE     GWPATH          YES, BRANCH TO FUNCTION PROCESSING
*
      CLI    0(R1),X'07'     TEST FOR ALIAS SELECTION
      BE     ALIAS           YES, BRANCH TO FUNCTION PROCESSING
*
      CLI    0(R1),X'FF'     TEST FOR END FUNCTION
      BE     END             YES, BRANCH TO FUNCTION PROCESSING
*
      B      RETURN
*
*****
```

Begin function

The begin function is a required element in any session management exit routine. This function is processed only once. It is processed before any of the other functions and is either initialized immediately following VTAM initialization, or is activated by issuing a MODIFY EXIT,ID=*exitname*,OPTION=ACT command. See [z/OS Communications Server: SNA Operation](#) for more information about this command. The begin function indicates to VTAM the functions for which the exit routine is to be invoked.

The fourth entry in the parameter list passed to the begin function contains the address of the exit routine functions. The session management exit routine must set the proper bit on (B'1') for each function for which VTAM is to call the routine. The bit settings for each of the functions associated with the sample routine follow:

Function**Secondary session authorization**

Byte 0, bit 1 or X'40'

Final accounting

Byte 0, bit 2 or X'20'

Gateway path selection

Byte 0, bit 3 or X'10'

Alias selection

Byte 1, bit 2 or X'20'

End function

Byte 0, bit 4 or X'08'

The fifth word in the parameter list passed to the BEGIN function contains the address of a 2-byte length field followed by the character string that was entered on the MODIFY EXIT, ID=ISTEXCAA,OPT=INACT,PARMS= parameter. If the exit is being driven during VTAM initialization, or the PARMS= parameter was not entered, this pointer will be zero.

Obtaining user storage

In this example, the begin function sets these indicators in the exit options field. It also obtains storage, using a conditional GETMAIN macroinstruction, to maintain information used by the various function routines. If the conditional GETMAIN fails, the exit terminates with an ABEND 999 and with REASON=SMEF. The format of the user storage area is shown in [Figure 23 on page 245](#).

The function parameter list, which is passed by VTAM to each exit routine, contains the address of a user data field. This field stores the address of the storage area acquired by the begin function. Functions subsequently called by VTAM can then read the user data field.

DEC(HEX)

Offset

0 (00)	TABLE INDICATOR
4 (04)	ADDRESS OF TABLE
8 (08)	PREFERRED GWPATH NAME
16 (10)	PREFERRED GWPATH SUBAREA
20 (14)	OLD GWPATH NAME
28 (1C)	OLD GWPATH SUBAREA ADDRESS
32 (20)	ALIAS NAME POOL ADDRESS
36 (24)	RESERVED
40 (28)	RESERVED
44 (2C)	RESERVED

Figure 23. User data storage area format

Loading NETID registration table

The sample code below for the begin function also loads a table that contains registered, valid network identifiers. The session management exit routine functions use this table to ensure that an LU session request is initiated from or directed to a registered network. In this example, the begin function sets indicators in the user storage area to inform other exit routine functions that the table is loaded and contains valid entries. The format of the network identifier registration table is shown in [Figure 24 on page 246](#).

DEC(HEX) Offset	
0 (0)	TABLE ENTRY COUNT
4 (4)	NETWORK IDENTIFIER #1
12 (C)	NETWORK IDENTIFIER #2
20 (14)	
NN (nn)	NETWORK IDENTIFIER #N

Figure 24. Network identifier registration table

The begin function can load the table at VTAM initialization or as a result of issuing the MODIFY EXIT command (see [z/OS Communications Server: SNA Operation](#) for information about using this command to activate, deactivate, or replace certain installation-wide exit routines). Additionally, there are commands that can prompt the session management exit routine to reload the table dynamically.

For example, when a new table is ready to be loaded, the system programmer can issue the following VTAM operator command to initiate a *dummy* LU session:

```
VARY NET,LOGON=RELOAD,ID=res
```

where *res* is some known resource.

The command in this example is issued at the gateway VTAM (G10VTAM). In an attempt to initiate the dummy LU session between the secondary logical unit, *res*, and the primary logical unit, RELOAD, VTAM invokes the specified session management exit routine functions. For example, the alias selection function can determine whether the session request is for the dummy LU session. When the session management exit function discovers the session setup is for the dummy LU session, the network identifier registration table is reloaded. After the table is reloaded, the function routine can reject the session request with a proper return code to VTAM.

If the load of the network identifier registration table fails, for example a network identifier table is not supplied, an indicator, X'F0', is set in the first byte of the user data storage area. If there are no entries in the table, for example table entry count is zero, the session management exit begin function sets a table indicator, X'FF', in the user data storage area. Any session management exit function routines subsequently invoked by VTAM use these table indicators. If the table indicator is set to X'FF' or to X'F0', the function routine does not allow a cross-network session to be established. The routine exits with the proper return code.

The begin function also obtains another user data storage area for use by the session management exit alias selection function. This storage is initialized and maintained by the session management exit routine. The begin function performs a conditional GETMAIN request to obtain 4096 bytes of virtual storage for an alias name pool and saves the address of the storage in the user data storage area. For more information concerning how this table is used, see [“Alias selection function”](#) on page 256.

When issuing the VARY NET,LOGON=RELOAD, ID=LUNAME, the LU must be known, active, and not in session if you want the SME to be called immediately. Otherwise, the LOGAPPL session and the SME call will occur after the current session ends.

The following sample code obtains the user data storage, loads a network identifier registration table, and sets the bits appropriately in the exit options field.

```
*****
*           SME ROUTINE - BEGIN FUNCTION
```



```

*****
BEGIN    EQU    *
*
** ISSUE GETMAIN TO OBTAIN USER DATA STORAGE AREA
*
BEGINS1  EQU    *
        LH      R2,GMSIZEUD      SIZE OF USER DATA STORAGE
        LH      R3,GMSPNO        SUBPOOL OF USER DATA STORAGE
        GETMAIN RC,LV=(R2),SP=(R3) REQUEST USER DATA STORAGE AREA
        LTR     R15,R15          DETERMINE IF GETMAIN SUCCESSFUL
        BZ      BEGINS1A         YES, CONTINUE PROCESSING
        B       ABEND            TERMINATE SME ROUTINE
GMSIZEUD DC    H'48'            SIZE OF USER DATA STORAGE
*
** SAVE ADDRESS OF USER DATA STORAGE IN USER DATA FIELD
*
BEGINS1A EQU    *
        L       R2,8(R11)        PARAMETER LIST - USER DATA FIELD ADDRESS
        ST      R1,0(R2)         SAVE USER DATA STORAGE - GETMAIN
        LR      R10,R1           RETAIN ADDRESS OF USER DATA STORAGE
        XC      0(48,R10),0(R10) CLEAR USER DATA STORAGE AREA
*
** LOAD NETID REGISTRATION TABLE
*
BEGINS2  EQU    *
        LA      R2,BEGINS2A      LOAD ERROR CONDITION RETURN ADDRESS
        LOAD    EP=NETIDTAB, LOAD MODULE NAME - NETID REGISTRATION TABLEX
        ERRET=(R2)              LOAD ERROR RETURN ROUTINE
        B       BEGINS2B         NO, CONTINUE PROCESSING
*
** LOAD FAILURE - SET TABLE INDICATOR
**
** NO NETID REGISTRATION TABLE:
**
** X'F0' - INDICATE TO OTHER SME FUNCTIONS DO NOT
**        PERMIT CROSS NETWORK SESSIONS TO SETUP
*
BEGINS2A EQU    *
        MVI     0(R10),X'F0'     LOAD FAILURE - SET TABLE INDICATOR
        B       BEGINS3         RETURN TO VTAM
*
** SAVE LOAD MODULE ADDRESS OF NETID REGISTRATION TABLE IN
** USER DATA STORAGE AREA AND TEST FOR VALID NETID ENTRIES
*
BEGINS2B EQU    *
        ST      R0,4(R10)        SAVE LOAD MODULE ADDRESS - USER DATA STORAGE
        LR      R3,R0            ADDRESS OF LOAD MODULE
        L       R2,0(R3)         LOAD TABLE ENTRY COUNT
        LTR     R2,R2            TEST ENTRY COUNT GREATER THAN ZERO
        BNZ     BEGINS3         CONTINUE PROCESSING
*
** INVALID NETID REGISTRATION TABLE - SET TABLE INDICATOR
**
** NO VALID NETID ENTRIES:
**
** X'FF' - INDICATE TO OTHER SME FUNCTIONS DO NOT
**        PERMIT CROSS NETWORK SESSIONS TO SETUP
*
BEGIN2C  EQU    *
        MVI     0(R10),X'FF'     INVALID NETID TABLE - SET TABLE INDICATOR
*
** OBTAIN STORAGE FOR ALIAS NAME POOL
*
BEGINS3  EQU    *
        LH      R2,GMSIZEAP      SIZE OF ALIAS NAME POOL STORAGE
        LH      R3,GMSPNO        SUBPOOL NUMBER FOR NAME POOL STORAGE
        GETMAIN RC,LV=(R2),SP=(R3) REQUEST STORAGE FOR ALIAS
                                NAME POOL STORAGE
*
        LTR     R15,R15          DETERMINE IF GETMAIN SUCCESSFUL
        BZ      BEGINS3A         YES, CONTINUE PROCESSING
        MVI     32(R10),X'FF'    NO ALIAS NAME POOL - SET INDICATOR
        B       BEGINS4         YES, CONTINUE PROCESSING
GMSIZEAP DC    H'4096'          SIZE OF ALIAS NAME POOL STORAGE
*
** SAVE ADDRESS OF ALIAS NAME POOL STORAGE IN USER DATA STORAGE
*
BEGINS3A EQU    *
        ST      R1,32(R10)       SAVE ALIAS NAME POOL STORAGE ADDRESS
        LA      R0,16            INITIALIZE COUNTER TO CLEAR STORAGE
BEGINS3B EQU    *
        XC      0(256,R1),0(R1)  CLEAR ALIAS NAME POOL STORAGE
        LA      R1,256(R1)       INCREMENT ADDRESS TO NEXT 256 STORAGE BYTES

```

```

      BCT   R0,BEGINS3B      CONTINUE PROCESSING ALL STORAGE
*
** SET SME EXIT OPTIONS FOR SUBSEQUENT SME FUNCTIONS
*
BEGINS4 EQU *
      L     R2,12(R11)       PARAMETER LIST - EXIT OPTIONS ADDRESS
      OI    0(R2),X'40'      SME OPTION - SECONDARY AUTHORIZATION
      OI    0(R2),X'20'      SME OPTION - FINAL ACCOUNTING
      OI    0(R2),X'10'      SME OPTION - GATEWAY PATH SELECTION
      OI    1(R2),X'20'      SME OPTION - ALIAS SELECTION
      OI    0(R2),X'08'      SME OPTION - END FUNCTION
      B     RETURN           RETURN TO VTAM
*****

```

Returning to VTAM

Upon completion of the above tasks, the begin function returns to VTAM. It is not executed again until VTAM is reinitialized or until the exit is reinitialized following use of the MODIFY EXIT command (see [z/OS Communications Server: SNA Operation](#) for information about using this command to activate, deactivate, or replace certain installation-wide exit routines). The code that performs the return to VTAM must free the storage used as the register save area for the session management exit routine, reestablish the registers for VTAM, and ensure the return is properly set. All function routines can use the following code to exit to VTAM.

```

*****
*
** RETURN TO VTAM
*
RETURN EQU *
      L     R2,4(R13)        RELOAD VTAM'S REGISTER SAVE AREA ADDRESS
      ST    R15,16(R2)       SET SME FUNCTION RETURN CODE IN VTAM'S
*                             REGISTER 15
      FREEMAIN R,LV=72,A=(R13) FREE REGISTER SAVE AREA STORAGE
      LR    R13,R2           RESET REGISTER 13 TO VTAM'S SAVE AREA ADDRESS
      LM    R14,R12,12(R13)  RESTORE VTAM'S REGISTERS
      BR    R14              EXIT TO VTAM
*****

```

Secondary authorization function

Because this sample session management exit routine does not use the initial authorization function, all LU sessions are authorized by the secondary authorization function. This sample session management exit routine function examines the network ID of the secondary and primary logical units (SLU and PLU).

If the network ID associated with both LUs is the same as the host's network ID, GWVTAM, that is passed to the routine in the environment vectors, the session setup is allowed to continue. However, if the network ID of either the SLU or the PLU differs from the host's network ID, the network identifier registration table is examined.

If the table indicator in the first byte of the user data storage area is set to either X'FF' or X'F0', the cross-network session setup is rejected. If the table indicator is not set to that value, the network ID of the appropriate LU is compared to the entries in the network identifier registration table to determine if the request is valid. If the network ID is not contained in the table, the session request is rejected.

The secondary authorization function parameter list contains the addresses of the PLU and the SLU resource identification control vectors (RICs). These RICs contain the network identifier associated with both the real and alias LU names.

Using the sample network environment shown in [Figure 22 on page 242](#), assume there is a cross-network session request from a terminal (N02L0256) in NET1 for a destination logical unit (A50CICS) in NET2. A sample portion of the PLU RIC that is passed to the session management exit routine in the gateway VTAM (G10VTAM) for this terminal-initiated session is shown in [Figure 25 on page 249](#).

DEC(HEX) Offset	
0 (00)	19 (VECTOR KEY)
1 (01)	nn (VECTOR LENGTH)
2 (02)	00 (REROUTE COUNT)
3 (03)	40 (USAGE INDICATOR)
4 (04)	08 (SSCPNAME LENGTH)
5 (05)	V50VTAM (SSCPNAME)
12 (0C)	04 (REAL NETID LENGTH)
13 (0D)	NET2 (REAL NETID NAME)
17 (11)	08 (REAL LU NAME LENGTH)
18 (12)	A50CICS (REAL LU NAME)
25 (19)	08 (ALIAS NETID LENGTH)
26 (1A)	NET1 (ALIAS NETID NAME)
30 (1E)	08 (ALIAS LU NAME LENGTH)
31 (1F)	A50CICS (ALIAS LU NAME)
32 (20)	1A (CONTROL VECTOR KEY)

Figure 25. Sample portion of PLU resource identification control vector

RICs contain more information related to the LU session request than is used by the sample session management exit routine. For example, the PLU and SLU RICs contain the real and alias names of the LU session partners. The session management exit routine can be expanded to authorize session setup based on the LU name of the DLU or OLU partner. The session management exit routine might analyze a specific naming convention associated with the LUs, or the network identifier registration table might be expanded to include LU name registration, depending upon the security requirements of the network or application session partners.

After it receives the RIC, the session management exit routine must locate the network ID of the gateway VTAM host. This information is the first entry following the vector list header.

```
*****
*      SME ROUTINE - SECONDARY AUTHORIZATION FUNCTION
*****
SECAUTH EQU *
        L      R2,0(R11)      ADDRESS OF ENVIRONMENT VECTORS
        CLI    3(R2),X'06'    ENSURE HOST NETID VECTOR
        BNE    ABEND          TERMINATE - NO NETID CONTROL VECTOR
        SR     R3,R3          CLEAR WORK REGISTER 3
        IC     R3,2(R2)       LENGTH OF HOST NETID
        LTR    R3,R3          ENSURE HOST NETID AVAILABLE
        BZ     ABEND          TERMINATE - NO HOST NETID START OPTION
        LA     R2,4(R2)       ADDRESS OF HOST NETID
*****
```

The next task is to examine the network identifiers that are passed for the PLU and SLU in the RICs. There are two network IDs in each RIC, one associated with the real name and the other with the alias name of the LU.

In this sample, the only network identifier that is important is the real network ID for each LU. The real network ID in each RIC must be examined to determine the direction of session setup. The direction of session setup is determined by which LU initiates the session, the mechanism used to initiate the session (terminal-initiated logon, automatic logon, or third-party logon), or both the LU and mechanism used.

The usage indicator (byte 4, bit 1) in the RIC identifies whether the resource, either PLU or SLU, is the target. Therefore, this indicator can be checked to determine whether the RIC for the PLU is the DLU or the OLU. The same task can be performed for the SLU by examining the indicator in the RIC.

The following sample code is used to examine the real network ID in the RICs for the primary and secondary LUs. The network ID in the RIC for the PLU is first compared to the host network ID. If both are the same, the real network ID in the RIC for the SLU must be examined. If the PLU and host network IDs are not the same, the real network ID for the PLU is compared to the entries in the network identifier registration table. If there is no match, the session request is considered to be from a network that is not valid and the session is rejected.

Note: In the sample, an execute (EX) instruction is used to compare the network IDs and the PLU or the SLU names because the name length in the RIC appears to be variable. VTAM, however, always uses 8-character names that are padded with blanks. Therefore, you are not required to use the EX instruction. You can use a compare (CLC) instruction with a length of eight to perform this task.

```
*****
*
** LOCATE PLU RESOURCE INFORMATION CONTROL VECTOR - REAL NETID
*
SECAUS1 EQU *
        LA R15,0          RETURN CODE - ASSUME ACCEPT SESSION REQUEST
        SR R5,R5          CLEAR REGISTER 5
        L  R4,12(R11)     PLU RIC ADDRESS
        LA R4,4(R4)       PLU RIC SSCPNAME ADDRESS
        IC R5,0(R4)       PLU RIC SSCPNAME LENGTH
        LTR R5,R5         TEST SSCPNAME FOR ZERO LENGTH
        BZ SECAUS3C       NO SSCPNAME - SESSION SETUP FAILURE
        AR R4,R5          PLU RIC REAL NETID ADDRESS VECTOR MINUS ONE
        LA R4,1(R4)       PLU RIC NETID LENGTH ADDRESS
        IC R5,0(R4)       PLU RIC NETID LENGTH
        LTR R5,R5         PLU RIC TEST NETID FOR ZERO LENGTH
        BZ SECAUS3C       PLU RIC NETID LENGTH INCORRECT
        BCTR R5,0         PLU RIC NETID LENGTH MINUS ONE
        LA R4,1(R4)       PLU RIC NETID ADDRESS
*
** COMPARE THE PLU RIC NETID TO HOST NETID IN THE ENVIRONMENT VECTOR
** IF PLU RIC NETID EQUALS HOST NETID THEN PROCESS SLU RIC NETID
** ELSE EXAMINE NETID REGISTRATION TABLE FOR VALID NETWORK IDENTIFIER
*
        EX R4,NETID       COMPARE PLU RIC REAL NETID TO HOST NETID
        BNE SECAUS3       PLU NETID NOT EQUAL GATEWAY VTAM
*
** LOCATE SLU RESOURCE INFORMATION CONTROL VECTOR - REAL NETID
*
SECAUS2 EQU *
        SR R5,R5          CLEAR WORK REGISTER 5
        L  R4,16(R11)     SLU RIC ADDRESS
        LA R4,4(R4)       SLU RIC SSCPNAME ADDRESS
        IC R5,0(R4)       SLU RIC SSCPNAME LENGTH
        LTR R5,R5         TEST SSCPNAME FOR ZERO LENGTH
        BZ SECAUS3C       NO SSCPNAME - SESSION SETUP FAILURE
        AR R4,R5          SLU RIC REAL NETID ADDRESS VECTOR MINUS ONE
        LA R4,1(R4)       SLU RIC NETID LENGTH ADDRESS
        IC R5,0(R4)       SLU RIC NETID LENGTH
        LTR R5,R5         SLU RIC TEST NETID FOR ZERO LENGTH
        BZ SECAUS3C       SLU RIC NETID LENGTH INCORRECT
        BCTR R5,0         SLU RIC NETID LENGTH MINUS ONE
        LA R4,1(R4)       SLU RIC NETID ADDRESS
*
** COMPARE THE SLU RIC NETID TO HOST NETID IN THE ENVIRONMENT VECTOR
** IF SLU RIC NETID EQUALS HOST NETID THEN MUST BE SAME NETWORK SESSION
** ELSE EXAMINE NETID REGISTRATION TABLE FOR VALID NETWORK IDENTIFIER
*
        EX R4,NETID       COMPARE SLU RIC REAL NETID TO HOST NETID
        BE RETURN         SLU NETID EQUALS GATEWAY VTAM - SAME NETWORK
*
** EXAMINE NETID REGISTRATION TABLE FOR VALID NETWORK IDENTIFIER
*
SECAUS3 EQU *
```

```

      L      R2,8(R11)      USER DATA FIELD ADDRESS
      L      R2,0(R2)      USER DATA STORAGE AREA ADDRESS
      CLI    0(R2),X'F0'    TEST FOR NETID TABLE LOADED
      BE     SECAUS3C       NETID TABLE NOT LOADED - REJECT
*
      CLI    0(R2),X'FF'    TEST FOR INVALID NETID TABLE
      BE     SECAUS3C       NETID TABLE INVALID - REJECT
*
      L      R2,4(R2)      NETID REGISTRATION TABLE ADDRESS
      L      R3,0(R2)      NETID TABLE ENTRY COUNT
      LA     R2,4(R2)      NETID TABLE ENTRY ADDRESS
*
** COMPARE THE PLU OR SLU RIC REAL NETID TO NETID REGISTRATION
** TABLE ENTRY
*
SECAUS3A EQU *
      EX     R5,NETID      COMPARE PLU/SLU RIC NETID TO TABLE ENTRY
      BE     RETURN        ACCEPT SESSION REQUEST - VALID NETID ENTRY
SECAUS3B EQU *
      LA     R2,8(R2)      NEXT NETID TABLE ENTRY ADDRESS
      BCT    R3,SECAUS3A   NETID TABLE ENTRY COUNT NOT
                          ZERO - CONTINUE SEARCH
*
*
** END NETID REGISTRATION TABLE SEARCH - NO VALID ENTRY OR NETID
** NOT LOADED.
** RETURN CODE - REJECT CROSS NETWORK SESSION REQUEST
*
SECAUS3C EQU *
      LA     R15,8         NO ENTRY IN NETID REGISTRATION TABLE
                          OR NETID TABLE NOT LOADED
*
      B      RETURN        EXIT TO VTAM
*
** INSTRUCTION EXECUTED TO COMPARE: PLU RIC REAL NETID TO HOST NETID
**
** PLU RIC REAL NETID TO NETID
** REGISTRATION TABLE ENTRY
** SLU RIC REAL NETID TO HOST NETID
** SLU RIC REAL NETID TO NETID
** REGISTRATION TABLE ENTRY
*
NETID    CLC    0(0,R4),0(R2)  NETID COMPARE INSTRUCTION
*
*****

```

Gateway path selection function

The gateway path selection function can be used when a SNA network interconnection environment uses multiple gateway NCPs. The gateway VTAM always attempts to establish a cross-network session by using the same gateway NCP over which the session already is established between two CDRMs. You can use the session management exit routine to modify this default gateway path selection. The gateway path selection function can either reorder the list of available gateway NCPs or delete entries from the list.

Each entry in the list is associated with a GWPATH definition statement following a CDRM definition statement. For example, assume the following CDRM and GWPATH definition statements are coded in the gateway VTAM (G10VTAM) in NET2:

```

VTAM01  CDRM
GWNCP30 GWPATH SUBAREA=30,ADJNET=NET1,ADJNETSA=1,ADJNETEL=1
GWNCP40 GWPATH SUBAREA=40,ADJNET=NET1,ADJNETSA=1,ADJNETEL=1

```

Assuming that both gateway NCPs (GWNCP30 and GWNCP40) are operative and all of the links are active, the first gateway NCP establishes the SSCP-SSCP session between the gateway VTAM (G10VTAM) and the VTAM host (VTAM01) in NET1.

The list of GWPATH definition statements passed to the session management exit routine is in the order in which they are coded. The gateway NCP that establishes the SSCP-SSCP session, however, is always placed first in the list. Therefore, in the example above, if GWNCP30 was not initially available for the SSCP-SSCP session, but was operative later, the order of the GWPATH entries would be reversed in the list.

Even though the gateway NCP is operative, links between the gateway NCP and the adjacent subarea might not be operative. As a result, a specific gateway NCP is selected to set up an LU session; however, the route from gateway NCP to the destination subarea is not available.

If the VTAM in NET1 (VTAM01) is a gateway SSCP that has multiple paths into network NET2, the gateway path selection function is called at session initiation to determine which gateway NCP should be used for the session. When called, the gateway path selection exit in VTAM01 is passed a list of paths arranged in the same order as the GWPATH macros in the cross-net CDRM definition, except that the NCP used for the CDRM-CDRM session is first in the list. The gateway path selection exit reorders the list and passes the results back to VTAM01. VTAM01 selects the first usable NCP from the reordered list for the session.

If the SSCP on the other side of the gateway (G10VTAM) can also function as a gateway SSCP into NET1, it will pass a single entry list to its gateway path selection exit. The single entry identifies the gateway NCP selected by VTAM01. Information about the gateway NCP selected by VTAM01 is delivered to G10VTAM on the CDINIT.

If VTAM01 and G10VTAM are in a back-to-back configuration separated by a null net, gateway path selection will be driven in both SSCPs. However, the CDINIT from the OLU network will not contain any information about the gateway NCP selected in the OLU network. The gateway path selection exit is driven in both VTAM01 and G10VTAM with the full list of gateway NCPs into the null network.

Examining LU names

In the sample network, assume that GWNCP40 is selected for the session between the terminal (N02L0256) and the application program (A50CICS). Route activation for this session does not take place until the BIND flows for the session. In this case, the BIND flows from the VTAM host (V50CDRM) to the gateway NCP (GWNCP40). Assume that the link is not operative between the gateway NCP (GWNCP40) and the NCP subarea (NCP02) in NET1. As a result, the virtual route (VR) cannot be activated and the LU-LU session fails. There is no information provided to the session management exit routine indicating that the route is unavailable, even though the gateway NCP is available. If the user at the terminal attempts to reinitiate the session, the session management exit routine, using the same logic, continues to select the same gateway NCP over which the complete network route is unavailable.

In the sample session management exit routine, the gateway path selection function performs the following logic to select a GWPATH from the list:

1. Examines the real LU name of the application program in the PLU and the SLU RIC to determine if the LU name is either A50CICS or A60CICS. If the DLU or OLU is neither of these names, the session management exit code uses the default gateway path selection.
2. If the real LU name is A50CICS, ensure that GWNCP30 is the first entry in the list. If the LU name is A60CICS, the session management exit reorders the list, if necessary, to ensure that GWNCP40 is the first entry in the GWPATH selection list

The following code locates the real LU name for the PLU or SLU. If the LU name is not a CICS® application program, the exit returns control to VTAM without modifying the GWPATH list.

```
*****
*      SME ROUTINE - GATEWAY PATH SELECTION FUNCTION
*****
GWPATH  EQU  *
*
** LOCATE USER DATA STORAGE AREA - GWPATH NAME AND SUBAREA ADDRESS
*
GWPTH1A EQU  *
        L    R1,8(R11)      USER DATA FIELD ADDRESS
        L    R1,0(R1)       USER DATA STORAGE AREA ADDRESS
*
** LOCATE PLU RESOURCE INFORMATION CONTROL VECTOR - REAL LUNAME
*
GWPTH1B EQU  *
        SR   R5,R5          CLEAR WORK REGISTER 5
        L    R4,12(R11)     PLU RIC ADDRESS
        LA   R4,4(R4)       PLU RIC SSCPNAME ADDRESS
        IC   R5,0(R4)       PLU RIC SSCPNAME LENGTH
        AR   R4,R5          PLU RIC REAL NETID ADDRESS VECTOR MINUS ONE
        LA   R4,1(R4)       PLU RIC NETID LENGTH ADDRESS
        IC   R5,0(R4)       PLU RIC NETID LENGTH
        AR   R4,R5          PLU RIC REAL LUNAME ADDRESS VECTOR MINUS ONE
        LA   R4,1(R4)       PLU RIC REAL LUNAME LENGTH ADDRESS
        IC   R5,0(R4)       PLU RIC REAL LUNAME LENGTH
        LTR  R5,R5         TEST PLU REAL NAME FOR ZERO LENGTH
```

```

        BE    GWPTH51C      YES, NO REAL PLU NAME
        BCTR  R5,0          PLU RIC REAL LUNAME LENGTH MINUS ONE
        LA    R4,1(R4)      PLU RIC REAL LUNAME ADDRESS
*
** COMPARE THE PLU RIC LUNAME TO APPLICATION NAME: A50CICS OR A60CICS
*
        EX    R5,CICS50      COMPARE PLU RIC REAL LUNAME TO ACICS50
        BE    GWPTH52A      YES, REORDER GWPATH AS NECESSARY
        EX    R5,CICS60      COMPARE PLU RIC REAL LUNAME TO ACICS60
        BE    GWPTH52B      YES, REORDER GWPATH AS NECESSARY
*
** LOCATE SLU RESOURCE INFORMATION CONTROL VECTOR - REAL LUNAME
*
GWPTH51C EQU *
        SR    R5,R5          CLEAR WORK REGISTER 5
        L     R4,16(R11)     SLU RIC ADDRESS
        LA    R4,4(R4)       SLU RIC SSCPNAME ADDRESS
        IC    R5,0(R4)       SLU RIC SSCPNAME LENGTH
        AR    R4,R5          SLU RIC REAL NETID ADDRESS VECTOR MINUS ONE
        LA    R4,1(R4)       SLU RIC NETID LENGTH ADDRESS
        IC    R5,0(R4)       SLU RIC NETID LENGTH
        AR    R4,R5          SLU RIC REAL LUNAME ADDRESS VECTOR MINUS ONE
        LA    R4,1(R4)       SLU RIC REAL LUNAME LENGTH ADDRESS
        IC    R5,0(R4)       SLU RIC REAL LUNAME LENGTH
        LTR   R5,R5          TEST SLU REAL NAME FOR ZERO LENGTH
        BE    GWPTHDEF       YES, NO SLU REAL NAME - USE DEFAULT GWPATH
        BCTR  R5,0          SLU RIC REAL LUNAME LENGTH MINUS ONE
        LA    R4,1(R4)       SLU RIC REAL LUNAME ADDRESS
*
** COMPARE THE SLU RIC LUNAME TO APPLICATION NAME: A50CICS OR A60CICS
*
        EX    R5,CICS50      COMPARE SLU RIC REAL LUNAME TO ACICS50
        BE    GWPTH52A      YES, REORDER GWPATH AS NECESSARY
        EX    R5,CICS60      COMPARE SLU RIC REAL LUNAME TO ACICS60
        BE    GWPTH52B      YES, REORDER GWPATH AS NECESSARY
*
** IF NEITHER THE PLU OR SLU RIC REAL LUNAME EQUALS A50CICS OR A60CICS
** THEN SET RETURN CODE TO PROCEED WITH DEFAULT GWPATH SELECTION LIST
*
GWPTHDEF EQU *
        LA    R15,4          RETURN CODE TO USE DEFAULT GWPATH LIST
        B     RETURN         RETURN CONTROL TO VTAM
*
** PLU OR SLU RIC REAL LUNAME EQUALS A50CICS -
** SAVE FIRST GWPATH ENTRY NAME AND SUBAREA ADDRESS IN USER
** DATA STORAGE
*
GWPTH52A EQU *
        MVC   8(8,R1),GWNC30  FIRST GWPATH NAME MUST BE GWNC30
        LA    R5,30           FIRST GWPATH SUBAREA MUST BE 30
        ST    R5,16(R1)       SAVE IN USER DATA STORAGE
        B     GWPTH53         CONTINUE, REORDER GWPATH LIST IF NECESSARY
*
** PLU OR SLU RIC REAL LUNAME EQUALS A60CICS -
** INITIALIZE REGISTERS TO MODIFY GWPATH SELECTION LIST AS NECESSARY
*
GWPTH52B EQU *
        MVC   8(8,R1),GWNC40  FIRST GWPATH NAME MUST BE GWNC40
        LA    R5,40           FIRST GWPATH SUBAREA MUST BE 40
        ST    R5,16(R1)       SAVE IN USER DATA STORAGE
        B     GWPTH53         CONTINUE, REORDER GWPATH LIST IF NECESSARY
*
** INSTRUCTIONS EXECUTED TO COMPARE: PLU RIC REAL LUNAME TO A50CICS
**                                     OR A60CICS
**                                     SLU RIC REAL LUNAME TO A50CICS
**                                     OR A60CICS
*
CICS50    CLC   0(8,R4),=CL8'A50CICS'  LUNAME COMPARE INSTRUCTION
CICS60    CLC   0(8,R4),=CL8'A60CICS'  LUNAME COMPARE INSTRUCTION
*
** CONSTANTS - GATEWAY NCP NAMES
*
GWNC30    DC    CL8'GWNC30'           GATEWAY NCP NAME - GWNC30
GWNC40    DC    CL8'GWNC40'           GATEWAY NCP NAME - GWNC40
*
*****

```

Modifying gateway path selection list

After the session management exit routine determines that the real LU name is associated with CICS, the gateway path selection function reorders the gateway path selection list as appropriate. If the application program name is A50CICS, the first entry in the list must be GWNCP30. Conversely, if the LU name is A60CICS, the first GWPATH entry must be GWNCP40. The above sample code locates the LU name in the RIC and, if the proper CICS name is determined, branches to the sample code below. In this case, the user data storage area is initialized with the gateway NCP name and the corresponding subarea address of the first entry in the list.

The session management exit routine must next locate the GWPATH selection list. Because only one gateway NCP might be available, the gateway path function must determine if the list contains more than one entry. If there is only one entry in the list, the exit routine returns to VTAM and uses the default list. However, if there is more than one entry in the gateway selection list, the code must read the first entry and reorder the list, if necessary.

The gateway path selection list is reordered by moving the first entry name and subarea address into user data storage and moving the preferred gateway NCP name and subarea address to the first entry in the list. The entry that was previously the first entry in the list must then be moved from user data storage back into the gateway path list to complete the swap.

If the gateway path NCP name (GWN) is blank, the preferred gateway NCP is inactive. In that case, the first entry in the selection list must be reset to the original name and subarea address.

Note: In the sample, the GWN is checked to determine whether the preferred gateway NCP is inactive because the SUBAREA operand was used on the GWPATH definition statement. If you use the GWN operand instead of the SUBAREA operand, the subarea address is set to 0, indicating that the preferred gateway path is inactive. If both operands are coded, there is no way to determine whether that gateway is inactive.

The following sample session management exit code performs these tasks. [Figure 26 on page 254](#) illustrates the contents of the entries in the gateway path selection list for the sample network environment.

DEC(HEX)

Offset	
0 (0)	nn (TOTAL LENGTH)
2 (2)	NET1 (with 4 blanks)
10 (A)	0002
12 (C)	14
13 (D)	GWNCP30 (with 1 blank)
21 (15)	0000001E
25 (19)	NET1 (with 4 blanks)
33 (21)	14
34 (22)	GWNCP40 (with 1 blank)
21 (15)	00000028
25 (19)	NET1 (with 4 blanks)

Figure 26. Gateway path selection list for sample session management exit routine

```
*****
GWPTHS3 EQU *
        LA R15,4          RETURN CODE ASSUMES DEFAULT GWPATH LIST
        L  R2,28(R11)      GATEWAY PATH SELECTION LIST ADDRESS
        LH R3,10(R2)       OBTAIN THE NUMBER OF GWPATH ENTRIES
```



```

        LA      R2,12(R2)      FIRST GWPATH ENTRY ADDRESS
        LA      R4,1           INITIALIZE REGISTER WITH VALUE OF ONE (1)
        CR      R3,R4          GWPATH ENTRIES GREATER THAN ONE
        BH      GWPTHS3A       YES, LOCATE NAME OF FIRST GWPATH ENTRY
        B       RETURN         RETURN CONTROL TO VTAM
*
** EXAMINE FIRST GWPATH SUBAREA ADDRESS (SUBAREA OPERAND)
**
** COMPARE SUBAREA ADDRESS ENTRY TO PREFERRED ADDRESS IN USER
** DATA STORAGE. IF BOTH ENTRIES EQUAL RETURN TO VTAM AND USE
** DEFAULT SELECTION LIST.
*
GWPTHS3A EQU *
        CLC     9(4,R2),16(R1)  FIRST GWNCPC SAME AS PREFERRED GWPATH
        BE      RETURN         PREFERRED GWNCPC FIRST, USE DEFAULT LIST
*
** FIRST GWPATH ENTRY NOT EQUAL TO PREFERRED GWPATH
** SAVE CURRENT FIRST ENTRY IN SELECTION LIST IN USER DATA STORAGE
** SWAP PREFERRED ENTRY WITH FIRST ENTRY IN SELECTION LIST
*
GWPTHS3B EQU *
        MVC     20(8,R1),1(R2)  SAVE GWNAME IN USER DATA STORAGE
        MVC     28(4,R1),9(R2)  SAVE SUBAREA ADDRESS IN USER DATA
*                               STORAGE
        MVC     1(8,R2),8(R1)   SWAP PREFERRED GWNAME FROM USER DATA
*                               STORAGE
        MVC     9(4,R2),16(R1)  SWAP PREFERRED SUBAREA ADDRESS FROM
*                               USER DATA STORAGE
        BCTR    R3,0            NUMBER OF GWPATH ENTRIES
*                               REMAINING IN SELECTION LIST
*
** LOCATE PREFERRED GWPATH ENTRY IN SELECTION LIST TO COMPLETE SWAP
**
** COMPARE THE NEXT SUBAREA ADDRESS ENTRY IN THE GWPATH SELECTION LIST
** TO THE PREFERRED ENTRY IN USER DATA STORAGE. IF EQUAL, ENSURE THE
** PREFERRED GWPATH IS NOT INACTIVE. COMPARE THE GWNCPC NAME TO BLANKS.
** IF ACTIVE, COMPLETE THE GWPATH SWAP, OTHERWISE RESET THE FIRST ENTRY
** TO THE ORIGINAL CONDITION.
*
GWPTHS3C EQU *
        SR      R4,R4          CLEAR WORK REGISTER 4
        IC      R4,0(R2)       OBTAIN GWPATH ENTRY LENGTH
        LA      R2,1(R2)       GWPATH ENTRY ADDRESS
        AR      R2,R4          NEXT GWPATH ENTRY
        CLC     9(4,R2),16(R1)  GWPATH SUBAREA ADDRESS USED AS FIRST
*                               ENTRY
        BNE     GWPTHS3D       NO, EXAMINE NEXT ENTRY
        CLC     1(8,R2),=XL8'40' CHECK IF GWPATH NAME BLANK
        BE      GWPTHS3E       PREFERRED GWNCPC NOT ACTIVE, RESET
*                               FIRST ENTRY
        MVC     1(8,R2),20(R1)  SWAP FIRST ENTRY GWNAME FROM USER
*                               DATA STORAGE
        MVC     9(4,R2),28(R1)  SWAP FIRST ENTRY SUBAREA ADDRESS
*                               FROM USER DATA STORAGE
        LA      R15,0          RETURN CODE, USE MODIFIED GWPATH LIST
        B       RETURN         RETURN CONTROL TO VTAM
*
** EXAMINE NEXT ENTRY IN GWPATH SELECTION LIST
**
** IF LIST EXHAUSTED AND PREFERRED GWPATH ENTRY NOT LOCATED THEN
** SWAP FIRST ENTRY BACK TO ORIGINAL GWPATH BECAUSE THE PREFERRED
** GWPATH ENTRY IS NOT AVAILABLE IN THE LIST.
*
GWPTHS3D EQU *
        BCT     R3,GWPTHS3C     LAST GWPATH ENTRY, NO CONTINUE PROCESS
GWPTHS3E EQU *
        L       R2,28(R11)      GATEWAY PATH SELECTION LIST ADDRESS
        LA      R2,12(R2)       FIRST GWPATH ENTRY ADDRESS
        MVC     1(8,R2),20(R1)  SWAP FIRST ENTRY GWNAME FROM USER
*                               DATA STORAGE
        MVC     9(4,R2),28(R1)  SWAP FIRST ENTRY SUBAREA ADDRESS
*                               FROM USER DATA STORAGE
        B       RETURN         USE ORIGINAL DEFAULT GWPATH LIST
*****

```

Alias selection function

The alias selection function of the session management exit routine performs various name translation tasks. VTAM maintains both a real and an alias network-qualified name for each LU session partner. VTAM can invoke this function to determine any one of the following four names:

- Real network-qualified name for the PLU or DLU
- Alias network-qualified name for the PLU or DLU
- Real network-qualified name for the SLU or OLU
- Alias network-qualified name for the SLU or OLU

In addition, the alias selection function can map a class-of-service (COS) name for one network to the COS name used in another network. It can also map logon mode entry names or indicate the SSCP owner for a particular resource name.

In this sample, the alias function translates the real LU name of a terminal in NET1 into an alias name in NET2. The primary application programs (CICS and IMS) in NET2 require the terminal names to be generated in these subsystems. All terminals in NET1 that require access to these application programs must be generated into the subsystems. Because the subsystem maintenance and coordination associated with a large quantity of other-network terminals can be difficult, the alias function translates a real terminal name into an alias terminal name.

The alias terminal name is selected from a preassigned pool of alias names. This pool can be pre-generated into the IMS and CICS application programs. When all the names in the pool are allocated to concurrent cross-network sessions, subsequent session requests fail. After a name is freed as a result of a session termination, another cross-network session can be established. Therefore, the preassigned alias name pool needs to contain sufficient entries to accommodate the peak concurrent cross-network session count for CICS and IMS.

The session management exit routine must first examine the indicators associated with the LU names to determine what translation is necessary. For example, assume there is a terminal-initiated session from N02L0256 for the application program A70IMS in NET2. There are four network-qualified names associated with this session request, as shown in [Figure 27 on page 256](#).

	Origin Logical Unit	Destination Logical Unit
Real Name	NET1.N02L0256	NET2.A70IMS
Alias Name	NET2.ALIASnnn	NET1.A70IMS

Figure 27. Real and alias network-qualified names

Translating DLU name

For a terminal-initiated session that uses CDINIT processing (or DSRLST), the first entry in the alias input parameter list is the DLU, A70IMS. The alias function examines the OLU/DLU status indicator and the DLU name that requires translation. If the DLU name is a NET2 application program (IMS or CICS), the same LU name is used for the real LU name and only the NETID of NET2 needs to be supplied.

The session management exit routine initializes the output fields to a value of A70IMS for the LU name (RTNNAME1) and NET2 for the NETID (RTNNET1). However, if the first entry is not the DLU, or the LU name is not one of the NET2 application programs, the direction of session setup does not permit the alias function to provide an alias name; that is, a session initiated from the CICS or IMS application program requires the real name of the terminal. This sample session management exit routine cannot map an alias terminal that is pre-generated in IMS or CICS into a specific terminal name in the attached network (NET1). Therefore, this sample session management exit routine returns control to VTAM and

indicates that no translation is performed. The session setup proceeds assuming the correct names are supplied. The following sample code processes the first entry in the alias input parameter list:

```
*****
*           SME ROUTINE - ALIAS SELECTION
*****
ALIAS      EQU      *
*
** LOCATE THE ALIAS PARAMETER LIST - INPUT AND OUTPUT ADDRESSES
*
ALIASS1    EQU      *
          L          R2,44(R11)          ALIAS INPUT PARAMETER LIST ADDRESS
          LA         R2,8(R2)           ALIAS FIRST LUNAME ADDRESS - REAL/ALIAS NAME
          L          R3,48(R11)          ALIAS OUTPUT PARAMETER LIST ADDRESS
*
** PROCESS FIRST LOGICAL UNIT NAME IN ALIAS INPUT PARAMETER LIST:
**
** IF FIRST ENTRY IS THE DLU AND NAME IS 'A50CICS', 'A60CICS',
** OR 'A70IMS' RETURN SAME NAME TO VTAM AND CHANGE THE NETID TO 'NET2',
** OTHERWISE, EXIT TO VTAM WITHOUT ALIAS TRANSLATION.
**
** REASONS: 1. TERMINAL INITIATED SESSION TO APPLICATION WHICH DOES NOT
**            REQUIRE ALIAS TRANSLATION.
**            2. APPLICATION INITIATED SESSION FOR WHICH SME EXIT CANNOT
**            MAP THE ALIAS NAME TO A REAL TERMINAL NAME
**            3. CDINIT-OTHER PROCESSING FOR WHICH SME EXIT DOES NOT
**            PROVIDE NECESSARY TRANSLATION.
*
ALIASS1A    EQU      *
          TM          1(R2),X'20'          FIRST ENTRY - DLU NAME FOR
*                                           CDINIT/DSRLST PROCESSING
          BNO         ALIASS2              NO, ALIAS TRANSLATION NOT PERFORMED
          CLC         8(8,R2),=CL8'A50CICS' ALIAS NAME EQUAL A50CICS
*                                           APPLICATION
          BE          ALIASS1C             YES, RETURN SAME LUNAME AND NET2 NETID
          CLC         8(8,R2),=CL8'A60CICS' ALIAS NAME EQUAL A60CICS
*                                           APPLICATION
          BE          ALIASS1C             YES, RETURN SAME LUNAME AND NET2 NETID
          CLC         8(8,R2),=CL8'A70IMS' ALIAS NAME EQUAL A70IMS
*                                           APPLICATION
          BE          ALIASS1C             YES, RETURN SAME LUNAME AND NET2 NETID
ALIASS1B    EQU      *
          LA          R15,0                NO TRANSLATION PERFORMED - NO ALIAS
*                                           APPLICATION
          B           RETURN              RETURN TO VTAM
ALIASS1C    EQU      *
          MVC         0(8,R3),8(R2)        REAL DLU NAME EQUALS ALIAS APPLICATION
*                                           NAME
          MVC         48(8,R3),=CL8'NET2' REAL NETID FOR DLU EQUALS NET2
          B           ALIASS2              CONTINUE PROCESSING, SECOND ALIAS ENTRY
*****
```

Translating OLU name

The next step is to translate the OLU name. The indicator for the OLU fields (NAME2) specifies that the real name is provided and an alias name must be returned. The sample exit routine allocates up to 999 names. The names consist of a required prefix, ALIAS, and a 3-digit suffix allocated from an alias-suffix table. The exit routine initializes and maintains this table in a user storage area. An alias table can be loaded in the begin function similar to the network identifier registration table. The sample code below examines the second entry of the alias input parameters and invokes a subroutine to allocate an alias name from the table.

Note: For this example, the CDRSCTI start option must be set to 0 so that the alias LU name assigned to the terminal can be deleted and reassigned to another real terminal after the session is terminated. The CDRSCTI start option can have a value other than 0 for real operations.

```
*****
ALIASS2    EQU      *
          LA          R2,32(R2)          ALIAS SECOND LUNAME ADDRESS -
*                                           REAL/ALIAS NAME
*
** PROCESS SECOND LOGICAL UNIT NAME IN ALIAS INPUT PARAMETER LIST:
**
** IF SECOND ENTRY IS THE REAL NAME OF THE TERMINAL, THEN
** ALLOCATE AN ALIAS NAME FROM THE ALIAS NAME POOL, OTHERWISE,
```

```

** EXIT TO VTAM WITHOUT ALIAS TRANSLATION.
**
** REASON: 1. THE TERMINAL IS THE OLU AND THE NAME REQUESTED BY VTAM
**           IS THE REAL NAME FOR THE OLU AND THE SME ROUTINE ONLY
**           PROVIDES ALIAS NAMES.
*
ALIASS2A EQU *
          TM 2(R2),X'80' SECOND ENTRY - REAL NAME SUPPLIED
          BO ALIASS2C YES, PERFORM ALIAS TRANSLATION
ALIASS2B EQU *
          LA R15,8 SOME TRANSLATION PERFORMED - NO ALIAS
*           APPLICATION
          B RETURN RETURN TO VTAM
ALIASS2C EQU *
*
** IF THE INDICATOR IN THE FIRST BYTE OF THE ALIAS STORAGE ADDRESS IN
** THE USER DATA STORAGE AREA IS X'FF', THEN STORAGE WAS NOT AVAILABLE
** FOR THE TABLE. NO ALIAS PROCESSING CAN BE PERFORMED, THEREFORE EXIT.
**
** OTHERWISE, BRANCH TO SUBROUTINE THAT ALLOCATES AN AVAILABLE ALIAS
** NAME SUFFIX FROM THE ALIAS NAME POOL. IF ALL ENTRIES ARE USED,
** REGISTER 15 (R15) CONTAINS A NON-ZERO RETURN CODE FOR VTAM WHICH
** DOES NOT ALLOW SESSION INITIATION TO CONTINUE. OTHERWISE, THE
** ADDRESS FOR THE ALIAS NAME SUFFIX IS ADDRESSABLE AT DISPLACEMENT
** ONE (1) USING REGISTER 1 (R1). THE SUFFIX IS APPENDED TO THE FIVE
** CHARACTER STRING 'ALIAS' TO FORM THE ALIAS NAME FOR THE TERMINAL.
*
          L R4,8(R11) PARAMETER LIST - USER DATA ADDRESS
          L R4,0(R4) USER DATA STORAGE ADDRESS
          CLI 32(R4),X'FF' EXAMINE IF ALIAS NAME STORAGE OBTAINED
          BNE ALIASS2E YES, CONTINUE ALIAS NAME PROCESSING
ALIASS2D EQU *
          LA R15,20 NO ALIAS FUNCTION AVAILABLE - FAIL
*           SESSION SETUP
          B RETURN RETURN TO VTAM - NO ALIAS APPLICATION
ALIASS2E EQU *
          L R4,32(R4) ALIAS NAME STORAGE ADDRESS
          BAL R14,ALIASS3 BRANCH TO ALIAS NAME ALLOCATION ROUTINE
          LTR R15,R15 ALIAS SUFFIX AVAILABLE
          BNZ RETURN RETURN TO VTAM - NO ALIAS ENTRY AVAILABLE
ALIASS2F EQU *
          MVC 8(5,R3),=C'ALIAS' ALIAS OUTPUT PARAMETER - ALIAS NAME
*           PREFIX
          MVC 13(3,R3),1(R1) ALIAS OUTPUT PARAMETER - ALIAS NAME
*           SUFFIX
          MVC 56(8,R3),=CL8'NET2' ALIAS OUTPUT PARAMETER - ALIAS NETID
          LA R15,8 SOME TRANSLATION PERFORMED - NO COS, LOGMODE
          B RETURN RETURN TO VTAM
*****

```

The subroutine that processes the alias suffix table initializes the storage, obtained in the begin function, during the first pass through the allocation process. During the first pass, no entry is allocated, as indicated by a X'00' in the first byte. After the entry is allocated, a X'FF' is moved to the first byte in the alias suffix entry. A binary counter for the entry number is converted to a 3-byte decimal value, the suffix, and is saved in the remaining three bytes of the 4-byte entry. The address of the alias suffix entry is then returned to the alias function that invoked the subroutine. The suffix is appended to the prefix string ALIAS to form the alias name.

The format of the alias suffix table is illustrated in [Figure 28 on page 259](#). An example of the subroutine code that performs the suffix allocation follows [Figure 28 on page 259](#).

DEC(HEX) Offset	
0 (0)	WORKAREA
8 (8)	Allocation indicator: X 00 - unallocated X FF - allocated
9 (9)	Three Digit Suffix X F0F0F0
12 (C)	Allocation indicator: X 00 - unallocated X FF - allocated
13 (D)	Three Digit Suffix X F0F0F1
16 (10)	Allocation indicator: X 00 - unallocated X FF - allocated
17 (11)	Three Digit Suffix X F0F0F2
//	
3992 (398)	Allocation indicator: X 00 - unallocated X FF - allocated
3993 (399)	Three Digit Suffix X F9F9F9

Figure 28. Alias terminal name pool—3-digit suffix

```

*****
ALIASS3 EQU *
*
** ALIAS NAME SUFFIX ALLOCATION SUBROUTINE
**
** R1 = ADDRESS OF ALIAS NAME SUFFIX ENTRY ALLOCATED
** R4 = ADDRESS OF ALIAS NAME SUFFIX STORAGE - WORKAREA FOR CONVERT
**      TO DECIMAL
** R6 = ALIAS SUFFIX ENTRY COUNTER - DECREMENTED
** R7 = ALIAS SUFFIX ENTRY COUNTER - CONSTANT
** R14 = RETURN ADDRESS - MAIN ROUTINE OF ALIAS FUNCTION
** R15 = RETURN CODE
*
      LA R1,8(R4)      ALIAS SUFFIX ADDRESS - FIRST ENTRY
      LA R6,1000      ALIAS SUFFIX ENTRY COUNTER - DECREMENT
      LA R7,1000      ALIAS SUFFIX ENTRY COUNTER - CONSTANT
ALIASS3A EQU *
      CLI 0(R1),X'00'  ALIAS SUFFIX ENTRY ALLOCATED
      BNE ALIASS3B     YES, INCREMENT ADDRESS TO NEXT ENTRY
      MVI 0(R1),X'FF'  NO, ALLOCATE ALIAS SUFFIX ENTRY
      SR R7,R6         CALCULATE ALIAS SUFFIX ENTRY VALUE - BINARY
      CVD R7,0(R4)     CONVERT ALIAS SUFFIX ENTRY TO DECIMAL -
*                       WORKAREA
      UNPK 1(3,R1),6(2,R4) UNPACK SUFFIX VALUE INTO PROPER
*                       SUFFIX ENTRY
      OI 3(R1),X'F0'   ENSURE DECIMAL VALUE - CHANGE SIGN OF
*                       LAST CHARACTER
      BR R14           RETURN TO INVOKING ROUTINE

```

```

ALIASS3B EQU *
LA R1,4(R1)      INDEX ALIAS SUFFIX ADDRESS TO NEXT ENTRY
BCT R6,ALIASS3A  DECREMENT ALIAS SUFFIX COUNTER -
*                CONTINUE IF NONZERO
LA R15,20        ALL ALIAS SUFFIX ENTRIES ALLOCATED -
*                ENTRIES EXHAUSTED
BR R14           RETURN TO INVOKING ROUTINE - TERMINATE
*                SESSION SETUP
*****

```

Final accounting function

Because the alias selection function is not invoked at session termination, the session management exit routine can only allocate the alias suffix entries. Another session management exit function must deallocate the alias suffix entry when a session ends. The final accounting function, function code X'03', performs this task.

The final accounting function examines the alias LU name in the RIC of the PLU and the SLU. If the LU name contains a 5-character prefix of ALIAS, the session management exit routine can deallocate the proper alias suffix entry. This is done by converting the last 3 characters of the alias name to a binary value. That value is an index into the alias suffix storage area. The code then resets the allocation indicator from X'FF'—X'00'.

The entire final accounting function is not included in this sample; however, the portion of code that deallocates the appropriate entry at session termination follows:

```

*****
*
**      SME ROUTINE - FINAL ACCOUNTING FUNCTION - DEALLOCATE ALIAS
*                SUFFIX ROUTINE
*
ALIASS4 EQU *
L R1,8(R11)      USER DATA FIELD ADDRESS
L R1,0(R1)        USER DATA STORAGE AREA ADDRESS
LA R15,0         NO FINAL ACCOUNTING RETURN CODE
CLI 32(R1),X'FF'  TEST IF ALIAS SUFFIX STORAGE OBTAINED
BE RETURN        NO, ALIAS TRANSLATION NOT PERFORMED -
*                VTAM RETURN
L R1,32(R1)      ALIAS SUFFIX STORAGE ADDRESS
*
** LOCATE PLU RESOURCE INFORMATION CONTROL VECTOR - ALIAS LUNAME
*
ALIASS4A EQU *
SR R5,R5         CLEAR WORK REGISTER 5
L R4,12(R11)     PLU RIC ADDRESS
LA R4,4(R4)      PLU RIC SSCPNAME ADDRESS
IC R5,0(R4)      PLU RIC SSCPNAME LENGTH
AR R4,R5         PLU RIC REAL NETID ADDRESS VECTOR
*                MINUS ONE
LA R4,1(R4)      PLU RIC NETID LENGTH ADDRESS
IC R5,0(R4)      PLU RIC NETID LENGTH
AR R4,R5         PLU RIC REAL LUNAME ADDRESS VECTOR
*                MINUS ONE
LA R4,1(R4)      PLU RIC REAL LUNAME LENGTH ADDRESS
IC R5,0(R4)      PLU RIC REAL LUNAME LENGTH
AR R4,R5         PLU RIC ALIAS NETID ADDRESS VECTOR
*                MINUS ONE
LA R4,1(R4)      PLU RIC ALIAS NETID LENGTH ADDRESS
IC R5,0(R4)      PLU RIC ALIAS NETID LENGTH
AR R4,R5         PLU RIC ALIAS LUNAME ADDRESS VECTOR
*                MINUS ONE
LA R4,1(R4)      PLU RIC ALIAS LUNAME LENGTH
LA R4,1(R4)      PLU RIC ALIAS LUNAME ADDRESS
CLC 0(5,R4),=C'ALIAS'  PLU RIC ALIAS NAME EQUAL 'ALIAS'
BE ALIASS4C      PLU ALIAS NAME ALLOCATED BY SME -
*                DEALLOCATE
*
** LOCATE SLU RESOURCE INFORMATION CONTROL VECTOR - REAL LUNAME
*
ALIASS4B EQU *
L R4,16(R11)     SLU RIC ADDRESS
LA R4,4(R4)      SLU RIC SSCPNAME ADDRESS
IC R5,0(R4)      SLU RIC SSCPNAME LENGTH
AR R4,R5         SLU RIC REAL NETID ADDRESS VECTOR
*                MINUS ONE

```

```

        LA      R4,1(R4)      SLU RIC NETID LENGTH ADDRESS
        IC      R5,0(R4)      SLU RIC NETID LENGTH
        AR      R4,R5         SLU RIC REAL NETID ADDRESS VECTOR
*                               MINUS ONE
        LA      R4,1(R4)      SLU RIC REAL LUNAME LENGTH ADDRESS
        IC      R5,0(R4)      SLU RIC REAL LUNAME LENGTH
        AR      R4,R5         SLU RIC ALIAS NETID ADDRESS VECTOR
*                               MINUS ONE
        LA      R4,1(R4)      SLU RIC ALIAS NETID LENGTH ADDRESS
        IC      R5,0(R4)      SLU RIC ALIAS NETID LENGTH
        AR      R4,R5         SLU RIC ALIAS LUNAME ADDRESS VECTOR
*                               MINUS ONE
        LA      R4,1(R4)      SLU RIC ALIAS LUNAME LENGTH
        LA      R4,1(R4)      SLU RIC ALIAS LUNAME ADDRESS
        CLC     0(5,R4),=C'ALIAS' SLU RIC ALIAS NAME EQUAL 'ALIAS'
        BNE     RETURN        SLU RIC ALIAS NAME NOT ALLOCATED
*                               BY SME - VTAM RETURN
*
*
** ALIAS NAME SUFFIX DEALLOCATION SUBROUTINE
**
** R1 = ADDRESS OF ALIAS NAME SUFFIX STORAGE
** R2 = ALIAS SUFFIX ENTRY INDEX
** R4 = ADDRESS OF ALIAS NAME NAME - PLU/SLU RIC
** R14 = RETURN ADDRESS - MAIN ROUTINE OF ALIAS FUNCTION
*
ALIASS4C EQU *
        PACK   0(8,R1),5(3,R4) PACK ALIAS SUFFIX VALUE
        CVB    R2,0(R1)        CONVERT ALIAS SUFFIX ENTRY TO BINARY
*                               INDEX VALUE
        SLL    R2,2            OFFSET OF SUFFIX ENTRY
        LA     R1,8(R2,R1)     CALCULATE ALIAS SUFFIX ENTRY ADDRESS
        MVI    0(R1),X'00'     DEALLOCATE ALIAS SUFFIX ENTRY
        B      RETURN         EXIT TO VTAM
*****

```

End function

The end function of the session management exit is invoked during VTAM termination. The session management exit routine uses this function to perform any cleanup required before VTAM termination is completed. In this sample, the begin function obtained virtual storage for maintaining user data and loaded a network registration table. Therefore, the end function must delete the table and free the storage before returning control to VTAM. The following code performs these necessary functions:

```

*****
*       SME ROUTINE - END FUNCTION
*****
END     EQU *
*
** DELETE NETID REGISTRATION TABLE
*
        L      R2,8(R11)      LOCATE USER DATA FIELD IN PARAMETER LIST
        L      R2,0(R2)      OBTAIN ADDRESS OF USER DATA STORAGE
        CLI    0(R2),X'F0'    EXAMINE IF NETID TABLE LOADED
        BE     FREEANPS      NO - TABLE NOT LOADED
        DELETE EP=NETIDTAB    ISSUE DELETE MACRO TO CANCEL
*                               PREVIOUS LOAD
*
** FREE ALIAS NAME POOL STORAGE
*
FREEANPS EQU *
*
        L      R2,8(R11)      LOCATE USER DATA FIELD IN PARAMETER LIST
        L      R2,0(R2)      OBTAIN ADDRESS OF USER DATA STORAGE
        CLI    32(R2),X'FF'   EXAMINE IF ALIAS NAME STORAGE OBTAINED
        BE     FREEUDSA      NO - NOT OBTAINED
        L      R3,32(R2)      OBTAIN ADDRESS ALIAS NAME POOL STORAGE
        FREEMAIN R,LV=4096,A=(R3)
*
** FREE USER DATA STORAGE AREA
*
FREEUDSA EQU *
*
        FREEMAIN R,LV=48,A=(R2)
        B      RETURN
        LTORG

```

```
END
*****
```

The fifth word in the parameter list passed to the exit for the END function contains the address of a 2-byte length field followed by the character string that is entered on the MODIFY EXIT, ID=ISTEXCAA,OPT=INACT,PARMS= parameter. If the PARMS= parameter was not entered, this pointer will be zero.

VTAM exit services

This topic describes steps that should be taken when invoking VTAM exit services as well as a sample invocation. This topic also provides information that is useful if you experience an error when you use VTAM exit services.

Design considerations

The following steps should be taken when invoking VTAM exit services:

1. Check the bit in the function code and related session information to be sure VTAM exit services is available. If this bit is not on, VTAM exit services is not available and cannot be used.
2. Get the pointer to the VTAM exit services parameter list from the session management exit parameter list. The VTAM exit services parameter list pointer is at displacement 72 (decimal) into the session management exit parameter list. The session management exit parameter list is pointed to by register 1 when the session management exit is called by VTAM.
3. Check the VTAM exit services parameter list pointer to be sure it is not zero.
4. Get the pointer to the supported functions bitmap. A pointer to the supported functions bitmap is at displacement 0 into the VTAM exit services parameter list.
5. Check the supported functions bitmap to be sure the bit is on which represents the correct function.
6. Build the text for the message to be issued.
7. Build the VTAM exit services message parameter list (EXMPL).
 - a. Put a pointer to the message text into the EXMPL.
 - b. Put the length of the message text into the EXMPL.
8. Put a pointer to the EXMPL into the VTAM exit services parameter list.
9. Put the function code into the VTAM exit services parameter list.
10. Put the address of the VTAM exit services parameter list into register 1.

Note: Be sure to save the pointer to the session management exit parameter list. It might be needed by the session management exit upon return from VTAM exit services.
11. Load the pointer to the VTAM exit services module (ISTIECX5) from the VTAM exit services parameter list into register 15.
12. Call the VTAM exit services module (using BALR 14,15).
13. Check the return code from VTAM exit services. The return code is in register 15 when VTAM exit services returns to the session management exit. If the return code is nonzero, take the appropriate action.

Example invocation of VTAM exit services

The following assembler code segment shows how a session management exit that uses the above design considerations could invoke VTAM exit services to issue a message on the system console. In this example, it is assumed that register 1 points to the session management exit parameter list (passed by VTAM to the session management exit when the session management exit is called).

```
L      3,4(,1)      * REG3 = ADDR OF FUNC & RELATED SESS
TM     3(3),X'01'    * EXIT SERVICES AVAILABLE?
```



```

      BNO    NOMSG          * NO, EXIT SERVICES NOT AVAILABLE
      LR     4,1            * SAVE POINTER TO SME PARAMETER LIST
      L      1,72(,1)       * PUT EXSPL ADDRESS IN REG 1
      LTR    1,1            * MAKE SURE EXSPL IS AVAILABLE
      BZ     NOMSG          * NO, MESSAGE FUNCTION NOT AVAILABLE
      L      3,0(,1)        * REG3 = ADDRESS OF SUPP FUNC BITMAP
      TM     0(3),X'40'     * MESSAGE FUNCTION SUPPORTED?
      BNO    NOMSG          * NO, MESSAGE FUNCTION NOT AVAILABLE
      LA     3,EXMPL        * ADDRESS OF EXMPL IN REG 3
      ST     3,12(,1)       * STORE EXMPL ADDRESS IN EXSPL
      MVI    4(1),X'01'     * INDICATE MESSAGE FUNCTION DESIRED
      L      15,8(,1)       * LOAD ADDRESS OF ISTIECX5
      BALR   14,15          * BRANCH TO EXIT SERVICES
      LR     1,4            * RESTORE CONTENTS OF REG 1
      LTR    15,15          * CHECK RETURN CODE
      BNZ    ABEND          * ABEND IF NOT SUCCESSFUL
NOMSG  L      13,4(13)      * RESTORE CALLER'S SAVE AREA
      LM     14,12,12(13)   * RESTORE CALLER'S REGISTERS
      BR     14            * RESTORE CALLER'S SAVE AREA
ABEND  EX     0,ABEND       * ABEND0C3
*****
* EXMPL TO SEND AN 'AVERAGE' MESSAGE *
*****
EXMPL  DC     XL2'00'       * RESERVED (2 BYTES)
      DC     AL2(AVGE-AVGS) * LENGTH OF MSG (2 BYTES)
      DC     A(AVG)         * ADDRESS OF MSG TEXT (4 BYTES)
*****
* AN 'AVERAGE' MESSAGE *
*****
AVGS   EQU    *
AVG     DC    C'THIS IS AN EXAMPLE OF AN AVERAGE MESSAGE WHICH'
      DC    C' AN SME MIGHT ATTEMPT TO ISSUE USING THIS NEW '
      DC    C'ENHANCEMENT: VTAM EXIT SERVICES. NOTICE THAT T'
      DC    C'HE MESSAGE TEXT IS DIVIDED INTO 56 BYTE BLOCKS'
      DC    C' REGARDLESS OF WHERE BLANKS MAY APPEAR IN THE '
      DC    C'MESSAGE. NOTICE THAT THE NAME OF THE EXIT IS I'
      DC    C'NCLUDED IN MESSAGE IST1282I'
AVGE    EQU    *

```

The above session management exit code resulted in the following messages being sent to the system console by VTAM exit services:

```

JOB      2  IST1282I MESSAGE FROM ISTEXCAA IN ISTEXCAA
IST1405I  THIS IS AN EXAMPLE OF AN AVERAGE MESSAGE WHICH AN SME MI
IST1405I  GHT ATTEMPT TO ISSUE USING THIS NEW ENHANCEMENT: VTAM EX
IST1405I  IT SERVICES. NOTICE THAT THE MESSAGE TEXT IS DIVIDED INT
IST1405I  O 56 BYTE BLOCKS REGARDLESS OF WHERE BLANKS MAY APPEAR I
IST1405I  N THE MESSAGE. NOTICE ALSO THAT THE NAME OF THE EXIT IS
IST1405I  INCLUDED IN MESSAGE IST1282I
IST314I  END

```

Note: VTAM exit services executes on behalf of the session management exit. VTAM performance could be degraded if the session management exit uses VTAM exit services excessively.

Problem determination

If VTAM exit services is not displaying messages as expected, consider the following points when diagnosing the problem:

- Messages might be truncated if MSGMOD=YES is being used. For more information about MSGMOD, see [z/OS Communications Server: SNA Operation](#).
- Message text must be in uppercase to appear on the console. Lowercase is not supported.
- The text is divided into 56 character segments when placed into VTAM messages without regard to spaces or other special characters. This can cause breaks in the text, which might not be desirable (such as in the middle of words). Alignment of data would be the responsibility of the exit. No blank suppression is performed on these messages.

The following examples illustrate this point:

Example 1

A single message (56 characters or fewer):

```
07.59.07 IST1282I MESSAGE FROM ISTEXCAA IN ISTEXCAA
IST1405I   THIS MESSAGE CONTAINS EXACTLY 52 CHARACTERS OF TEXT.
IST314I   END
```

Example 2

A 267-character message:

```
07.59.07 IST1282I MESSAGE FROM ISTEXCAA IN ISTEXCAA
IST1405I   THIS IS A MULTILINE MESSAGE WHICH CONTAINS 267 CHARACTER
IST1405I   S OF TEXT PROVIDED BY THE SESSION MANAGEMENT EXIT: ISTEX
IST1405I   CAA. NOTICE THAT THE TEXT IS DIVIDED INTO 56 CHARACTER S
IST1405I   EGMENTS AND PLACED INTO IST1405I MESSAGES WITHOUT REGARD
IST1405I   TO SPACES OR ANY OTHER SPECIAL CHARACTERS.
IST314I   END
```

If this problem is not causing the display error, you must determine whether the problem is caused by the session management exit or by VTAM exit services. In order to determine whether the problem is caused by the session management exit or by VTAM exit services, the input to VTAM exit services should be examined. The following MVS SLIP command can be used to trace the input passed to VTAM exit services by the session management exit:

```
SLIP SET,IF,PVTMOD=(ISTIECXs,0),A=TRACE,
      TRDATA=(1R?,+13,1R?+C?,+7,1R?+C?+4?,+nnnn),J=NET,END
```

where:

ISTIECXs

The name of the VTAM exit services module.

NET

The job name VTAM (be sure to use the job name VTAM as it runs on your system).

Register 1

Points to the VTAM exit services parameter list (EXSPL) (X'14' bytes long).

EXSPL+C

Points to the EXMPL (X'8' bytes long).

EXMPL+4

Points to the message text passed by the session management exit.

nnnn

The length of the message text passed by the session management exit to VTAM exit services. The *nnnn* value should be specified in hexadecimal and should be one less than the actual length of the message text.

Abend situation

If an abend occurs in the session management exit or VTAM exit services, VTAM issues message IST793E and continues as though the exit had not been coded. If such an error is suspected to be in VTAM exit services, the MVS SLIP command could be used to obtain a dump of VTAM at the time of the abend. If the abend is found to be in VTAM exit services, consider the following possibilities:

- The pointer to the VTAM exit services parameter list was nonzero, but was not valid when the session management exit called VTAM exit services.
- The pointer to the EXMPL (the pointer to the input parameter list in the VTAM exit services parameter list) was nonzero, but was not valid when the session management exit called VTAM exit services.
- The pointer to the message text in the EXMPL was nonzero, but was not valid when the session management exit called VTAM exit services.
- Some portion of the message text could not be accessed by VTAM exit services (for example, the session management exit passed a message length in the EXMPL which exceeded the storage area owned by the session management exit).

Any of these conditions could cause VTAM exit services (ISTIECXS) to abend, causing deactivation of the session management exit. If an abend occurred due to one of these reasons, the session management exit should be corrected.

If an abend or any other problem occurs in VTAM exit services and you have determined that the session management exit is not at fault, obtain a dump and a VTAM Internal Trace and contact the IBM Support Center. It might be necessary to use a trap or the MVS SLIP command to collect the proper documentation. See [z/OS Communications Server: SNA Operation](#) and [z/OS Communications Server: SNA Diagnosis Vol 1, Techniques and Procedures](#) for more information about VTAM Internal Trace. See [z/OS MVS System Commands](#) for more information about the MVS DUMP and SLIP commands.

Appendix D. Sample configuration services XID exit routine

This topic provides the following information:

- A description of the process used by the sample exit to identify the switched major node name and the names of dynamic switched devices. If you write your own exit, you can modify the process.
- Guidelines for modifying the sample exit's name generation function.
- A description of the CSDATA file that the exit can write to when the connection status feature of the exit is active.

A configuration services XID exit routine allows you to:

- Provide VTAM information to create dynamic representations of switched devices and switched major nodes without disrupting a switched network.
- Record connection and disconnection times for switched PUs.
- Tell VTAM to process or deny a request for contact (REQCONT) from a switched device that is already defined to VTAM.
- Tell VTAM to process or deny a request for PU activation (REQACTPU) from a predefined switched device that is supported by DLUR.

For a complete description of the exit, see [“Configuration services XID exit routine”](#) on page 85.

Generating a dynamic switched major node name

To use the multiple dynamic switched major nodes function, the exit routine should include a procedure that will generate a new major node name for the switched resource, set the appropriate bit in the build vector, along with the other necessary information, and return the build vector to VTAM.

Consider the following information when generating a major node name:

- The procedure should determine whether a new major node name will be generated or whether an existing name will be used for the dynamic resources that are being created.

This is the part of the procedure that enables you to group several switched resources under a specific major node. You will have to code whatever algorithm is necessary to determine when to create a new major node name or use an existing one.

- If a new major node name is being generated:
 - The name must be in the range 1–8 characters
 - The first 3 characters must not be IST
 - The first character must be alphabetic or national, and the remaining characters must be alphabetic, numeric, or national.
- If the exit returns the name of a major node that already exists:
 - The major node cannot be an application major node
 - The major node must be a dynamic switched major node
 - The dynamic switched major node must be active

Note: If the exit does not return a major node name, VTAM uses ISTDWMMN as the default name.

Device identification

If the exit returns the begin vector with the dynamic build bit set, VTAM queues the XID vector to the exit whenever any of the following attempts to dial in:

- An unknown switched PU type 1, 2, 2.1 device
- An unknown switched PU type 2 or 2.1 for DLUR-supported devices
- A casually connected PU type 4 or 5 device

The exit uses the information provided in the XID vector to construct a build vector and any necessary resource entry blocks.

The exit needs a way to identify the unknown device. A device allowed to participate in dynamic builds can have its definition in either a created CPNDEF or NIDDEF definition file. If the device is identified by a CPNAME, its definition is in the CPNDEF file. If the device is identified by a NODEID (IDBLK and IDNUM), its definition is in the NIDDEF file.

Both the NIDDEF and the CPNDEF definition file names must match your exit name.

The exit searches for a valid, unique name for the unknown device. The exit checks the X'10' bit, which indicates that VTAM received from the dependent LU requester (DLUR) a PU name on the REQACTPU in a X'0E' vector and did not find a defined PU with that name. If the bit is set, the exit will use this PU name to build a resource entry block for the device. (See Table 55 on page 92 for bit and PU name offsets.) If the bit is not set, the exit then attempts to find a CPNAME field. If there is no CPNAME field, the exit attempts to find a NODEID field. If the unknown device does not have either a CPNAME field or a NODEID field, the exit invokes the name generation function.

If the exit finds a valid CPNAME field or NODEID field for the device, the exit searches for the device's definition in either the CPNDEF or NIDDEF file. If the exit does not find a definition in CPNDEF or NIDDEF, the exit invokes the name generation function.

If the exit finds the device's definition in either the CPNDEF or the NIDDEF definition file, the exit looks for a complete resource name or model name pair for the device. If the exit finds a complete resource name or model name pair, the exit builds a resource entry block for the device. The exit then puts the build vector header and flag information in front of the list of resource entry blocks and returns to VTAM. If the exit does not find a complete resource name or model name pair for the device, the exit denies the REQCONT (or REQACTPU) from the unknown device and returns to VTAM.

The CPNDEF implementation supplied with the sample exit routine ISTEXCCS can define a single PU 2.1 connection to a CP by using CPNAME. Any subsequent attempt to define a dynamic PU for a connection by using the same CPNAME will fail because the name returned to use for the dynamic PU is already in use by VTAM.

If the name generation function cannot construct a name for the device, the exit returns to VTAM. The X'80' bit in byte 0 of the build vector is set to indicate to VTAM that it is to deny the REQCONT (or REQACTPU).

Device definition files

Within a CPNDEF or NIDDEF definition file, a device's definition is grouped by consecutive records in the following order:

Record

Content

1

DEVICE ID

The DEVICE ID is either a CPNAME or a NODEID. The NODEID consists of the IDBLK and IDNUM.

2

PU NAME

PU NAME is the name you want the PU to have once the connection is established.

3

PU MODEL and a station address (the station address, ADDR, is optional).

PU MODEL is the PU label in the model major node.

4

LU NAME

This is the actual name that VTAM uses to refer to this particular LU, a unique net name.

5

LU MODEL and a LOCADDR value

The LOCADDR field is optional and can be used to override the LOCADDR value specified in the subject model.

Any other records that follow within a particular device's definition group are either LU NAME or LU MODEL pairs as described in records 4 and 5 above. This is the model name on the LU definition in the model major node.

Each record must be 80 bytes in length. Following is a description of the format for each record:

00000000011111111122222222223333333333444444444455555555556 . . . 8			
123456789012345678901234567890123456789012345678901234567890 0			
deviceid line commentary (optional)			
puname		.	
pumodel xxx		.	
luname		.	
lumodel xxx		.	
luname		.	
lumodel xxx		.	
.		.	
.		.	
.		.	

Column

1-8

A left-adjusted alphanumeric CP name (1-8 bytes in length) in the CPNDEF file or, an EBCDIC representation of the hexadecimal NODEID (8 bytes in length) in the NIDDEF file

9

Blank

10-17

A left-adjusted PU NAME, PU MODEL, LU NAME, or LU MODEL

18

Blank

19-21

If the device's definition contains a LOCADDR value, these columns contain a right-adjusted LOCADDR (0-255).

22-34

Blanks

35-80

Optional line commentary

All padding within the fields is done with blanks (X'40'). All byte positions up to and including column 34 that are not defined must contain blanks (X'40').

Figure 29 on page 270 shows an example of a network with PUs and LUs. Following the figure is a table with an example NIDDEF and CPNAME definition file for that network.

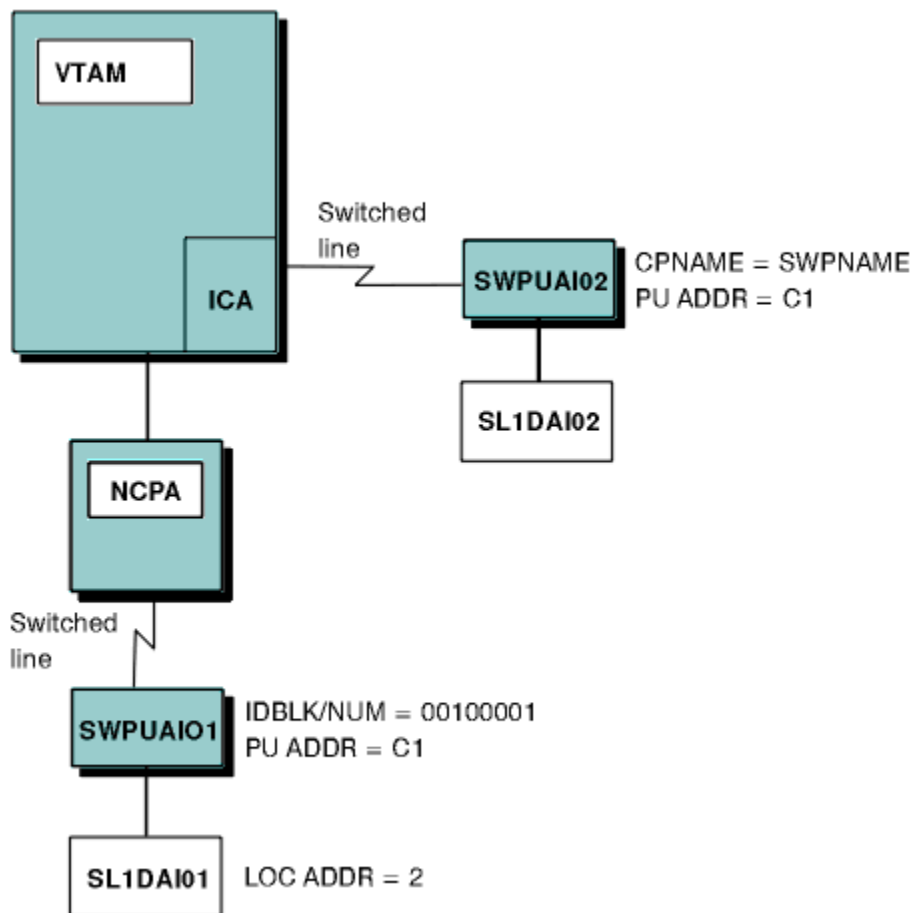


Figure 29. NIDDEF and CPNAME example network

Table 137. NIDDEF and CPNAME definition file example

Record	Content	NIDDEF	CPNAME
1	DEVICE ID	00100001	
2	PU NAME	SWPUAI01	
3	PU MODEL; there also can be an ADDR (station address)	SPUMOD01	
4	LU NAME	SLIDAI01	
5	LU MODEL; there also can be a LOCADDR	SLUMOD01 2	
4	LU NAME	SLIDAI0A	
5	LU MODEL; there also can be a LOCADDR	SLUMOD01	
1	CPNAME		SWPNAME
2	PU NAME		SWPUAI02
3	PU MODEL; there also can be an ADDR (station address)		SPUMOD01

Table 137. NIDDEF and CPNAME definition file example (continued)

Record	Content	NIDDEF	CPNAME
4	LU NAME		SLIDAI02

After a NIDDEF file is created, do a file definition so that VTAM will be able to locate the FILEDEF. See *z/OS Communications Server: SNA Network Implementation Guide* for more information about dynamically defining a switched connection in a single-domain environment.

Name generation function

In addition to allowing the specification of PU and LU names and PU and LU model names in external files, the exit provides an internal name generation function. This function constructs names for the specified device ID by concatenating data from the input XID to declared values that are internal to the exit. The name generation function is contained entirely within the ISTECCS module.

Invocation

The exit invokes the name generation function if any of the following conditions occurs:

- XID processing is unable to gain access to the contents of a CPNDEF or NIDDEF file because the file is not defined by a DDNAME JCL statement.
- XID processing successfully opens a CPNDEF or NIDDEF file, but no entry for CP name or node ID is found for the input XID.

The exit does not invoke the name generation function if any of the following conditions occurs:

- XID processing detects an I/O error while data is being retrieved from the CPNDEF or NIDDEF file.
- XID processing successfully opens the CPNDEF or NIDDEF file, an entry (CPNAME or NODEID) is found for the input XID, but the entry contents are in error or in an incorrect format.

Name generation table

The declares in the exit used to generate names are grouped into a name generation table. If necessary, you can change the table entries to meet the needs of your installation. (See “[Tailoring the name generation table](#)” on page 273 for more information.) Because the entries in the name generation table are of variable length and each entry contains the address of the next entry, you can think of the table as a chained list of table entries.

Operation

When the exit invokes the name generation function, the exit extracts the block ID from the XID passed by VTAM. The exit searches the name generation table entries for an entry that corresponds to the block ID.

If the exit does not locate an entry in the table that corresponds to the XID, the exit returns control to VTAM with the X'80' bit in byte 0 of the build vector set to tell VTAM to deny the REQCONT (or the REQACTPU).

If the exit does locate an entry in the table that corresponds to the XID, the name generation function extracts data from the entry and uses it to construct names in the resource entry blocks in the build vector. When the exit returns control to VTAM, the X'80' bit in byte 0 of the build vector is reset.

Name generation table entry format

Each entry in the name generation table must contain the following required fields:

- Pointer to the control block for the next entry in the table

Chaining the entries allows for variable length control blocks and minimizes storage requirements.

- ID code

This code is the block ID that the entry represents.

If the ID code is nonzero, the following fields are also required for the table entry:

- PU station address, or 0
- Name prefix character
- PU model name

The extension bit must be turned on in the BEGIN vector, or VTAM will use the old length. In addition, the following optional overrides for Model Major Node definition of a PU can be specified and requested:

- MAXDATA
- MAXOUT

If the block ID is 0, the entry is considered not valid and is not used. If the block ID is nonzero, it is assumed that the remainder of the entry contains valid data. An LU subentry must contain the following required fields:

- LU model name
- LU local address
- Subentry duplication factor

The extension bit must be turned on in the BEGIN vector, or VTAM will use the old length. In addition, the following optional overrides for Model Major Node definition of an LU subentry can be specified and requested:

- Default LOGMODE name
- LOGTAB—Interpret table
- MODE table
- LOGAPPL value
- USS table

Although the overrides might not be requested, the storage for the optional fields is still required.

The subentry duplication factor is the number of subentries (LUs) to be defined under the PU.

PU name construction

Following is a description of the format of the PU name constructed in a resource entry block of the build vector:

CNNNNNSS

Character

C

The *name prefix character* as specified in the name generation table entry.

NNNNN

The EBCDIC representation of the block NUM extracted from the input XID.

SS

The station address. If the station address is 0, SS is two blanks.

LU name construction

Following is a description of the format of the LU names constructed in the build vector:

CNNNNNLL

Character

C

The *name prefix character* as specified in the name generation table entry.

NNNNN

The EBCDIC representation of the block NUM extracted from the input XID.

LL

The EBCDIC representation of the local address.

Tailoring the name generation table

Modification of the sample exit's name generation table might be necessary to meet some unique need of your installation. You can change any or all of the entries. You can tailor the name generation table entries as follows:

- You can deactivate any entry so that the name generation function does not use the entry's data.
- You can modify the entry so that alternate characters are used to construct PU and LU names.
- You can add entries to the table.

Consider the following restrictions if you modify the name generation table:

- If you modify the table, reassemble the ISTECCS module and replace the current object file.
- Erroneous use of the duplication factor in the LU subentry can cause wrapping of the local address. That is, X'FF' wraps to X'00'.
- There are limits on the number of LU names the table can generate. These limits are not checked by the exit unless the number of LU resource entry blocks would cause overflow of the build vector.
- Make sure that your PU station address, if used, will not also be specified as an LU local address.
- Incorrect modification of labels and pointers can result in assembly or logic errors.
- It is advisable that you maintain the alphabetical order of the prefix characters.
- If the same block ID is coded in more than one entry in the table, the exit uses the first entry it finds with the block ID and ignores the remaining entries in the table that have that same block ID.

Deactivating an entry

If the ID code in the name generation table entry is 0, the entry is ignored. If you change the ID code from nonzero to 0 to deactivate the entry, you should also comment out any LU subentries to preserve storage. Deactivation of an entry by changing the pointer in the previous entry is possible, but not advisable.

Modifying an entry

You can modify an entry in the ISTECCS module by substituting your data in place of the sample data. Pay close attention to the format of the data fields and the alignment of the LU subentries. You should not alter the assembler declare statements except to change the characters and numbers within the quotation marks or parentheses.

Adding an entry

You can add an entry to the name generation table by following these steps:

1. Select a unique name prefix character.
2. Select a position within the chain for your new entry. It is advisable that you select the position of the new entry by alphabetizing the prefix characters.
3. Duplicate the preceding entry and change the label on the new entry to NGENT α , where α is your new prefix character.
4. Change the pointer in the preceding entry to your label name.
5. Modify the entry with your data.

Connection status records

The configuration services XID routine can monitor connection and disconnection times of switched devices. When the connection status feature has been activated, the exit writes to the CSDATA file. This file contains 80-byte records that are written directly from the connection status vector that VTAM passes to the exit. The connection status feature writes a record to the CSDATA file when a resource, that was processed by the dynamic build feature, connects or disconnects from the network. A record includes fields that identify:

- Whether the resource is connecting or disconnecting
- The name of the line to which the resource is connecting or was connected, or, for DLUS-supported resources, the name of the DLUR
- The name of the PU
- A timestamp, in time-of-day format, that indicates when the connection or disconnection occurred
- The PU's station ID and CPNAME

You can use records to determine which PU gained access to the network, over which line, and at what time. You can also use records to determine how long a PU was connected to the network. You might want to write a tool to extract data from the file and create a summary report of the activity of resources that are dynamically connecting to and disconnecting from the network.

The format of a record in the CSDATA file is identical to the format of the connection status vector defined in the ISTEXCCS module. A record is padded on the right with enough blanks (X'40') to complete an 80-byte record.

Appendix E. Command verification exit routine

The example that follows is intended as an educational tool. It shows how a command verification exit routine can be used to control VTAM functions. The code in this example is only part of a complete exit routine and will not work as presented here.

You can use a command verification exit to screen commands that affect critical nodes in your network. If you use the exit for this function, include a list of critical nodes in the exit. The following sample code illustrates how the exit routine can be coded to perform the screening function. The sample code also illustrates how you can create the list of critical nodes.

In this sample, if the node's ID is in the list of critical nodes, but the OVERRIDE=YES option is not specified on the command, the exit does not modify the command.

Sample command verification exit routine

```
ISTCMMND CSECT
    USING *,15
< standard linkage >
* Determine if this invocation is to activate the exit - Reg 0 = X'10'
    C    0,=X'00000010'
    BE    ACTIVE
* Determine if this invocation is to deactivate the exit - Reg 0 = X'20'
    C    0,=X'00000020'
    BE    INACT
* Otherwise this must be a command screening code
* Search for protected ID and OVERRIDE
    L    8,20(1)          Get address of command string
    L    7,16(1)          Get address of length
    LH   7,0(7)           Get command length
    AR    7,8             R7 has last address to check
    CLC   0(10,8),=C'VARY INACT' Is this a VARY INACT
    BNE    FINAL          Do no further checking
    A     8,=F'10'        Increment past VARY INACT
    SR    0,0             Clear register 0
* Locate ID parameter to check node or OVERRIDE keyword
KEYCHECK
    CLC   0(3,8),=C'ID='    Is this ID=
    BE    IDFOUND          ID parameter found
    CLC   0(9,8),=C'OVERRIDE=' Is this OVERRIDE=
    BE    OVERRIDE         OVERRIDE parameter found
    .
* Check ID value to determine if this is a critical node
IDFOUND
    CLC   0(8,8),=C'CRITICAL' Is this node CRITICAL
    BE    MATCH            Get current command length
    CLC   0(5,8),=C'CRITB'    Is this node CRITB
    BE    MATCH            Get current command length
    .
* Indicate match found for ID, if OVERRIDE is not found
* the command will fail
MATCH
    B     KEYCHECK          Go back to check for OVERRIDE
* Search for OVERRIDE option - it is assumed to be at the end of the
string
OVERRIDE
    A     8,=F'9'           Bump past override
    CLC   0(3,8),=C'YES'     YES OVERRIDE
    BE    YES               Override found
    CLC   0(2,8),=C'NO'      NO OVERRIDE
    BE    NO                Get current command length
* Override not found - if ID is in list of critical nodes, set
* command verification code to fail command
IDFOUND
    L     2,12(1)           Get address of exit code
    LA    9,0               Use value 0 to set return code
    ST    9,0(2)            Set command verification code to 0
    .
*
    B     FINAL
* Remove OVERRIDE=xxx specification from command string
REMOVE
    .
* Indicate OVERRIDE parameter option
```

```

YES      L      2,12(1)      Get address of exit code
        LA      9,0          Use value 0 to set return code
        ST      9,0(2)      Set command verification code to 0
        *                               to allow command to continue
        B      FINAL
NO       L      2,12(1)      Get address of exit code
        LA      9,4          Use value 0 to set verification code
        ST      9,0(2)      Set command verification code to 4
        *                               to fail command (IST1201I issued)
        B      FINAL
* Do any processing needed at activation
ACTIVE   .
        B      FINAL
* Do any processing needed at inactivation
INACT    .
        B      FINAL
* If a severe error occurs and the invocation of the exit should
* be terminated, set register 15 to X'80' to deactivate the exit
SEVERE   LA      15,X'80'
        .
        B      FINAL
* Complete return linkage and return to caller
FINAL    .
        < standard linkage >
        BR      14
DATAD    DSECT
        DS      0F
        DS      18F
BLANK    DC      C' '
BASESAVE DS      F
RETCODE  DS      F
ENDDATAD EQU    *
END      ITCMMND

```

Appendix F. Sample USERVAR exit routine for TPF sessions

IBM supplies a functional sample USERVAR exit routine under the VTAM module, ISTEXCUV, in SYS1.SAMPLIB. The sample exit is specific to the Transaction Processing Facility (TPF) environment. This topic contains sections of code from the sample exit routine.

IBM supplies both source code and object code for this routine. You can use the sample USERVAR exit routine as it is coded, modify the sample, or write your own exit routine. You can obtain a complete current listing of the sample exit routine by printing the file. If you write your own USERVAR exit routine, you should use the interface information described in [“USERVAR exit routine”](#) on page 109. See [“USERVAR exit routine for TPF sessions”](#) on page 180 for a discussion of the function of the sample USERVAR exit.

The sample USERVAR exit allows session users to request TPF services by using predefined generic names for the TPF application programs. VTAM calls this exit for exit activation, exit deactivation, when a USERVAR needs to be added, updated, or deleted, or when there is a request to find the best control logical unit (CLU) for a session.

The logon manager defines its USERVARs as volatile. The sample USERVAR exit translates only volatile USERVARs so that the logon manager can ensure proper load balancing and maintain CLU session limits. When the sample exit receives a translation request for a static or dynamic USERVAR, the exit turns off the translation flag, sets a nonzero return code, and returns control to VTAM.

Function code X'04' processing

When the sample exit receives a request to translate a USERVAR, the exit processes the request. The exit must first check the OLU status because translation processing when the OLU is the SLU differs from when the OLU is the PLU.

If the OLU is the SLU for the session, and the USERVAR is volatile, the sample USERVAR exit does not translate the name. The exit instead sets the value of the USERVAR equal to the generic USERVAR name. Although the name was not translated, the exit turns on the translation flag. The translation flag must be turned on to ensure that VTAM drives the logon manager through the logon exit.

If the OLU is the SLU for the session, and the USERVAR is dynamic or static, the exit turns off the translation flag and sets a nonzero return code. VTAM does not establish the session.

The following code performs the USERVAR exit's portion of this processing.

```
* /*****
* /* This segment processes request for USERVAR translation when
* /* UVEFLAG1 indicates SLU STATUS. No true translation occurs, but
* /* if the USERVAR is VOLATILE then VALNAME is set equal to UVEGAPNM
* /* and the translate bit is set on to cause Logon Manager to be
* /* driven. If USERVAR is not VOLATILE the translate bit is turned
* /* off and a bad type return code is set.
* /*
* /* If STATIC or DYNAMIC
* /* Turn off translate bit
* /* Set RC=X'25'
* /* Free local storage then exit
* /*
* /* If VOLATILE
* /* Turn on translate bit
* /* Update USERVAR value field in USERVAR EXIT parameter list with
* /* generic TPF application name
* /* Set RC=X'0'
* /* Free local storage then exit
* /*
* /*****
*
INVKSLU DS 0H ROUTINE FOR FINDING CLU REQUEST
* WHEN OLU IS SLU
TM UVEFLAG1,X'08' CHECK IF STATIC USERVAR @VDA
```

	BO	DYNLSLU	IF NOT STATIC, CONTINUE	@VDA
	NI	UVEFLAG1,X'DF'	TURN OFF TRANSLATE BIT	@VDA
	L	R15,FULL25	SET BAD TYPE CODE FOR R15	@VDA
	B	ENDFREEA	IF STATIC, EXIT MODULE	@VDA
DYNLSLU	TM	UVEFLAG1,X'10'	CHECK IF DYNAMIC USERVAR	@VDA
	BO	VOLSLU	IF NOT DYNAMIC, CONTINUE	@VDA
	NI	UVEFLAG1,X'DF'	TURN OFF TRANSLATE BIT	@VDA
	L	R15,FULL25	SET BAD TYPE CODE FOR R15	@VDA
	B	ENDFREEA	IF DYNAMIC, EXIT MODULE	@VDA
VOLSLU	OI	UVEFLAG1,X'20'	TURN ON TRANSLATION BIT	@VCC
	MVC	UVEVALNM,UVEGAPNM	SET NAMES EQUAL	@V5C
	L	R15,NULL	SET GOOD CODE FOR R15	
	B	ENDFREEA	TO EXIT THIS MODULE	
*				

If the OLU is the PLU for the session, and the USERVAR is volatile, the sample USERVAR exit processes the translation. All USERVAR translations occur in the logon manager host. The exit queries the logon manager data base by issuing the REQTAIL macro for function code X'04' processing. When called with function code X'04', the REQTAIL macro transfers control to the logon manager. The logon manager invokes the CLU search exit routine to compare all contending CLUs. After the CLU search exit has compared all contending CLUs, it identifies the best CLU for the session. The CLU search exit returns the name of the best CLU to the logon manager. The logon manager finds the application program LU name associated with the best CLU and returns that name to the REQTAIL macro. The REQTAIL macro returns the application program LU name to the USERVAR exit in the REQTAIL macro parameter list. The REQTAIL macro also returns a completion code indicating the search was successful.

After the successful return from the REQTAIL macro, the USERVAR exit:

- Replaces the value it received from VTAM in the USERVAR parameter list with the name returned by the REQTAIL macro (the USERVAR table is not updated)
- Turns on the translation flag
- Sets the return code to 0
- Returns control to VTAM

If the OLU is the PLU for the session, and the USERVAR is dynamic or static, the exit turns off the translation flag and sets a nonzero return code. VTAM does not establish the session.

The following code performs the USERVAR exit's portion of this processing.

```
* /*****
* /* For VOLATILE USERVARs only this segment initializes the REQTAIL *
* /* parameter list from the input parameter list and issues 'REQTAIL*
* /* SEARCH'. If return code is zero, sends best CLU name to caller *
* /* and set best-CLU-found bit on; else, sends error return code. *
* /* Storage is allocated for register save area (72 bytes) and for *
* /* the REQTAIL parameter list (72 bytes). *
* /* *
* /* If STATIC or DYNAMIC *
* /* Turn off translate bit *
* /* Set RC=X'25' *
* /* Free local storage then exit *
* /* *
* /* If VOLATILE *
* /* Allocate storage for REQTAIL parameter list (144 bytes) *
* /* IF storage allocation failed *
* /* Set RC=X'20' *
* /* Free local storage then exit *
* /* ELSE *
* /* Issue REQTAIL macro to search the best CLU *
* /* IF REQTAIL-SEARCH failed *
* /* RC=X'26' when Logon Manager was not available *
* /* RC=X'27' when CLU not found *
* /* RC=X'30' when Cross memory initialization failed *
* /* RC=X'34' when APLB not found *
* /* RC=X'38' when Subarea address not found *
* /* RC=X'46' when Logon manager exit not initialized *
* /* Free storage allocated for REQTAIL parameter list *
* /* Free local storage then exit *
* /* ELSE *
* /* Turn on best-CLU-found bit *
* /* Update USERVAR value field in USERVAR EXIT parameter *
* /* list with target application LU name *
* /* Free storage allocated for REQTAIL parameter list *
```



```

* /*          Set RC=X'0'          *
* /*          Free local storage then exit          @VBC*
* /*****
*
INVKPLU  DS      0H                      ROUTINE FOR FINDING CLU REQUEST
*
      TM      UVEFLAG1,X'08'          CHECK IF STATIC USERVAR          @VDA
      BO      DYNPLU                  IF NOT STATIC, CONTINUE          @VDA
      NI      UVEFLAG1,X'DF'          TURN OFF TRANSLATE BIT          @VDA
      L      R15,FULL25                SET BAD TYPE CODE FOR R15          @VDA
      B      ENDFREEA                  IF STATIC, EXIT MODULE          @VDA
DYNPLU   TM      UVEFLAG1,X'10'          CHECK IF DYNAMIC USERVAR          @VDA
      BO      VOLPLU                  IF NOT DYNAMIC, CONTINUE          @VDA
      NI      UVEFLAG1,X'DF'          TURN OFF TRANSLATE BIT          @VDA
      L      R15,FULL25                SET BAD TYPE CODE FOR R15          @VDA
      B      ENDFREEA                  IF DYNAMIC, EXIT MODULE          @VDA
VOLPLU   L      R0,FULL144              LENGTH OF NEEDED MEMORY (72 FOR
*                                     REGS/72 FOR ELMCSPPL) @V5C @V9M
      STORAGE OBTAIN,COND=YES,LENGTH=(0) STORAGE REQTAIL PARMLIST
*
      LTR     R15,R15                  TEST SUCCESS OF STORAGE          @V9M
      BNZ     FAILGET2                STORAGE FAILED                    @V9M
      NI      UVEFLAG1,X'DF'          INITIALIZE BEST-CLU-FOUND BIT
*                                     TO ZERO                          @VAC
      L      R8,R15,FULL144           LENGTH OF ACQUIRED AREA          @V5C
      BCTR    R8,R0                    MAKE LENGTH CORRECT
      EX      R8,MVCLEAR              INITIALIZE AREA TO ZERO
      LR      R4,R1                    ADDRESS FROM STORAGE
      A      R4,FULL72                PAST REGISTER SAVE AREA
      USING   ELMCSPPL,R4             MAKE ADDRESSABLE BY NAMES
      LR      R13,R1                  ADDRESS OF REGISTER SAVE AREA
      MVC     CSPSPTNM(8),UVEOLUNM    SESSION PARTNER NAME          @V5C
      MVC     CSPSPTSA(4),UVEOLUSA    SESSION PARTNER SUBAREA        @V5A
      MVC     CSPCLUNM(8),NULL        THE REQUESTING CLU NAME          @V5A
      MVC     CSPSPNPT(4),UVESPNPT    NAME LIST POINTER              @V5A
      MVC     CSPTARLU(8),NULL        TARGET APPLICATION LU           @V5A
      MVC     CSPCOSNM(8),UVECOSNM    COS NAME                      @V5A
      MVC     CSPGAPNM(8),UEVGAPNM    GENERIC TPF APPLICATION        @V5A
      MVC     CSPFLAG1(1),UVEFLAG1    FLAG                          @V5A
      MVC     CSPRUPTTR(4),NULL        REQTAIL POINTER              @V5A
      MVC     CSPCLUPT(4),NULL        THE BEST CLU POINTER           @V5A
      OI      CSPFLAG1,X'10'          SESSION THROUGH USERVAR        @V7A
      OI      CSPFCODE+3,X'04'        INDICATE SEARCH
      L      R5,IEFUFELD              ADDRESS OF USER FIELD IN XCB
      L      R5,0(R5)                ADDRESS OF EXCB TO R5
      A      R5,FULL24                TO EXTENSION AREA
      ST      R5,CSPECBXP             STORE ADDRESS OF EXCB EXTENSION
      BAL     R14,CALLREQ              ISSUE REQTAIL MACRO
      LTR     R15,R15                  SEE IF REQTAIL SUCCEEDED
      BNZ     BADFLAG3                REQTAIL RETURN NOT GOOD
      OI      UVEFLAG1,X'20'          TURN ON NAME TRANSLATION BIT
      MVC     UVEVALNM,CSPTARLU       APPLICATION LU FROM SEARCH      @V5C
      L      R0,FULL144              LENGTH OF NEEDED MEMORY (72 FOR
*                                     REGS + 72 FOR ELMCSPPL) @V5C
      STORAGE RELEASE,COND=YES,LENGTH=(0),ADDR=(1) FREE STORAGE
      L      R15,NULL                  SET GOOD CODE FOR R15
      B      ENDFREEA                  TO EXIT THIS MODULE
*
CALLREQ  ST      R14,SAVER14           FOR RETURN                      @V9M
      REQTAIL
      L      R14,SAVER14             ISSUE REQTAIL MACRO            @V9M
      BR     R14                     RESTORE THE ADDRESS              @V9M
*                                     RETURN                          @V9M
*
FAILGET2 L      R15,FULL20            CODE FOR STORAGE FAIL @V1C @V9M
      B      ENDFREEA                EXIT FROM MODULE                @V9M
*
BADFLAG3 L      R0,FULL144            LENGTH OF NEEDED MEMORY (72 FOR
*                                     REGS + 72 FOR ELMCSPPL) @V5C
      LR     R11,R15                  SAVE RETURN CODE                @VAA
      STORAGE RELEASE,COND=YES,LENGTH=(0),ADDR=(1) FREE STORAGE
      LR     R15,R11                  RESTORE RETURN CODE IN R15      @VAC
      B      ENDFREEA                TO EXIT THIS MODULE
*
BADENTRY L      R15,FULL3C            INVALID ENTRY CODE
*                                     SENDS ERROR CODE IN R15        @V1C
      B      ENDFREEA                TO EXIT THIS MODULE

```

Return codes

If you code your own USERVAR exit, there is a range of acceptable return codes. These ranges are discussed in “Final register contents” on [page 111](#). If you use the sample USERVAR exit, there are specific return codes that the exit might return to VTAM on output. Those return codes are:

X'00'

Processing completed successfully

X'20'

Failure to obtain storage (local storage, exit control block, or the exit parameter list)

X'25'

USERVAR type is not volatile (The logon manager sets this return code. The USERVAR exit only returns it to VTAM.)

X'26'

Logon manager is not available (The logon manager sets this return code. The USERVAR exit only returns it to VTAM.)

This return code indicates that the error occurred during logon manager initialization. You should check the console for error messages resulting from starting the logon manager.

X'27'

CLU was not found (The logon manager sets this return code. The USERVAR exit returns it only to VTAM.)

This return code can indicate that no CLU was found that meets the minimum requirements, as shown in “Criteria for contending CLUs” on [page 177](#).

Activate or reactivate a CLU that meets the minimum requirements. This return code can also indicate that there is an error in the REQTAIL macro parameter list, which is described in [Table 122 on page 189](#). Review your parameter list to ensure that it is coded correctly.

X'30'

Cross-memory initialization failed (The logon manager sets this return code. The USERVAR exit only returns it to VTAM.)

This return code indicates that VTAM has been denied access to logon manager address space. See [z/OS Communications Server: SNA Network Implementation Guide](#) for information about errors in your logon manager definition.

X'34'

APLB not found (The logon manager sets this return code. The USERVAR exit only returns it to VTAM.)

This return code indicates that the application program is not known to the logon manager or is not currently active. You should activate or reactivate the application program for the logon manager.

X'38'

Subarea address not found (The logon manager sets this return code. The USERVAR exit only returns it to VTAM.)

X'3A'

Unrecognized invoke flag

X'3B'

OLU status is not valid

X'3C'

Entry code is not valid or the IEFUFELD address is 0.

X'46'

Logon manager exit not initialized (The logon manager sets this return code. The USERVAR exit only returns it to VTAM.)

If the return code is greater than X'00', the session request will fail. If the return code is greater than X'80', the session request will fail, the USERVAR is deleted, and the USERVAR exit is deactivated.

Appendix G. Sample CLU search exit routine for TPF sessions

IBM supplies a functional sample control logical unit (CLU) search exit routine under the logon manager module name ELMCLUEX in SYS1.SAMPLIB. The sample exit is specific to the Transaction Processing Facility (TPF) environment. This topic contains sections of code from the sample exit routine.

IBM supplies both source code and object code for this routine. You can use the sample CLU search exit routine as it is coded, modify the sample, or write your own exit routine. Although this exit module is replaceable, the logon manager cannot function without it. You can obtain a complete current listing of the sample exit routine by printing the file. If you write your own CLU search exit routine, you should use the interface information described in [“CLU search exit routine” on page 181](#). Refer there also for a discussion of the function of the sample CLU search exit.

At the completion of logon manager initialization, the logon manager calls the sample CLU search exit with function code X'01' for exit initialization. After the exit is initialized, the logon manager can invoke the exit with one of the other function codes defined for the exit. The function the exit performs depends on the function code. The following is a summary of the exit function codes:

X'01'

Exit initialization

X'02'

A CLU session is being initiated and the CLU is available

X'03'

An application program has become available

X'04'

Two CLUs are to be compared

X'05'

An application program is no longer available

X'06'

A CLU session is being terminated and the CLU is no longer available

X'07'

Exit termination

Function code X'04' processing

The logon manager invokes the CLU search exit with function code X'04' when the exit is to compare contending CLUs and choose the best CLU to service the session. See [“Criteria for contending CLUs” on page 177](#) for a discussion of the selection algorithm the sample exit uses to choose the best CLU.

If there is only one contending CLU, the exit is called with information about only that CLU. If there is more than one contending CLU, the exit compares them a pair at a time. In each iteration, the exit selects the better of the two CLUs to be the current best CLU. In the first iteration, the current best CLU is the first contending CLU the logon manager encounters. In each subsequent iteration, the exit compares the current best CLU from the previous iteration to another contending CLU. The exit continues to compare pairs of CLUs until there are no more contending CLUs. At that point, the exit returns the name of the better CLU from the final iteration to the logon manager as the best CLU. If no best CLU is found after the list of possible CLUs is exhausted, the session request fails.

The following code performs this processing:

COMPARE4 DS	0H	PROCESS FUNCTION CODE 4
*		COMPARING CLUS
CLC	CSECLUNM,CHARZERO	IS THERE A CONTENDER?
BE	FAILCLU1	NO - RETURN TO CALLER
CLI	CSECLUNM,X'00'	IS THERE A CONTENDER?

	BE	FAILCLU1	NO - RETURN TO CALLER	
	CLC	CSECLUNM,BLANKS	IS THERE A CONTENDER?	
	BE	FAILCLU1	NO - RETURN TO CALLER	
	CLC	CSE2NAME,CHARZERO	CURRENT BEST CLU EXISTS?	
	BE	FAILCLU2	NO - CONTENDER WINS	
	CLI	CSE2NAME,X'00'	CURRENT BEST CLU EXISTS?	
	BE	FAILCLU2	NO - CONTENDER WINS	
	CLC	CSE2NAME,BLANKS	CURRENT BEST CLU EXISTS?	
	BE	FAILCLU2	NO - CONTENDER WINS	
COMPARE2	CLC	CSECLUHC,CSE2HCNT	COMPARE HOP COUNTS	@V2C
	BL	FAILCLU2	CLU2 HAS HIGHER HOP COUNT	
	BH	FAILCLU1	CLU1 HAS HIGHER HOP COUNT	
	CLC	CSECLUDS,CSE2DUPS	COMPARE DUPLICATE SESSION	@V6A
	BL	FAILCLU2	CLU2 DUP SESSIONS HIGHER	@V6A
	BH	FAILCLU1	CLU1 DUP SESSIONS HIGHER	@V6A
	CLC	CSECLUAS,CSE2SCNT	COMPARE SESSION COUNTS	@V6A
	BH	FAILCLU1	CLU1 HAS MORE SESSIONS	@V6A
FAILCLU2	DS	0H		
	MVC	CSE2AREA,CSE1AREA	MOVE CLU1 TO WINNER'S SPACE	
FAILCLU1	DS	0H	CLU2 WINS	
	B	NULLRTCD	RETURN TO CALLER	

Sample CLU search exit routine for TPF sessions

If you code your own CLU search exit, there is a range of acceptable return codes. These ranges are discussed in [“Return codes” on page 182](#). If you use the sample CLU search exit, there are specific return codes that the exit might return to VTAM on output. Those return codes are:

X'00'

Processing completed successfully

X'20'

Failure to obtain storage

X'3C'

Function code is not valid

If the return code is greater than X'00', the session request will fail. If the return code is greater than X'80', the session request will fail and the CLU search exit is deactivated.

Architectural specifications

This appendix lists documents that provide architectural specifications for the SNA Protocol.

The APPN Implementers' Workshop (AIW) architecture documentation includes the following architectural specifications for SNA APPN and HPR:

- APPN Architecture Reference (SG30-3422-04)
- APPN Branch Extender Architecture Reference Version 1.1
- APPN Dependent LU Requester Architecture Reference Version 1.5
- APPN Extended Border Node Architecture Reference Version 1.0
- APPN High Performance Routing Architecture Reference Version 4.0
- SNA Formats (GA27-3136-20)
- SNA Technical Overview (GC30-3073-04)

The following RFC also contains SNA architectural specifications:

- RFC 2353 *APPN/HPR in IP Networks APPN Implementers' Workshop Closed Pages Document*

RFCs are available at <http://www.rfc-editor.org/rfc.html>.

Accessibility

Accessible publications for this product are offered through [IBM Documentation for z/OS](#).

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 the 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 and IBM's Online Privacy Statement at 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.

Policy for unsupported hardware

Various z/OS elements, such as DFSMS, JES2, JES3, 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 publication documents intended Programming Interfaces that allow the customer to write programs to obtain the services of z/OS Communications Server.

Policy for unsupported hardware

Various z/OS elements, such as DFSMS, HCD, JES2, JES3, 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.

Trademarks

IBM, the IBM logo, and 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](http://www.ibm.com/legal/copytrade.shtml) at www.ibm.com/legal/copytrade.shtml.

Bibliography

This bibliography contains descriptions of the documents in the z/OS Communications Server library.

z/OS Communications Server documentation is available online at the z/OS Internet Library web page at <http://www.ibm.com/systems/z/os/zos/library/bkserv/>.

z/OS Communications Server library updates

Updates to documents are also available on RETAIN and in information APARs (info APARs). Go to <https://www.ibm.com/mysupport> to view information APARs.

- [z/OS Communications Server V2R1 New Function APAR Summary](#)
- [z/OS Communications Server V2R2 New Function APAR Summary](#)
- [z/OS Communications Server V2R3 New Function APAR Summary](#)
- [z/OS Communications Server V2R4 New Function APAR Summary](#)

z/OS Communications Server information

z/OS Communications Server product information is grouped by task in the following tables.

Planning

Title	Number	Description
z/OS Communications Server: New Function Summary	GC27-3664	This document is intended to help you plan for new IP or SNA functions, whether you are migrating from a previous version or installing z/OS for the first time. It summarizes what is new in the release and identifies the suggested and required modifications needed to use the enhanced functions.
z/OS Communications Server: IPv6 Network and Appl Design Guide	SC27-3663	This document is a high-level introduction to IPv6. It describes concepts of z/OS Communications Server's support of IPv6, coexistence with IPv4, and migration issues.

Resource definition, configuration, and tuning

Title	Number	Description
z/OS Communications Server: IP Configuration Guide	SC27-3650	This document describes the major concepts involved in understanding and configuring an IP network. Familiarity with the z/OS operating system, IP protocols, z/OS UNIX System Services, and IBM Time Sharing Option (TSO) is recommended. Use this document with the z/OS Communications Server: IP Configuration Reference .

Title	Number	Description
z/OS Communications Server: IP Configuration Reference	SC27-3651	This document presents information for people who want to administer and maintain IP. Use this document with the z/OS Communications Server: IP Configuration Guide . The information in this document includes: <ul style="list-style-type: none"> • TCP/IP configuration data sets • Configuration statements • Translation tables • Protocol number and port assignments
z/OS Communications Server: SNA Network Implementation Guide	SC27-3672	This document presents the major concepts involved in implementing an SNA network. Use this document with the z/OS Communications Server: SNA Resource Definition Reference .
z/OS Communications Server: SNA Resource Definition Reference	SC27-3675	This document describes each SNA definition statement, start option, and macroinstruction for user tables. It also describes NCP definition statements that affect SNA. Use this document with the z/OS Communications Server: SNA Network Implementation Guide .
z/OS Communications Server: SNA Resource Definition Samples	SC27-3676	This document contains sample definitions to help you implement SNA functions in your networks, and includes sample major node definitions.
z/OS Communications Server: IP Network Print Facility	SC27-3658	This document is for systems programmers and network administrators who need to prepare their network to route SNA, JES2, or JES3 printer output to remote printers using TCP/IP Services.

Operation

Title	Number	Description
z/OS Communications Server: IP User's Guide and Commands	SC27-3662	This document describes how to use TCP/IP applications. It contains requests with which a user can log on to a remote host using Telnet, transfer data sets using FTP, send electronic mail, print on remote printers, and authenticate network users.
z/OS Communications Server: IP System Administrator's Commands	SC27-3661	This document describes the functions and commands helpful in configuring or monitoring your system. It contains system administrator's commands, such as TSO NETSTAT, PING, TRACERTE and their UNIX counterparts. It also includes TSO and MVS commands commonly used during the IP configuration process.
z/OS Communications Server: SNA Operation	SC27-3673	This document serves as a reference for programmers and operators requiring detailed information about specific operator commands.
z/OS Communications Server: Quick Reference	SC27-3665	This document contains essential information about SNA and IP commands.

Customization

Title	Number	Description
z/OS Communications Server: SNA Customization	SC27-3666	<p>This document enables you to customize SNA, and includes the following information:</p> <ul style="list-style-type: none"> • Communication network management (CNM) routing table • Logon-interpret routine requirements • Logon manager installation-wide exit routine for the CLU search exit • TSO/SNA installation-wide exit routines • SNA installation-wide exit routines

Writing application programs

Title	Number	Description
z/OS Communications Server: IP Sockets Application Programming Interface Guide and Reference	SC27-3660	This document describes the syntax and semantics of program source code necessary to write your own application programming interface (API) into TCP/IP. You can use this interface as the communication base for writing your own client or server application. You can also use this document to adapt your existing applications to communicate with each other using sockets over TCP/IP.
z/OS Communications Server: IP CICS Sockets Guide	SC27-3649	This document is for programmers who want to set up, write application programs for, and diagnose problems with the socket interface for CICS using z/OS TCP/IP.
z/OS Communications Server: IP IMS Sockets Guide	SC27-3653	This document is for programmers who want application programs that use the IMS TCP/IP application development services provided by the TCP/IP Services of IBM.
z/OS Communications Server: IP Programmer's Guide and Reference	SC27-3659	This document describes the syntax and semantics of a set of high-level application functions that you can use to program your own applications in a TCP/IP environment. These functions provide support for application facilities, such as user authentication, distributed databases, distributed processing, network management, and device sharing. Familiarity with the z/OS operating system, TCP/IP protocols, and IBM Time Sharing Option (TSO) is recommended.
z/OS Communications Server: SNA Programming	SC27-3674	This document describes how to use SNA macroinstructions to send data to and receive data from (1) a terminal in either the same or a different domain, or (2) another application program in either the same or a different domain.
z/OS Communications Server: SNA Programmer's LU 6.2 Guide	SC27-3669	This document describes how to use the SNA LU 6.2 application programming interface for host application programs. This document applies to programs that use only LU 6.2 sessions or that use LU 6.2 sessions along with other session types. (Only LU 6.2 sessions are covered in this document.)
z/OS Communications Server: SNA Programmer's LU 6.2 Reference	SC27-3670	This document provides reference material for the SNA LU 6.2 programming interface for host application programs.

Title	Number	Description
z/OS Communications Server: CSM Guide	SC27-3647	This document describes how applications use the communications storage manager.

Diagnosis

Title	Number	Description
z/OS Communications Server: IP Diagnosis Guide	GC27-3652	This document explains how to diagnose TCP/IP problems and how to determine whether a specific problem is in the TCP/IP product code. It explains how to gather information for and describe problems to the IBM Software Support Center.
z/OS Communications Server: ACF/TAP Trace Analysis Handbook	GC27-3645	This document explains how to gather the trace data that is collected and stored in the host processor. It also explains how to use the Advanced Communications Function/Trace Analysis Program (ACF/TAP) service aid to produce reports for analyzing the trace data information.
z/OS Communications Server: SNA Diagnosis Vol 1, Techniques and Procedures and z/OS Communications Server: SNA Diagnosis Vol 2, FFST Dumps and the VIT	GC27-3667 GC27-3668	These documents help you identify an SNA problem, classify it, and collect information about it before you call the IBM Support Center. The information collected includes traces, dumps, and other problem documentation.
z/OS Communications Server: SNA Data Areas Volume 1 and z/OS Communications Server: SNA Data Areas Volume 2	GC31-6852 GC31-6853	These documents describe SNA data areas and can be used to read an SNA dump. They are intended for IBM programming service representatives and customer personnel who are diagnosing problems with SNA.

Messages and codes

Title	Number	Description
z/OS Communications Server: SNA Messages	SC27-3671	This document describes the ELM, IKT, IST, IUT, IVT, and USS messages. Other information in this document includes: <ul style="list-style-type: none"> • Command and RU types in SNA messages • Node and ID types in SNA messages • Supplemental message-related information
z/OS Communications Server: IP Messages Volume 1 (EZA)	SC27-3654	This volume contains TCP/IP messages beginning with EZA.
z/OS Communications Server: IP Messages Volume 2 (EZB, EZD)	SC27-3655	This volume contains TCP/IP messages beginning with EZB or EZD.
z/OS Communications Server: IP Messages Volume 3 (EZY)	SC27-3656	This volume contains TCP/IP messages beginning with EZY.
z/OS Communications Server: IP Messages Volume 4 (EZZ, SNM)	SC27-3657	This volume contains TCP/IP messages beginning with EZZ and SNM.
z/OS Communications Server: IP and SNA Codes	SC27-3648	This document describes codes and other information that appear in z/OS Communications Server messages.

Index

A

- accessibility
 - contact IBM [285](#)
- accounting
 - exit routine (ISTAUCAG) [81](#)
 - session management function [18](#)
- activating
 - modifiable VTAM exit routines [174](#)
 - using MODIFY EXIT command [174](#)
- adjacent CP name vector, network-qualified (directory services management exit routine) [137](#)
- adjacent link station list information vector [63](#)
- adjacent link station name information vector [64](#)
- adjacent link station selection function in ISTEEXCAA
 - APPN considerations for [27](#)
 - description of [25](#)
 - final register contents [26](#)
 - parameter list [26](#)
- adjacent SSCP name list [55](#)
- adjacent SSCP selection function in ISTEEXCAA
 - description of [22](#)
 - final register contents [22](#)
 - parameter list [22](#)
- alias name translation facility for CNM table [213](#)
- alias selection function in ISTEEXCAA
 - description of [23](#)
 - final register contents [24](#)
 - input parameter list [57](#)
 - output parameter list [61](#)
 - parameter list [23](#)
 - sample code
 - first entry of input parameter list [257](#)
 - second entry of input parameter list [257](#)
 - suffix allocation subroutine [259](#)
 - sample processing [256](#)
 - sample session flow [62](#)
 - USERVAR information [2](#)
- alias selection input parameter list [57](#)
- alias selection output parameter list [61](#)
- ALS list information vector [63](#)
- ALS name information vector [64](#)
- ALS selection function in ISTEEXCAA
 - APPN considerations for [27](#)
 - description of [25](#)
 - final register contents [26](#)
 - parameter list [26](#)
- alternate CDS selection function, directory services management exit routine (ISTEXCDM) [125](#)
- assistive technologies [285](#)
- authorization
 - exit routine (ISTAUCAT) [82](#)
 - session management function
 - initial authorization [14](#)
 - secondary authorization [17](#)

B

- begin function
 - directory services management exit routine (ISTEXCMD) [120](#)
 - session management exit routine (ISTEXCAA)
 - sample code [246](#)
- begin vector in ISTEEXCCS [90](#)
- begin vector in ISTEEXCSD [102](#)
- border node
 - options [134](#)
 - parameter list structure [118–120](#)
 - selection function [123](#)
 - subnetwork routing list structure [139](#)
- build vector in ISTEEXCCS [93](#)

C

- CDS list structure (directory services management exit routine) [141](#)
- CDS selection function, directory services management exit routine (ISTEXCDM) [124](#)
- central resource registration function, directory services management exit routine (ISTEXCDM) [125](#)
- CLSDST PASS processing
 - after SSCP takeover [15](#)
 - ILU resource identification control vector [15](#)
 - initial authorization function [14](#), [15](#)
 - secondary authorization function [15](#)
 - session identification user data field [62](#)
- CLU search exit routine (ELMCLUEX)
 - description of [181](#)
 - sample exit routine for TPF sessions [281](#)
- command verification exit routine (ISTCMMND)
 - description of [105](#)
 - sample exit routine [275](#)
- communication network management (CNM) routing table
 - description [213](#)
 - format [214](#)
 - installation [214](#)
- Communications Server for z/OS, online information [xxii](#)
- configuration services XID exit routine (ISTEXCCS)
 - description of [85](#)
 - MODIFY EXIT command [88](#)
 - multiple dynamic switched major nodes [87](#)
 - resource entry block [94](#)
 - sample exit routine
 - device identification [267](#)
 - monitoring connection status [274](#)
 - name generation function [271](#)
 - tracing information [86](#)
- vector data formats
 - begin vector [90](#)
 - build vector [93](#)
 - connection status vector [96](#)
 - end vector [97](#)
 - failure vector [97](#)

configuration services XID exit routine (ISTEXCCS) (*continued*)
 vector data formats (*continued*)
 XID vector [91](#)
 connection status vector in ISTEXCCS [96](#)
 connection status, monitoring [96](#), [274](#)
 contact
 z/OS [285](#)
 control blocks, TSO (time sharing option)/VTAM
 IKTIPARM [203](#)
 IKTMPL [204](#)
 IKTOPARM [204](#)
 IKTWESTD [205](#)
 IKTXSA [208](#)
 control logical unit search exit routine (ELMCLUEx)
 description of [181](#)
 sample exit routine for TPF sessions [281](#)
 CPNAME field [268](#)
 CPNDEF definition file [268](#)
 criteria for contending CLUs in ELMCLUEx [177](#)
 CSDATA file [274](#)

D

data areas, TSO (time sharing option)/VTAM
 IKTIPARM [203](#)
 IKTMPL [204](#)
 IKTOPARM [204](#)
 IKTWESTD [205](#)
 IKTXSA [208](#)
 deactivating
 modifiable VTAM exit routines [174](#)
 using MODIFY EXIT command [174](#)
 DELIVER RU [215](#)
 design requirements
 CLU search exit routine [183](#)
 command verification exit routine [106](#)
 configuration services XID exit routine [87](#)
 directory services management exit routine [116](#)
 generic resource resolution exit routine [143](#)
 performance monitor exit routine [149](#)
 selection of definitions for dependent LUs exit routine [100](#)
 session accounting exit routine [81](#)
 session authorization exit routine [83](#)
 session management exit routine [4](#)
 USERVAR exit routine [111](#)
 virtual route pacing window size calculation exit routine [78](#)
 virtual route selection exit routine [71](#)
 destination logical unit (DLU)
 directory services management exit routine (ISTEXCDM) [136](#)
 session management exit routine (ISTEXCAA)
 adjacent SSCP vector [56](#)
 gateway information vector [55](#)
 directory services management exit routine (ISTEXCDM)
 alternate CDS selection function [125](#)
 begin function [120](#)
 CDS selection function [124](#)
 central resource registration function [125](#)
 description of [115](#)
 design requirements [116](#)
 diagnostic information [117](#)
 DLU information structure [136](#)

directory services management exit routine (ISTEXCDM) (*continued*)
 end function [126](#)
 environment vector list [126](#)
 exit options [132](#)
 final register contents [116](#)
 function code and related search information [128](#)
 initial authorization function [121](#)
 initial register contents [115](#)
 network-qualified adjacent CP name vector [137](#)
 OLU information structure [135](#)
 parameter descriptions [126](#)
 parameter list summary [118](#)
 PCID modifier structure [138](#)
 replaced function [126](#)
 replacing function [126](#)
 search correlator structure [137](#)
 search task list [138](#)
 tracing information [115](#)
 user data field [132](#)
 user-supplied parameter data [132](#)
 DLU (destination logical unit)
 directory services management exit routine (ISTEXCDM) [136](#)
 session management exit routine (ISTEXCAA)
 adjacent SSCP vector [56](#)
 gateway information vector [55](#)
 DNS, online information [xxiii](#)
 downstream load utility (DSCU) message routing [213](#)
 dynamic definition
 dependent logical unit [97](#)
 independent logical unit [25](#)
 switched connection [85](#)
 dynamic definition request vector in ISTEXCSD [102](#)
 dynamic selection of session connections for independent LUs [25](#)
 dynamic switched definitions [85](#)

E

ELMCLUEx (CLU search exit routine)
 description of [181](#)
 sample exit routine for TPF sessions [281](#)
 encryption key [104](#), [105](#)
 end function
 directory services management exit routine (ISTEXCDM) [126](#)
 session management exit routine (ISTEXCAA)
 description of [33](#)
 final register contents [34](#)
 parameter list [33](#)
 sample code [261](#)
 sample processing [261](#)
 end vector in ISTEXCCS [97](#)
 end vector in ISTEXCSD [105](#)
 environment vector list
 directory services management exit routine (ISTEXCDM) [126](#)
 session management exit routine (ISTEXCAA) [34](#)
 ER-TESTED RU [215](#)
 exit options
 directory services management exit routine (ISTEXCDM) [132](#)
 session management exit routine (ISTEXCAA) [43](#)
 exit replaced function in ISTEXCAA

exit replaced function in ISTEEXCAA (*continued*)
 description of [29](#)
 final register contents [29](#)
 parameter list [29](#)
 exit replacement function in ISTEEXCAA
 description of [28](#)
 final register contents [29](#)
 parameter list [28](#)
 exit routines, logon manager
 CLU search (ELMCLUEx)
 description of [181](#)
 design requirements [183](#)
 parameter descriptions [183](#)
 register contents [182](#)
 sample exit routine for TPF sessions [281](#)
 exit routines, TSO (time sharing option)/VTAM
 attention handler for 3270 terminals (IKTIDSX3) [198](#)
 attention handler for 3767 and 3770 terminals (IKTRTX3) [201](#)
 editing for 3767, 3770, and 2741 terminals (IKTRTX4) [201](#)
 editing on unsupported terminals (IKTGETXT) [197](#)
 error handling for unsupported terminals (IKTCASX1) [194](#)
 I/O manager initialization (IKTINX2) [200](#)
 input edit for 3270 terminals (IKTIDSX2) [198](#)
 input edit for 3767 and 3770 terminals (IKTRTX2) [201](#)
 logon edit (IKTINX1) [199](#)
 output edit for 3270 terminals (IKTIDSX1) [197](#)
 output edit for 3767, 3770, and 2741 terminals (IKTRTX1) [200](#)
 TGET edit for 3270 terminals (IKTIDSX4) [198](#)
 user message language-hardware verification (IKTCASX2) [195](#)
 exit routines, VTAM
 activating [175](#)
 command verification (ISTCMMND)
 description of [105](#)
 design requirements [106](#)
 examples of processing [108](#)
 final register contents [106](#)
 initial register contents [106](#)
 parameter list [107](#)
 sample [275](#)
 configuration services XID (ISTEXCCS)
 description of [85](#)
 design requirements [87](#)
 final register contents [87](#)
 initial register contents [86](#)
 parameter lists [88](#)
 sample exit routine [267](#)
 vector descriptions [89](#)
 deactivating [175](#)
 directory services management exit (ISTEXCDM)
 design requirements [116](#)
 final register contents [116](#)
 initial register contents [115](#)
 generic resource resolution (ISTEXCGR)
 design requirements [143](#)
 parameter lists [144](#)
 processing of [147](#)
 register contents [142](#)
 installing [173](#)
 passing user-defined data to [175](#)
 exit routines, VTAM (*continued*)
 performance monitor (ISTEXCPM)
 description of [148](#)
 design requirements [149](#)
 multiple exit support [148](#)
 parameter lists [151](#)
 register contents [149](#)
 vector descriptions [153](#)
 replacing [175](#)
 selection of definitions for dependent LUs (ISTEXCSD)
 description of [97](#)
 design requirements [100](#)
 final register contents [99](#)
 initial register contents [99](#)
 parameter list [100](#)
 vector descriptions [101](#)
 session accounting (ISTAUCAG)
 description of [81](#)
 design requirements [81](#)
 final register contents [81](#)
 initial register contents [81](#)
 session authorization (ISTAUCAT)
 description of [82](#)
 design requirements [83](#)
 final register contents [83](#)
 initial register contents [83](#)
 parameter list [84](#)
 session management (ISTEXCAA)
 description of [1](#)
 design requirements [4](#)
 final register contents [3](#)
 initial register contents [3](#)
 parameter descriptions [34](#)
 parameter list [6](#)
 sample exit routine [241](#)
 session management exit routine [4](#)
 USERVAR (ISTEXCUV)
 description of [109](#)
 design requirements [111](#)
 final register contents [111](#)
 initial register contents [110](#)
 parameter descriptions [113](#)
 parameter list [112](#)
 sample exit routine for TPF sessions [277](#)
 VR pacing window size calculation (ISTPUCWC)
 bounds specification [77](#)
 description of [77](#)
 design requirements [78](#)
 final register contents [78](#)
 initial register contents [78](#)
 parameter list [79](#)
 used with IMS [80](#)
 VR selection exit routine
 changing the VR selection list [74](#)
 description of [70](#)
 design requirements [71](#)
 final register contents [71](#)
 initial register contents [71](#)
 parameter list [72](#)
 Exit Services, VTAM [67](#)
 explicit route characteristics table [79](#)
 extended recovery facility (XRF)
 function code for [37](#)
 session switch [21](#)

F

- failure vector in ISTECCS [97](#)
- final accounting function in ISTECAA
 - description of [18](#)
 - final register contents [19](#)
 - parameter list [18](#)
- fully qualified associated resource name control vector [51](#)
- function code and related session information
 - directory services management exit routine (ISTEXCDM) [128](#)
 - session management exit routine (ISTEXCAA)
 - description of [36](#)
 - function code [37](#), [38](#)
 - INIT OTHER CD processing [37](#)
 - related session information [38](#)

G

- gateway
 - class-of-service names [52](#)
 - information vector (OLU) [52](#)
 - NCP name [52](#)
 - path selection function in ISTECAA
 - description of [19](#)
 - final register contents [20](#)
 - locating real LU name, sample code [252](#)
 - modifying GWPATH list, sample code [254](#)
 - parameter list [20](#)
 - sample processing [251](#)
 - path selection list [53](#)
- gateway path selection list [53](#)
- generic resource resolution exit routine
 - design requirements [143](#)
 - invoke flags [145](#)
 - member list [145](#)
 - parameter lists [145](#)
 - processing, sample [147](#)
 - register contents [142](#)
 - user data field [144](#)
- GWPATH definition statement [19](#)

H

- high performance data transfer
 - CSM buffer pool vector [167](#)
 - CSM storage usage vector [167](#)
 - performance detector identifiers [154](#)
- high performance routing (HPR)
 - environment vectors [35](#), [36](#)
 - function code and related session information [37](#), [38](#)
 - global environment vector [154](#)
 - parameter list [30](#)
 - performance data vector identifiers [154](#)
 - RTP data vector [170](#), [171](#)
 - session flow [227](#)
 - session management exit options [45](#)
 - virtual route selection function
 - description [32](#)
 - final register contents [33](#)
 - parameter list [32](#)
- HSRTSIZE start option [211](#)

I

- IKTCASX1, error handling for unsupported terminals [194](#)
- IKTCASX2, user message language-hardware verification [195](#)
- IKTGETXT, editing on unsupported terminals [197](#)
- IKTIDSX1, output editing for 3270 terminals [197](#)
- IKTIDSX2, input editing for 3270 terminals [198](#)
- IKTIDSX3, attention handler for 3270 terminals [198](#)
- IKTIDSX4, TGET edit for 3270 terminals [198](#)
- IKTINX1, logon edit [199](#)
- IKTINX2, I/O manager initialization [200](#)
- IKTIPARM [203](#)
- IKTMPL [204](#)
- IKTOPARM [204](#)
- IKTRTX1, output edit for 3767, 3770, and 2741 terminals [200](#)
- IKTRTX2, input edit for 3767 and 3770 terminals [201](#)
- IKTRTX3, attention handler for 3767 and 3770 terminals [201](#)
- IKTRTX4, edit for 3767, 3770 and 2741 terminals [201](#)
- IKTWESTD [205](#)
- IKTXSA [208](#)
- ILU resource identifier control vector [45](#)
- IMS (Information Management System) [80](#)
- Information APARs [xx](#)
- Information Management System (IMS) [80](#)
- INIT OTHER CD processing
 - alias selection
 - function [23](#)
 - output parameter list [61](#)
 - ALS selection function [25](#)
 - DLU adjacent SSCP vector [56](#)
 - function code and related session information [37](#)
 - initial authorization function
 - description of processing [14](#), [15](#)
 - when driven [3](#)
 - OLU adjacent SSCP vector [56](#)
- INIT-LOAD RU [215](#)
- initial accounting function in ISTECAA
 - description of [18](#)
 - final register contents [19](#)
 - parameter list [18](#)
- initial authorization function
 - directory services management exit routine (ISTEXCDM) [115](#)
 - session management exit routine (ISTEXCAA)
 - description of [14](#)
 - final register contents [16](#)
 - for INIT OTHER CD processing [14](#), [15](#)
- installing
 - CNM routing tables [214](#)
 - description of [213](#)
 - IBM supplied information [213](#)
 - message routing [213](#)
 - sample table [219](#)
 - VTAM exit routines [173](#)
- Internet, finding z/OS information online [xxii](#)
- ISSTMGC00, an RU supplemental table [214](#)
- ISTAUCAG (session accounting exit routine)
 - description of [81](#)
 - design requirements [81](#)
 - final register contents [81](#)
 - initial register contents [81](#)

ISTAUCAG (session accounting exit routine) (*continued*)

library for VTAM exit routine [173](#)

ISTAUCAT (session authorization exit routine)

design requirements [83](#)

final register contents [83](#)

initial register contents [83](#)

parameter list [84](#)

ISTCMMND (command verification exit routine)

description of [105](#)

sample exit routine [275](#)

ISTEXCAA (session management exit routine)

adjacent link station (ALS) selection function

description of [25](#)

final register contents [26](#)

parameter list [26](#)

adjacent SSCP selection function

description of [22](#)

final register contents [22](#)

parameter list [22](#)

alias selection function

description of [23](#)

final register contents [24](#)

INIT OTHER CD processing [23](#), [61](#)

input parameter list [57](#)

output parameter list [61](#)

parameter list [23](#)

sample code [257](#)

sample processing [256](#)

begin function

description of [13](#)

final register contents [14](#)

parameter list [13](#)

sample code [246](#)

sample processing [244](#)

description of [1](#)

design requirements [4](#)

end function

description of [33](#)

final register contents [34](#)

parameter list [33](#)

sample code [261](#)

sample processing [261](#)

example of functions [2](#)

exit replaced function

description of [29](#)

final register contents [29](#)

parameter list [29](#)

exit replacement function

description of [28](#)

final register contents [29](#)

parameter list [28](#)

final register contents [3](#)

fully qualified associated resource name control vector [51](#)

gateway path selection function

description of [19](#)

final register contents [20](#)

locating real LU name, sample code [252](#)

modifying GWPATH list, sample code [254](#)

parameter list [20](#)

sample processing [251](#)

initial and final accounting functions

description of [18](#)

final register contents [19](#)

ISTEXCAA (session management exit routine) (*continued*)

initial and final accounting functions (*continued*)

parameter list [18](#)

initial authorization function

description of [14](#)

final register contents [16](#)

INIT OTHER CD processing [14](#)

initial register contents [3](#)

parameter descriptions [34](#)

parameter list summary [6](#)

parameters

alias selection input parameter list [57](#)

alias selection output parameter list [61](#)

ALS list information vector [63](#)

ALS name information vector [64](#)

DLU adjacent SSCP vector [56](#)

DLU gateway information vector [55](#)

environment vector list [34](#)

exit options [43](#)

function code and related session information [36](#)

gateway path selection list [53](#)

ILU resource identifier control vector [45](#)

OLU adjacent SSCP vector [56](#)

OLU gateway information vector [52](#)

PLU resource identifier control vector [45](#)

session authorization data vector [66](#)

session identifier [52](#)

session initiation user data field [62](#)

session management data area [69](#)

SLU resource identifier control vector [45](#)

SSCP name list [55](#)

TCP/IP Information Control vector [70](#)

time-of-day field [52](#)

user data field [43](#)

USERVAR resource identifier control vector [45](#)

VR/TP list information vector [65](#)

return codes

adjacent SSCP selection function [22](#)

alias selection function [24](#)

ALS selection function [26](#)

begin function [14](#)

end function [34](#)

exit replaced function [30](#)

exit replacement function [29](#)

gateway path selection function [20](#)

initial and final accounting functions [19](#)

initial authorization function [16](#)

secondary authorization function [18](#)

virtual route selection function [31](#)

XRF session switch function [21](#)

sample cross-network session [221](#)

sample cross-network session for INIT OTHER CD [222](#)

sample exit routine

description of [241](#)

function processing and sample code [244](#), [261](#)

function selection [243](#)

initialization [241](#)

invoking VTAM Exit Services [262](#)

secondary authorization function

description of [17](#)

final register contents [18](#)

parameter list [17](#)

sample code [250](#)

sample processing [248](#)

ISTEXCAA (session management exit routine) (*continued*)
 virtual route selection function
 description of [30](#)
 final register contents [31](#)
 parameter list [31](#)
 with cross-network or single-network sessions [1](#)
 XRF session switch function
 description of [21](#)
 final register contents [21](#)
 parameter list [21](#)

ISTEXCCS (configuration services XID exit routine)
 description of [85](#)
 MODIFY EXIT command [88](#)
 multiple dynamic switched major nodes [87](#)
 resource entry block [94](#)
 sample exit routine
 device identification [267](#)
 monitoring connection status [274](#)
 name generation function [271](#)
 vector data formats
 begin vector [90](#)
 build vector [93](#)
 connection status vector [96](#)
 end vector [97](#)
 failure vector [97](#)
 XID vector [91](#)

ISTEXCDM (directory services management exit routine)
 alternate CDS selection function [125](#)
 begin function [120](#)
 CDS selection function [124](#)
 central resource registration function [125](#)
 description of [115](#)
 design requirements [116](#)
 DLU information structure [136](#)
 end function [126](#)
 environment vector list [126](#)
 exit options [132](#)
 final register contents [116](#)
 function code and related search information [128](#)
 initial authorization function [121](#)
 initial register contents [115](#)
 network-qualified adjacent CP name vector [137](#)
 OLU information structure [135](#)
 parameter list summary [118](#)
 PCID modifier structure [138](#)
 replaced function [126](#)
 replacing function [126](#)
 search correlator structure [137](#)
 user-supplied parameter data [132](#)

ISTEXCSD (selection of definitions for dependent LUs)
 description of [97](#)
 resource entry block [103](#)
 vector descriptions
 begin vector [102](#)
 dynamic definition request vector [102](#)
 end vector [105](#)

ISTEXCUV (USERVAR exit routine)
 description of [109](#)
 exit routine for TPF sessions [180](#)
 sample exit routine for TPF sessions [277](#)

ISTEXCVR (virtual route selection routine)
 description of [70](#)
 design requirements [71](#)
 examples of processing [75](#)

ISTEXCVR (virtual route selection routine) (*continued*)
 final register contents [71](#)
 initial register contents [71](#)
 parameter list [72](#)

ISTMGC01, a default CNM routing table
 example for two application programs [216](#)
 installing [214](#)
 installing CNM tables
 header format [214](#)
 table entry format [214](#)
 request types routed by [215](#)
 unsolicited requests [215](#)
 with user written alias translation facility [216](#)

ISTPUCWC (virtual route pacing window size calculation routine)
 description of [77](#)
 design requirements [78](#)
 final register contents [78](#)
 initial register contents [78](#)
 parameter list [79](#)
 use with IMS [80](#)

K

keyboard
 navigation [285](#)
 PF keys [285](#)
 shortcut keys [285](#)

L

libraries for VTAM exit routines [173](#)
 license, patent, and copyright information [287](#)
 link-editing exit routines [173](#)
 logon manager, CLU search exit routine (ELMCLUEx) [181](#)
 logon-interpret routine requirements
 final register contents [217](#)
 initial register contents [216](#)
 operation [218](#)
 parameter list [217](#)

M

macroinstructions, TSO (time sharing option)/VTAM
 IKTIPTARM [203](#)
 IKTMPL [204](#)
 IKTOPARM [204](#)
 IKTWESTD [205](#)
 IKTXSA [208](#)

mainframe
 education [xx](#)

message authentication [104](#), [105](#)

MODIFY EXIT command
 activating exit routines [174](#)
 configuration services XID exit routine parameter list [89](#)
 deactivating exit routines [174](#)
 enhancement [244](#), [262](#)
 parameter string [112](#)
 passing user-defined data to exit routines [175](#)
 replacing exit routines [174](#)
 SDDL exit routine parameter list [101](#)

module names for exit routines [174](#)
 monitoring of connection status [96](#), [274](#)

- monitoring of switched connection time [96, 274](#)
- multiple dynamic switched major nodes
 - build vector format [93](#)
 - configuration services XID exit routine [85](#)
 - design requirements [87](#)
 - generating a node name [267](#)
- multiple node persistent session
 - application data vector [154, 173](#)
 - application recovery data vector [154, 172](#)
 - function code and related session information [42](#)
 - session management exit options [45](#)
 - structure data vector [173](#)

N

- name field of VTAM macros [61](#)
- navigation
 - keyboard [285](#)
- network identifier (NETID) registration table, sample [245](#)
- network-qualified adjacent CP name vector (directory services management exit routine) [137](#)
- NIDDEF definition file [268](#)
- NMVT RU [215](#)
- NODEID field [268](#)
- NSRU types routed [214](#)

O

- OLU (origin logical unit)
 - directory services management exit routine (ISTEXCDM) [135](#)
 - session management exit routine (ISTEXCAA)
 - adjacent SSCP vector [56](#)
 - gateway information vector [52](#)
- operator commands
 - activating exit routines [174](#)
 - deactivating exit routines [174](#)
 - enhancement [244, 262](#)
 - parameter string [112](#)
 - passing user-defined data to exit routines [175](#)
 - replacing exit routines [174](#)
 - SDDLU exit routine parameter list [101](#)
- origin logical unit (OLU)
 - directory services management exit routine (ISTEXCDM) [135](#)
 - session management exit routine (ISTEXCAA)
 - adjacent SSCP vector [56](#)
 - gateway information vector [52](#)
- OSRTSIZE start option [212](#)

P

- pacing response [77](#)
- parameter data, user-supplied (directory services management exit routine) [132](#)
- parameter lists
 - CLU search exit routine
 - application program activation and deactivation [185](#)
 - CLU comparison [186](#)
 - CLU session initiation and termination [185](#)
 - exit initialization and termination [184](#)
 - command verification exit routine [107](#)

- parameter lists (*continued*)
 - configuration services XID exit routine [88](#)
 - directory services management exit routine [118](#)
 - generic resource resolution exit routine [144](#)
 - performance monitor exit routine [150](#)
 - selection of definitions for dependent LUs exit routine [100](#)
 - session authorization exit routine [84](#)
 - session management exit routine
 - adjacent SSCP selection function [22](#)
 - alias selection function [23](#)
 - ALS selection function [26](#)
 - begin function [13](#)
 - end function [33](#)
 - exit replaced function [29](#)
 - exit replacement function [28](#)
 - gateway path selection function [20](#)
 - initial and final accounting function [18](#)
 - secondary authorization function [17](#)
 - virtual route selection function [31](#)
 - XRF session switch function [21](#)
 - USERVAR exit routine [112](#)
 - virtual route pacing window size calculation exit routine [79](#)
 - virtual route selection exit routine [72](#)
- path information unit (PIU), VR pacing windows [77](#)
- PCID modifier structure, directory services management exit routine [138](#)
- performance data vector identifiers [154](#)
- performance monitor exit routine (ISTEXCPM)
 - basic route data vector [168](#)
 - design requirements [149](#)
 - final register contents [149](#)
 - global APPN directory services border node vector [165](#)
 - global APPN directory services vector [164](#)
 - global APPN topology data vector [166](#)
 - global environment vector [154](#)
 - global installation-wide exit vector [159](#)
 - global session vector [162](#)
 - global storage buffer pool vector [159](#)
 - global storage CSA vector [161](#)
 - global storage GETBLK vector [162](#)
 - global storage private storage vector [161](#)
 - initial register contents [149](#)
 - multiple exit support [148](#)
 - parameter list [150](#)
 - performance data parameter list [151](#)
 - performance data vector general format [153](#)
 - user data field [151](#)
- PIU (path information unit), VRpacing window [77](#)
- PLU fully qualified associated resource name control vector [51](#)
- PLU resource identifier control vector [45](#)
- prerequisite information xx
- procedure related identifier (PRID) [213](#)

R

- RECFMS RU [215](#)
- RECMS RU [215](#)
- replaced function, directory services management exit routine (ISTEXCDM) [126](#)
- replacement tables, defining [211](#)
- replacing

- replacing (*continued*)
 - modifiable VTAM exit routines [174](#)
 - using MODIFY EXIT command [174](#)
- replacing function, directory services management exit routine (ISTEXCDM) [126](#)
- REQTAIL macro [189](#)
- request types routed by CNM interface [215](#)
- resource entry block
 - in ISTEXCCS [94](#)
 - in ISTEXCSD [103](#)
- return codes
 - CLU search exit routine
 - acceptable range [182](#)
 - returns from the sample exit for TPF sessions [282](#)
 - command verification exit routine [107](#)
 - configuration services XID exit routine [87](#)
 - directory services management exit [116](#)
 - selection of definitions for dependent LUs exit routine [99](#)
 - session authorization exit routine [83](#)
 - session management exit routine
 - adjacent SSCP selection function [22](#)
 - alias selection function [24](#)
 - ALS selection function [26](#)
 - begin function [14](#)
 - end function [34](#)
 - exit replaced function [30](#)
 - exit replacement function [29](#)
 - gateway path selection function [20](#)
 - initial and final accounting function [19](#)
 - initial authorization function [16](#)
 - secondary authorization function [18](#)
 - virtual route selection function [31](#)
 - XRF session switch function [21](#)
 - USERVAR exit routine
 - acceptable range [111](#)
 - returns from the sample exit for TPF sessions [280](#)
 - virtual route selection exit routine [71](#)
 - VR pacing window size calculation exit routine [78](#)
- RFC (request for comments)
 - accessing online [xxii](#)
- ROUTE-INOP RU [215](#)

S

- sample
 - CLU search exit routine for TPF sessions [281](#)
 - command verification exit routine [275](#)
 - configuration services XID exit routine [267](#)
 - session management exit routine [241](#)
 - USERVAR exit routine for TPF sessions [277](#)
- SDDL exit routine
 - description of [97](#)
 - resource entry block [103](#)
 - vector descriptions
 - begin vector [102](#)
 - dynamic definition request vector [102](#)
 - end vector [105](#)
- search correlator structure, directory services management exit routine (ISTEXCDM) [137](#)
- search information, directory services management exit routine (ISTEXCDM) [128](#)
- search task list, directory services management exit routine [138](#)

- secondary authorization function in ISTEXCAA
 - description of [17](#)
 - final register contents [18](#)
 - parameter list [17](#)
 - sample code [250](#)
 - sample processing [248](#)
- selection of definitions for dependent LUs exit routine (ISTEXCSD)
 - description of [97](#)
 - resource entry block [103](#)
 - vector descriptions
 - begin vector [102](#)
 - dynamic definition request vector [102](#)
 - end vector [105](#)
- session accounting exit routine (ISTAUCAG)
 - description of [81](#)
 - design requirements [81](#)
 - final register contents [81](#)
 - initial register contents [81](#)
 - library for VTAM exit routine [173](#)
- session accounting exit routine (ISTAUCAT)
 - design requirements [83](#)
 - final register contents [83](#)
 - initial register contents [83](#)
 - parameter list [84](#)
- session authorization data vector [66](#)
- session flows
 - mixed subarea/APPN
 - ILU-initiated (part 1 of 2) [235](#)
 - PLU-initiated (part 1 of 2) [237](#)
 - SLU-initiated [239](#)
 - subarea
 - cross-network for CDINIT [221](#)
 - cross-network for INIT_OTHER_CD [222](#)
 - DSRLST for alias selection [231](#)
 - ILU-initiated (third-party) for alias selection [227](#)
 - PLU-initiated for alias selection [225](#)
 - SLU-initiated for alias selection [224](#)
 - SLU-initiated with USERVAR name for alias selection [229](#)
 - VR selection for boundary function LUs [235](#)
- session identifier [52](#)
- session initiation user data field [62](#)
- session management data area [69](#)
- session management exit routine (ISTEXCAA)
 - adjacent link station (ALS) selection function
 - description of [25](#)
 - final register contents [26](#)
 - parameter list [26](#)
 - adjacent SSCP selection function
 - description of [22](#)
 - final register contents [22](#)
 - parameter list [22](#)
 - alias selection function
 - description of [23](#)
 - final register contents [24](#)
 - INIT OTHER CD processing [23](#), [61](#)
 - input parameter list [57](#)
 - output parameter list [61](#)
 - parameter list [23](#)
 - sample code [257](#)
 - sample processing [256](#)
 - begin function
 - description of [13](#)

- session management exit routine (ISTEXCAA) (*continued*)
 - begin function (*continued*)
 - final register contents [14](#)
 - parameter list [13](#)
 - sample code [246](#)
 - sample processing [244](#)
 - description of [1](#)
 - design requirements [4](#)
 - end function
 - description of [33](#)
 - final register contents [34](#)
 - parameter list [33](#)
 - sample code [261](#)
 - sample processing [261](#)
 - example of functions [2](#)
 - exit replaced function
 - description of [29](#)
 - final register contents [29](#)
 - parameter list [29](#)
 - exit replacement function
 - description of [28](#)
 - final register contents [29](#)
 - parameter list [28](#)
 - final register contents [3](#)
 - fully qualified associated resource name control vector [51](#)
 - gateway path selection function
 - description of [19](#)
 - final register contents [20](#)
 - locating real LU name, sample code [252](#)
 - modifying GWPATH list, sample code [254](#)
 - parameter list [20](#)
 - sample processing [251](#)
 - initial and final accounting functions
 - description of [18](#)
 - final register contents [19](#)
 - parameter list [18](#)
 - initial authorization function
 - description of [14](#)
 - final register contents [16](#)
 - INIT OTHER CD processing [14](#)
 - initial register contents [3](#)
 - parameter descriptions [34](#)
 - parameter list summary [6](#)
 - parameters
 - alias selection input parameter list [57](#)
 - alias selection output parameter list [61](#)
 - ALS list information vector [63](#)
 - ALS name information vector [64](#)
 - DLU adjacent SSCP vector [56](#)
 - DLU gateway information vector [55](#)
 - environment vector list [34](#)
 - exit options [43](#)
 - function code and related session information [36](#)
 - gateway path selection list [53](#)
 - ILU resource identifier control vector [45](#)
 - OLU adjacent SSCP vector [56](#)
 - OLU gateway information vector [52](#)
 - PLU resource identifier control vector [45](#)
 - session identifier [52](#)
 - session initiation user data field [62](#)
 - SLU resource identifier control vector [45](#)
 - SSCP name list [55](#)
 - time-of-day field [52](#)

- session management exit routine (ISTEXCAA) (*continued*)
 - parameters (*continued*)
 - user data field [43](#)
 - USERVAR resource identifier control vector [45](#)
 - VR/TP list information vector [65](#)
 - return codes
 - adjacent SSCP selection function [22](#)
 - alias selection function [24](#)
 - ALS selection function [26](#)
 - begin function [14](#)
 - end function [34](#)
 - exit replaced function [30](#)
 - exit replacement function [29](#)
 - gateway path selection function [20](#)
 - initial and final accounting functions [19](#)
 - initial authorization function [16](#)
 - secondary authorization function [18](#)
 - virtual route selection function [31](#)
 - XRF session switch function [21](#)
 - sample cross-network session [221](#)
 - sample cross-network session for INIT OTHER CD [222](#)
 - sample exit routine
 - description of [241](#)
 - function processing and sample code [244](#), [261](#)
 - function selection [243](#)
 - initialization [241](#)
 - invoking VTAM Exit Services [262](#)
 - secondary authorization function
 - description of [17](#)
 - final register contents [18](#)
 - parameter list [17](#)
 - sample code [250](#)
 - sample processing [248](#)
 - tracing information [2](#)
 - virtual route selection function
 - description of [30](#)
 - final register contents [31](#)
 - parameter list [31](#)
 - with cross-network or single-network sessions [1](#)
 - XRF session switch function
 - description of [21](#)
 - final register contents [21](#)
 - parameter list [21](#)
- shortcut keys [285](#)
- SLU fully qualified associated resource name control vector [51](#)
- SLU resource identifier control vector [45](#)
- SLU table name vector [50](#)
- SNA protocol specifications [283](#)
- softcopy information [xx](#)
- SSCP selection function, adjacent, in ISTEXCAA
 - description of [22](#)
 - final register contents [22](#)
 - parameter list [22](#)
- summary of changes [xxv](#)
- switched connection time, monitoring [96](#), [274](#)
- SYS1.LPALIB [173](#)

T

- TCAS processing [193](#)
- TCP/IP
 - online information [xxii](#)
 - TCP/IP Information Control vector [70](#)

Technotes [xx](#)

terminal control address space processing [193](#)

time sharing option (TSO)/VTAM exit routines

- attention handler for 3270 terminals (IKTIDSX3) [198](#)
- attention handler for 3767 and 3770 terminals (IKTRTX3) [201](#)
- editing for 3767, 3770, and 2741 terminals (IKTRTX4) [201](#)
- editing on unsupported terminals (IKTGETXT) [197](#)
- error handling for unsupported terminals (IKTCASX1) [194](#)
- I/O manager initialization (IKTINX2) [200](#)
- input edit for 3270 terminals (IKTIDSX2) [198](#)
- input edit for 3767 and 3770 terminals (IKTRTX2) [201](#)
- logon edit (IKTINX1) [199](#)
- output edit for 3270 terminals (IKTIDSX1) [197](#)
- output edit for 3767, 3770, and 2741 terminals (IKTRTX1) [200](#)
- TGET edit for 3270 terminals (IKTIDSX4) [198](#)
- user message language-hardware verification (IKTCASX2) [195](#)

time-of-day field [52](#)

TPF logon manager, CLU search exit routine (ELMCLUEx) [181](#)

TR-INQ RU [215](#)

trademark information [290](#)

TSO (time sharing option)/VTAM control blocks

- IKTIPARM [203](#)
- IKTMPL [204](#)
- IKTOPARM [204](#)
- IKTWESTD [205](#)
- IKTXSA [208](#)

TSO (time sharing option)/VTAM data areas

- IKTIPARM [203](#)
- IKTMPL [204](#)
- IKTOPARM [204](#)
- IKTWESTD [205](#)
- IKTXSA [208](#)

TSO (time sharing option)/VTAM exit routines

- attention handler for 3270 terminals (IKTIDSX3) [198](#)
- attention handler for 3767 and 3770 terminals (IKTRTX3) [201](#)
- editing for 3767, 3770, and 2741 terminals (IKTRTX4) [201](#)
- editing on unsupported terminals (IKTGETXT) [197](#)
- error handling for unsupported terminals (IKTCASX1) [194](#)
- I/O manager initialization (IKTINX2) [200](#)
- input edit for 3270 terminals (IKTIDSX2) [198](#)
- input edit for 3767 and 3770 terminals (IKTRTX2) [201](#)
- logon edit (IKTINX1) [199](#)
- output edit for 3270 terminals (IKTIDSX1) [197](#)
- output edit for 3767, 3770, and 2741 terminals (IKTRTX1) [200](#)
- TGET edit for 3270 terminals (IKTIDSX4) [198](#)
- user message language-hardware verification (IKTCASX2) [195](#)

TSO (time sharing option)/VTAM macroinstructions

- IKTIPARM [203](#)
- IKTMPL [204](#)
- IKTOPARM [204](#)
- IKTWESTD [205](#)
- IKTXSA [208](#)

U

user data field

- command verification exit routine [108](#)
- directory services management exit routine (ISTEXCDM) [132](#)
- generic resource resolution exit routine (ISTEXCGR) [144, 150](#)
- performance monitor exit routine (ISTEXCPM) [151](#)
- session management exit routine (ISTEXCAA) [43](#)

user interface

- ISPF [285](#)
- TSO/E [285](#)

user-supplied parameter data (directory services management exit routine) [132](#)

USERVAR exit routine (ISTEXCUV)

- description of [109](#)
- exit routine for TPF sessions [180](#)
- sample exit routine for TPF sessions [277](#)

USERVAR resource identifier control vector [45](#)

unsolicited network services RUs [213](#)

V

virtual route

- descriptor block format [74](#)
- examples of processing [75](#)
- pacing window size calculation exit routine [77](#)
- selection exit routine [70](#)

virtual route pacing window size calculation exit routine (ISTPUCWC)

- description of [77](#)
- design requirements [78](#)
- final register contents [78](#)
- initial register contents [78](#)
- parameter list [79](#)
- use with IMS [80](#)

virtual route selection exit routine (ISTEXCVR)

- changing the virtual route selection list [74](#)
- description of [70](#)
- design requirements [71](#)
- examples of processing [75](#)
- final register contents [71](#)
- initial register contents [71](#)
- parameter list [72](#)

virtual route selection function in ISTEXCAA

- description of [30](#)
- final register contents [31](#)
- parameter list [31](#)

virtual route/transmission priority list information vector [65](#)

VR pacing window size calculation exit routine

- description of [77](#)
- design requirements [78](#)
- final register contents [78](#)
- initial register contents [78](#)
- parameter list [79](#)
- use with IMS [80](#)

VR selection exit routine

- changing the virtual route selection list [74](#)
- description of [70](#)
- design requirements [71](#)
- examples of processing [75](#)
- final register contents [71](#)
- initial register contents [71](#)

- VR selection exit routine (*continued*)
 - parameter list [72](#)
- VR/TP list information vector [65](#)
- VRPWSnn operand (PATH definition statement) [77](#)
- VTAM
 - CNM routing table [213](#)
 - exit routines
 - activating [175](#)
 - command verification (ISTCMMND) [105](#)
 - configuration services XID (ISTEXCCS) [85](#)
 - deactivating [175](#)
 - directory services management exit (ISTEXCDM) [115](#)
 - installing [173](#)
 - passing user-defined data to [175](#)
 - performance monitor (ISTEXCPM) [148](#)
 - replacing [175](#)
 - selection of definitions for dependent LUs (ISTEXCSD) [97](#)
 - session accounting (ISTAUCAG) [81](#)
 - session authorization (ISTAUCAT) [82](#)
 - session management (ISTEXCAA) [1](#)
 - USERVAR (ISTEXCUV) [109](#)
 - virtual route selection (ISTEXCVR) [70](#)
 - VR pacing window size calculation (ISTPUCWC) [77](#)
 - Exit Services [67](#)
 - VTAM module names for exit routines [174](#)
 - VTAM terminal I/O coordinator processing [193](#)
 - VTAM, online information [xxii](#)
 - VTIOC processing [193](#)

W

- window size calculation, default algorithm [77](#)

X

- XID vector in ISTEXCCS [91](#)
- XRF (extended recovery facility)
 - function code for [37](#)
 - session switch [21](#)
- XRF session switch function in ISTEXCAA
 - final register contents [21](#)
 - parameter list [21](#)

Z

- z/OS Basic Skills Information Center [xx](#)
- z/OS, documentation library listing [291](#)



Product Number: 5655-ZOS

SC27-3666-70

