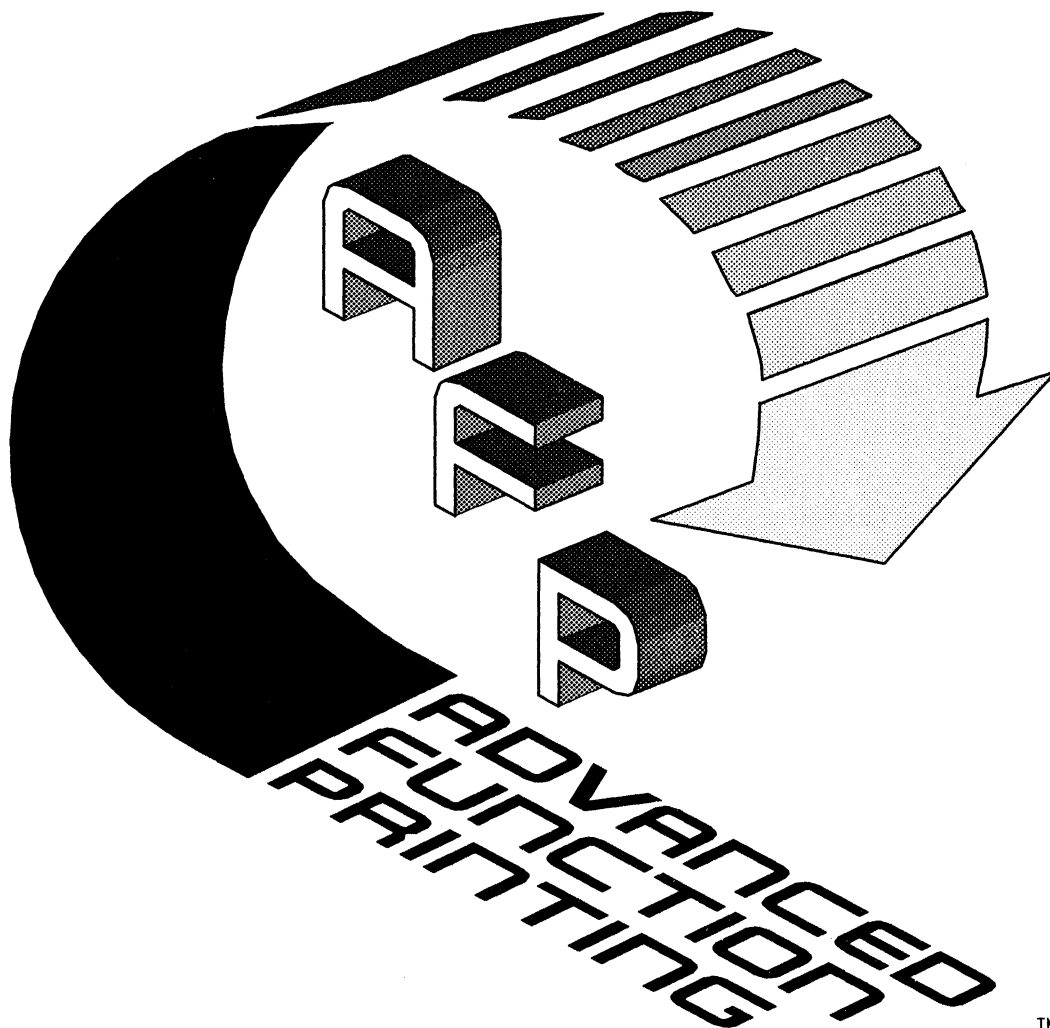




Overlay Generation Language/370 Diagnosis Guide and Reference

LH40-0208-00



TM

Note!

Before using this information and the product it supports, be sure to read the general information in "Notices" on page v.

First Edition (October 1990)

This is a new publication that replaces LV32-0510, which is now obsolete. This edition applies to the IBM Overlay Generation Language/370 (OGL/370) licensed program Release 1.0, Program Number 5688-191 (which replaces Program Numbers 5665-308 for MVS, 5664-293 for VM, and 5666-324 for VSE), and to any subsequent release of the program until otherwise indicated in new editions or technical newsletters.

Because changes are made periodically to the information herein, consult the latest *IBM System/370, 30xx, 4300, and 9370 Processors Bibliography*, GC20-0001 for the editions that are applicable and current before using this publication in connection with the operation of IBM systems.

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM licensed program in this publication is not intended to state or imply that only IBM's licensed program may be used. Any functionally equivalent program may be used instead.

Order publications through your IBM representative or the IBM branch office serving your locality. Publications are not stocked at the address given below.

A form for reader's comments appears at the back of this publication. If the form has been removed, address your comments to:

IBM Corporation
Information Development
Department 588
P.O. Box 1900
Boulder, Colorado 80301-9191

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

© Copyright International Business Machines Corporation 1984, 1990. All rights reserved.

Note to U.S. Government Users — Documentation related to restricted rights — Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract with IBM Corp.

Contents

Part One. Diagnosis Guide	1
Introduction	3
People Involved	3
Diagnostician	3
IBM Support Center Representative	4
IBM Specialist	4
IBM Change Team	4
Specifying a Program Failure	5
Getting Started	7
Developing a Keyword String	9
Component Identification Keyword	10
Type-of-Failure Keyword	11
Module Keyword	20
Modifier Keywords	21
Maintenance and Release Level Keyword	23
Search Argument Procedure	25
Calling the IBM Support Center	27
APARs	28
Part Two. Diagnosis Reference	29
Program Overview	31
OGL/370 Process Flow	33
Detailed Program Structure	35
Command Processing	35
Image Optimization	37
AFP Data Stream Generation	39
Module Directory	41
Control Block Directory	45
Service Aids	47
CBDUMP Command	47
The CBDUMP Command Statement	47
Using CBDUMP	48
The Scenario	48
Narrowing the Problem	48
Output from the CBDUMP Command	49
Logic Dump	52
Master Control Block	54
Font Control Block	55

Segment Control Block	57
Group Control Block	58
Line Control Block	60
Unoptimized Raster Pattern Control Block	62
Optimized Raster Pattern Control Block	64
Text Control Block	66
Final Disposition Report	68
 Appendix A. Message-to-Module Cross-Reference	 69
 Appendix B. Listing an Overlay under MVS	 73
 Appendix C. Listing an Overlay under VSE	 75
 Appendix D. OGL/370 Storage Requirements	 77
Area of Raster Patterns	77
 Appendix E. National Language Support	 79
Defining the Languages in MVS	79
Defining the Languages in VM	80
Defining the Languages in VSE	81
 Appendix F. Related Publications	 83
 Glossary	 85

Figures

1. Message Identification Structure	16
2. Overlay Generation Language/370	30
3. Overlay Generation Language/370 Process Flow	32
4. Overlay Generation Language/370 (Command Processing)	34
5. Overlay Generation Language/370 (Image Optimization)	36
6. Overlay Generation Language/370 (AFP Data Stream Generation)	38
7. Specifying the CBDUMP Command	47
8. An Example of a Source Listing	50
9. An Example of a Logic Dump	52
10. An Example of a Master Control Block Dump	54
11. An Example of a Font Control Block Dump	55
12. An Example of a Segment Control Block Dump	57
13. An Example of a Group Control Block Dump	58
14. An Example of a Line Control Block Dump	60
15. An Example of an Unoptimized Raster Pattern Control Block Dump	62
16. An Example of an Optimized Raster Pattern Control Block Dump	64
17. An Example of a Text Control Block Dump	66
18. An Example of a Final Disposition Report (VM)	68

Notices

References in this publication to products or services of IBM do not suggest or imply that IBM will make them available in all countries where IBM does business or that only products or services of IBM may be used. Noninfringing equivalents may be substituted, but the user must verify that such substitutes, unless expressly designated by IBM, work correctly. No license, expressed or implied, to patents or copyrights of IBM is granted by furnishing this document.

Programming Interface

This book is intended to help diagnose problems concerning OGL/370, and to communicate them to an IBM Support Center representative. The book contains diagnostic information for OGL/370. **WARNING:** Do not use Diagnostic information as a programming interface.

Trademarks and service marks

The following are trademarks of the IBM Corporation:

OGL®
OGL/370.
PSF®

Preface

This book gives information that helps you diagnose failures in the IBM Overlay Generation Language/370 (OGL/370) licensed program, Program Number 5688-191 (which replaces Program Numbers 5665-308 for MVS, 5664-293 for VM, and 5666-324 for VSE), and communicate them to an IBM Support Center representative.

This book helps you communicate with IBM Support Center representatives to isolate the source of an OGL/370 failure. This book describes the program organization and flow of control, but it does not provide you with enough detail to change or to correct the program logic. A thorough review of the *Advanced Function Printing: Diagnosis Guide* is strongly recommended as an aid in problem isolation.

This book is organized in two major divisions: "Part One. Diagnosis Guide" and "Part Two. Diagnosis Reference."

Part One. Diagnosis Guide

Part One contains information to help you diagnose failures in OGL/370 and describe them to an IBM representative. The information in this part is useful only for diagnosing failures in the operation of the OGL/370 program. We assume, therefore, that you have already determined that OGL/370 is the failing component and that a usage error did not cause the failure. "Part One. Diagnosis Guide" is divided into the following chapters:

- "Introduction" explains the keyword concept and how to use the book.
- "Specifying a Program Failure" explains keyword strings in more detail. It outlines how to build one and use it to help solve your problem.
- "Getting Started" tells you about preliminary activities and information gathering.
- "Developing a Keyword String" explains in detail how to create a keyword string.
- "Search Argument Procedure" tells you how to use the keyword string to search the IBM software support data base.
- "Calling the IBM Support Center" gives you guidance about the information to supply when contacting IBM.

Part Two. Diagnosis Reference

Part Two contains detailed information about OGL/370 processing and diagnostic aids. Part Two is designed to help you understand the flow of control through OGL/370 so you can communicate better with an IBM program specialist about a problem; to help you diagnose program failures; and to help you to provide diagnostic information to IBM.

This book is not intended to supply all the information necessary for module maintenance.

Part Two contains the following chapters:

- "Program Overview" presents an overview of OGL/370, describing its physical characteristics, purposes, and functions. It also provides a high level description of the internal structure of OGL/370.
- "Module Directory" summarizes the functions of each program-logic module.
- "Control Block Directory" summarizes the functions of the control blocks which are available to the user for OGL/370 diagnosis.
- "Service Aids" contains descriptions of the dumps or tools and their output, if any, that collect or analyze diagnostic information for this program. The internal trace facility for OGL/370 is included in this section.
- Appendix A, "Message-to-Module Cross-Reference" lists the messages produced by OGL/370 with the module or modules that detect the need for the message.
- Appendix B, "Listing an Overlay under MVS" provides sample job control language (JCL) to print an overlay (in hexadecimal format) for possible APAR documentation.
- Appendix C, "Listing an Overlay under VSE" provides sample job control statements (JCS) to print an overlay (in hexadecimal format) for possible APAR documentation.
- Appendix D, "OGL/370 Storage Requirements" provides information which helps you to estimate OGL/370's virtual storage requirements.
- Appendix E, "National Language Support" describes how to change the default and alternate languages.
- Appendix F, "Related Publications" lists publications containing related information.
- "Glossary" contains a glossary of specialized terms that are used in this book.

Use of this Book

Before contacting the IBM Support Center, use this book as follows:

- Employ the processes described in "Part One. Diagnosis Guide" to determine the program component and the type of failure and to verify that the current problem has not been previously reported and corrected.
- Study "Program Overview" on page 31 in "Part Two: Diagnosis Reference" to become familiar with OGL/370 concepts.
- If you know which function is in operation, study "OGL/370 Process Flow" in "Part Two: Diagnosis Reference" for information on processing paths used by the suspected area of malfunction.

Prerequisite Knowledge

To use this book, you need a basic understanding of OGL/370 at the level contained in the *OGL/370: User's Guide and Reference*.

For a list of related publications, refer to Appendix F, "Related Publications" on page 83.

Summary of Amendments

Although this is a new book (LH40-0208), it is essentially a revision of *OGL/370: Diagnosis Guide and Reference*, LV32-0510-01. It includes changes brought about by the changeover from OGL Release 2 to OGL/370 Release 1.0. (OGL/370 is one program product, and operates under the three operating systems: MVS, VSE, and VM.)

Major differences between the old book and this one include:

- The description of CBDUMP—it now allows partial, as well as full tracing.
- The module directory includes many new and renamed modules.

Part One. Diagnosis Guide

Introduction

This diagnosis guide helps you to describe failures in the operation of the IBM Overlay Generation Language/370 (OGL/370) licensed program through the use of keywords. A keyword is a word that describes one characteristic of a program failure. You are guided to develop a keyword string that describes a program failure.

By following the instructions in this book, you can determine whether your failure has been previously documented and corrected. This book can assist you in communicating with product support personnel.

After you have developed a keyword string, you can use these keywords to help you discover whether a correction exists for the failure. You can do this either by using the keywords to search an IBM software support data base, or by providing them when you contact your IBM Support Center.

The term *software support data base* is used in this book to represent the Information/System (I/S) or the Early Warning System (EWS). The I/S is an online data base that contains up-to-date resolutions of authorized program analysis reports (APARs). The EWS is a microfiche copy of the information contained in the I/S. If you do not have access to the software support data base, you should contact your IBM Support Center and give them the keywords, plus any other relevant information (see "Calling the IBM Support Center" on page 27).

If the failure is known, there is usually a correction for it. If not, the keywords can help IBM correct the problem.

People Involved

The people involved in the diagnosis and correction process are listed below, along with a brief description of their tasks.

Diagnostician

The diagnostician is the principal user of this book. His or her primary task is to identify the cause of a failure and, if the cause is OGL/370, to describe the failure using a keyword string.

If the diagnostician has access to the software support data base then he or she can use the string to search the data base. This search helps determine if the same failure has been described previously and, if so, whether a correction exists. The diagnostician can contact the IBM Support Center for help in doing the search.

IBM Support Center Representative

The Support Center representative has three functions in the diagnosis and correction process:

- To help in the diagnostic search
- To provide a correction if one exists
- To refer the problem to an IBM specialist if a correction is not found.

IBM Specialist

The IBM specialist contacts the diagnostician. Together they:

- Verify that the diagnostician used the correct keywords in creating the keyword string
- Gather additional required information (described in "Calling the IBM Support Center" on page 27) about the failure.

If the problem has not been reported previously, the IBM specialist tries to develop a bypass to help the customer continue productive work. The specialist also refers the problem to the IBM Change Team by preparing an authorized program analysis report (APAR). He or she gives the customer the APAR number to use when sending requested documentation to the IBM Change Team.

IBM Change Team

The tasks of the IBM Change Team are:

- To develop corrections for program problems reported on APARs
- To make corrections available to the customer reporting the problem
- To modify the keyword string to describe the failure more accurately, if necessary
- To add the keyword string and the program correction to the software support data base.

Specifying a Program Failure

Each keyword describes one characteristic of a program failure. The first keyword in a keyword string identifies the OGL/370 licensed program by its component identification number. To some extent, each keyword after the first is optional. If circumstances make it particularly difficult to follow the instructions given for choosing some keyword, you can omit it. In general, however, if you contact IBM, you will be asked to identify your problem with a full keyword string as described in the following pages. A complete keyword string for OGL/370 identifies the following:

- Component identification number
- Type of failure
- Failing module, modifier, or both
- Maintenance level.

A search of an indexed data base with only the component identification number keyword detects all reported problems for the entire licensed program. Adding other keywords to the search argument reduces the number of matches and increases the chances of locating a match for your particular problem.

Your search is most successful if you follow these rules:

- Use only the keywords in this book.
- Spell the keywords exactly as they are given here.
- Follow the keyword procedure in the order shown.

Getting Started

Before you develop a keyword string, do the following:

1. If the OGL/370 licensed program has been changed by a program temporary fix (PTF) since you last used it, note the PTFs and the modules that were changed. If the error occurs as the result of changes in the program and cannot be corrected, note the change that caused the error.
2. Correct all problems diagnosed by messages.
3. Note the sequence of events that led to the error condition. This information may be useful in developing a keyword string and is needed to submit an APAR.
4. Examine your specifications (such as JCL/JCS or command stream) to verify that they have been specified correctly. If an error is found, correct it and continue the task.

Developing a Keyword String

To develop a keyword string, follow the steps in each of the procedures (described later) for identifying characteristics of a program failure, unless a branch to another procedure is specified.

Note: The procedures described here for developing a keyword string are intended to ensure that any two people will develop an identical set of keywords when working on the same type of problem caused by the same program error.

The keyword string is **highlighted** throughout the procedures. Each step provides a partial keyword string that describes what is known at that time about the program problem. A partial keyword string may require inserting a specific piece of information. (For example, you might need to add the identifier of the message received to the MSGx keyword.) The information to be added will be known at that time. You should continue to develop the keyword string until you are instructed to use it as a search argument.

If you have access to the software support data base, a search of the data base is recommended as soon as the keyword string is complete enough to make a search reasonably productive without requiring excessive research of the problem. If an early search is unsuccessful, instructions for continuing the problem diagnosis are given.

Go to "Component Identification Keyword" on page 10 to begin developing a keyword string.

Component Identification Keyword

The component identification number is the first keyword in a keyword string and is used whenever OGL/370 is suspected of being the failing component. The component identification keyword should be used with at least one other keyword to search the software support data base. Used alone, it will produce a full listing of APARs against OGL/370.

Procedure:

1. Select the correct component identification number. The component identification number for OGL/370 is:
568819101 (for MVS, VM, and VSE)
2. Go to "Type-of-Failure Keyword" on page 11 to determine the type of failure that occurred.

Type-of-Failure Keyword

The following keywords identify the types of failures that can occur in OGL/370:

Keyword	Type of Failure
ABENDxxx	Processing of OGL/370 ends abnormally. Go to "ABENDxxx" on page 12.
DOC	Information is missing or incorrect in one of the OGL/370 manuals. Go to "DOC" on page 13.
INCORROUT	Output from the program is incorrect or missing. Go to "INCORROUT" on page 14.
LOOP	The program is doing something repetitively. Go to "LOOP" on page 15.
MSGx	An OGL/370 error message has been received. Go to "MSGx" on page 16.
PERFM	Performance is degraded. Use this keyword only when no other keyword is appropriate. Go to "PERFM" on page 18.
WAIT	The program does not seem to be doing anything. Go to "WAIT" on page 19.

Procedure: From the preceding list, select one keyword that best describes the program failure, and follow the procedure for that particular failure type.

ABENDxxx

Use **ABENDxxx** when the host system or the program that produces OGL/370 output ends abnormally.

Do not use this keyword if the abnormal end was forced by the host system because of a prolonged wait state or an endless loop. For those situations, refer to the WAIT and LOOP keywords.

Procedure:

1. If the abend identification includes a code, add it to the keyword, replacing the xxx. For example, use ABEND0C6 if an 0C6 abend occurred. The format of the partial keyword string is:

Component ID No.	Type
-------------------------	-------------

For example:

568819101	ABEND0C6
------------------	-----------------

2. Locate the address at which the abend occurred and record all the available information about the OGL/370 module in control prior to the abend.
3. Go to "Module Keyword" on page 20 to select a module name for use in developing the keyword string.

DOC

Use **DOC** when a programming problem appears to be caused by incorrect or missing information in one of the OGL/370 publications.

Procedure:

1. If the problem is in an OGL/370 publication, place the order number after the DOC keyword, and omit the hyphens. For example, if the order number is S544-3702-1, the DOC keyword is **S54437021**.
2. Prepare a description of the problem. You should also include this information in the error description when submitting an APAR. The format of the keyword string is:

Component ID No.	Type	Order No.
For example:		
568819101	DOC	S54437021

3. If you do not have access to the software support data base, go to step 5.
4. To determine if documentation deficiency has been reported as a problem, go to "Search Argument Procedure" on page 25. If the search is unsuccessful, return to this procedure list to continue.
5. If this is a documentation deficiency that may cause lost time for other users, please contact your IBM Support Center to inform them about the problem. You should be able tell them the text location of the error in the manual and describe the problem it caused. Go to "Calling the IBM Support Center" on page 27.
6. If the documentation problem is less severe, use the reader's comment form attached to the back of the manual to suggest improvements to the publication.

INCORROUT

Use **INCORROUT** when the expected output was not received or when the output was different from what was expected.

Possible OGL/370 **INCORROUT** sub-keywords are:

- Output from the source listing (**SOURCE**—messages)
- Output from normal processing (**OVLY**—the generated overlay)

The printed document (the result of the overlay)

A print of the overlay advanced function printing (AFP) data stream (output from LIBRPRINT under VSE or AMSPRINT under MVS).¹

Procedure: The format of the partial keyword string is:

Component ID No.	Type	Output
------------------	------	--------

For example:

568819101	INCORROUT	OVLY
-----------	-----------	------

1. If the incorrect output is a message, consider using the **MSGx** keyword. Also, see Appendix A, "Message-to-Module Cross-Reference" on page 69 to determine the module that detected the error.
2. To reduce your search, go to "Modifier Keywords" on page 21 and select a command name.

¹ Data streams created in the VM environment must be sent to an alternate operating system to be printed.

LOOP

Use **LOOP** if some part of the program code executes endlessly or if some part of the output repeats endlessly.

Do not use this keyword for an endlessly repeated message or an intentional loop used to wait for some resource. However, you may use this keyword with, or in addition to, the **MSGx** keyword. For these conditions, refer to the **MSGx** or the **WAIT** keyword.

Note: You might be more successful in finding a matching problem description if you build two sets of keywords, one with the keyword **WAIT** and one with the keyword **LOOP**.

Procedure:

1. Have the operator cancel the job with a dump using local installation procedures.
2. If OGL/370 suspends activity for no apparent reason, it could be in a loop or wait state. A loop can be evident by a symptom such as a repeatedly printed page of output. If a loop is not evident, use the **WAIT** keyword.
3. If you have a complete storage dump, locate the address at which the loop occurred. The format of the partial keyword string is:

Component ID No.	Type
------------------	------

For example:

568819101	LOOP
-----------	------

4. Go to "Module Keyword" on page 20 to select a module name for use in developing the keyword string.

MSGx

Use **MSGx** when any of the following conditions occur:

- A message is issued that indicates an internal program error.
- A message is not issued in some set of conditions that should cause it to be issued.
- A message is issued in some set of conditions that should not cause it to be issued.
- Data is missing from a message, or message data is incorrect.

Each message issued by OGL/370 is identified by a set of 8 characters in the form DZInnnnI. As described by Figure 1, **DZI** is the program identifier for OGL/370, **nnnn** is the message serial number, and **I** identifies the type code.

However, if the message begins with some other set of letters, refer to your operating system documentation to determine the source of the message.

Figure 1 shows the message identification structure used by OGL/370.

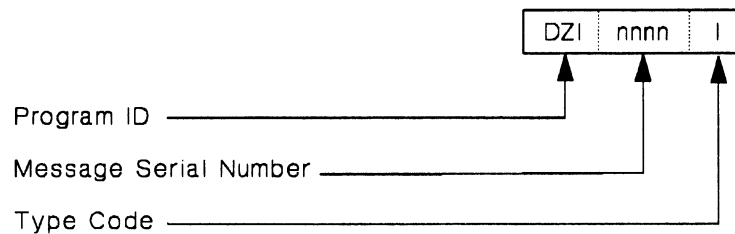


Figure 1. Message Identification Structure

Procedure:

1. Replace the **x** in the **MSGx** message keyword with the 8-character message identification (DZInnnnI). For example, if the message identification is DZI0104I, the message keyword is:

Component ID No.	Type
------------------	------

For example:

568819102	MSGDZI0104I
-----------	-------------

2. Use Appendix A, "Message-to-Module Cross-Reference" on page 69 to identify the module or modules issuing the message.

The format of the partial keyword string is:

Component ID No.	Type	Module
------------------	------	--------

For example:

568819101	MSGDZI0501I	DZILKORN
-----------	-------------	----------

Repeat the string for multiple module keywords from the same message:

568819101	MSGDZI0501I	DZILKPLC
568819101	MSGDZI0501I	DZILKTXI
568819101	MSGDZI0501I	DZILRWIX

3. Go to "Modifier Keywords" on page 21 and select the command to be added to the keyword string.

PERFM

If you suspect a problem with some aspect of the performance of OGL/370, you may build a keyword string to describe the problem and see whether the problem has been reported and solved previously.

Note: There is a difference between a performance problem, as indicated by the **PERFM** type-of-failure keyword, and performance tuning.

Procedure:

Use this procedure only when you encounter a problem with some aspect of performance for a period of time and are unable to correct the problem.

1. Record the actual performance, the expected performance, and the source of the expected performance criteria.
2. Use **PERFM** as your type-of-failure keyword.

The format of the partial keyword string is:

Component ID No.	Type
------------------	------

For example:

568819101	PERFM
-----------	-------

3. Go to "Modifier Keywords" on page 21 and select the command to be added to the keyword string.

WAIT

The host system, OGL/370, or some program that services OGL/370 has suspended activity, without issuing a message, while waiting for some condition to be satisfied. If the reason for the wait is not obvious, have the operator cancel the job with a dump.

Use this keyword when the following conditions exist:

- No activity over a period of time
- No repeated visual output
- Wait bit "on" in the host system's program status word (PSW).

Do not use **WAIT** if the wait occurs after an abend or because of an endless loop in OGL/370. In such instances, use the **ABEND** or **LOOP** keyword.

Note: You might be more successful in finding a matching problem description if you build two keyword strings, one with the keyword **WAIT** and one with the keyword **LOOP**.

Procedure:

1. Have the operator cancel the job with a dump using local installation procedures, and note the wait bit status in the PSW.

The format of the partial keyword string is:

Component ID No.	Type
------------------	------

For example:

568819101	WAIT
-----------	------

2. Go to "Module Keyword" on page 20 to select a module name for use in developing the keyword string.

Module Keyword

This keyword is used to identify the module related to the program failure.

Procedure:

1. Obtain a storage dump of the area of virtual storage where the OGL/370 code was loaded.
2. Find the instruction address where:
 - the abend or dump occurred
 - the SVC for the WAIT was issued
 - the loop occurred.
3. Back up from that instruction until you find a six- to eight-character module name (for example DZILSCTL) followed by a module date and an IBM copyright statement.

Add the module name to the keyword string.

The format of the partial keyword string is:

Component ID No.	Type	Module
------------------	------	--------

For example:

568819101	LOOP	DZILSCTL
-----------	------	----------

4. Go to "Modifier Keywords" on page 21 and select the appropriate command to be added to the keyword string.

Modifier Keywords

The modifier keywords are optional and can be used to identify a particular problem area. These keywords consist of OGL/370 commands and subcommands. Wherever you use a modifier keyword, you can also use the abbreviation specified in Table 1 on page 22.

Procedure:

1. Refer to Table 1 on page 22, which shows a complete list of the modifier keywords and their descriptions.
2. Use that list to select the modifier keyword that represents the area in which you suspect the failure occurred. Additional information can be acquired by reviewing OGL/370 control statements in the *OGL/370: User's Guide and Reference*.

If a grid that you specified within a group fails to appear on the overlay, your partial set of keywords may look like this:

Component ID No.	Type	Module	Modifier
------------------	------	--------	----------

For example:

568819101	INCORROUT	DZILKDEF	DRAWMASK
-----------	-----------	----------	----------

Another partial set of keywords may contain a sub-keyword:

Component ID No.	Type	Module	Modifier
------------------	------	--------	----------

For example:

568819101	PERFM	DZILKDEF	DEFINE PATTERN
-----------	-------	----------	-------------------

A sub-keyword always follows its modifier keyword.

3. Go to "Maintenance and Release Level Keyword" on page 23 and follow those instructions.

Table 1. Modifier Keywords.

Commands/ Subcommands	Description	Abbreviation
CBDUMP	Prints the control block information used in building the overlay.	none
CONTROL	Specifies the options in effect for the execution of the program.	CNL
DEFINE	Defines a pattern or group of statements for later use in the overlay definition.	DEF
GROUP	A subcommand of DEFINE. DEFINE GROUP identifies a group of statements that define a relocatable section that can be placed on the overlay.	GR
PATTERN	A subcommand of DEFINE. DEFINE PATTERN defines a raster image that can be placed on the overlay.	PA
DRAWBOX	Directs the program to draw boxes on the overlay.	DBX
WITHTEXT	A subcommand of DRAWBOX that directs the program to place one or more lines of text in a box.	WT
DRAWCIRCLE	Directs the program to draw circles on the overlay.	DCR
WITHTEXT	A subcommand of DRAWCIRCLE that directs the program to place one or more lines of text in a circle.	WT
DRAWMASK	Causes a grid to be drawn on the overlay.	DMK
DRAWPATH	Directs the program to draw a path on the overlay.	DPT
DRAWRULE	Permits the specification of rules (lines) to be drawn on the overlay (length, weight, type).	DRL
FONT	Identifies a font to be used in constructing the overlay.	FNT
ORIENT	Defines the orientation of the overlay relative to the paper.	ORN
OVERLAY	Establishes the name, total width and height dimension, and offset of the overlay.	OVL
PLACE	Directs the program to place a previously identified resource (group, pattern, or segment) on the overlay.	PLC
GROUP	A subcommand of PLACE that directs the program to place a section defined with DEFINE GROUP on the overlay.	GR
PATTERN	A subcommand of PLACE that directs the program to place a pattern defined with DEFINE PATTERN on the overlay.	PA
SEGID	A subcommand of PLACE that directs the program to place a page segment identified with SEGMENT on the overlay.	SE
POSITION	Directs the program to move an internal marker to any position (single pel) on the overlay.	POS
SEGMENT	Identifies a page segment to be placed on the overlay.	SEG
SETTEXT	Directs the program to place one or more lines of text on the overlay.	TXT
SETUNITS	Establishes the horizontal and vertical spacing defaults.	SET

Note: The abbreviations in Table 1 are only meaningful for creating a keyword string.

Maintenance and Release Level Keyword

This keyword identifies the maintenance level of OGL/370. Each program temporary fix (PTF) applied to OGL/370 is given a 5-digit identifier. For example, if the last PTF applied was PTF 11355, the keyword is UP11355. To determine the PTF ID, see the *Program Directory*.

Use PTF numbers as a keyword only if you feel that the PTF has caused the problem. The match you are looking for might have been found for a program with an earlier or a later PTF level than yours.

The cover of the *Program Directory* shows the release and modification levels of your OGL/370 program. You must specify these as a three digit number as part of your keyword string.

Procedure:

1. Specify the number of the latest PTF applied to your OGL/370 program. Use this number as a keyword only if you feel that the PTF caused the problem.
2. Specify the program release level as 110, representing Version 1, Release 1, and Modification 0.

You now have all the necessary information for an effective search of known problems in the software support data base.

Your completed set of keywords may look like this:

Component ID No.	Type	Modname	Maintlvl	Releaselvl
568819101	LOOP	DZILSCTL	UP11355	100

For example:

3. If you do not have access to an IBM software support data base, go to "Calling the IBM Support Center" on page 27.
4. If you do have access to a software support data base, go to "Search Argument Procedure" on page 25 and perform the necessary actions.

Search Argument Procedure

Each software support data base keyword describes one aspect of a program failure. The more precisely the keyword describes the program failure, the more selective (narrow) the resulting search will be, thus yielding fewer matches.

The following procedure explains how to use the keyword string as a search argument in a software support data base.

1. Search the software support data base using the complete keyword string developed.
2. Eliminate from the list of matches those APAR fixes that have already been applied to your system.
3. Compare each remaining APAR closing description with the current failure symptoms.
4. If a match and a fix are found, apply and test the fix.
5. If a match is found, but a fix for the problem is not available, notify your IBM representative so you can be contacted when a fix is available.
6. If a match is not found, broaden the search by using the following techniques. The techniques are given in the recommended order of use.
 - Drop keywords from the right of the search argument to broaden the search.
 - Consider dropping the keyword that you, as diagnostician, are least sure of.
7. If a match is still not found using the preceding techniques, go to "Calling the IBM Support Center" on page 27.

Calling the IBM Support Center

Follow these procedures if you do not have access to the software support data base, or if you do have access and the search did not find a match for your problem. Before you call the IBM Support Center to report a problem, be prepared to supply the following information:

- Customer number
- Current service level (PTF list and list of APAR fixes applied)
- Keyword string or strings used to search the software support data base
- CPU number (serial - model).

You may be asked to supply any or all of the following information to describe the environment, activities, or OGL/370 functions related to the problem:

- A description of the problem
- JCL (MVS), JCS (VSE), or system interpreter (VM) listings
- Storage dump (at time of failure)
- Link edit map
- Console printout
- A copy of the source listing to help reproduce the error
- A copy of as small a portion of output as needed to illustrate the failure (sample overlay, or overlay as used by an application program)
- Hexadecimal listing of any overlays created (Appendix B, "Listing an Overlay under MVS" on page 73 and Appendix C, "Listing an Overlay under VSE" on page 75 explain how to do this)
- CBDUMP output (see "CBDUMP Command" on page 47 for further information)
- Copies of any other traces or dumps taken
- For a **WAIT** failure — a complete description of the resource being waited for and the program module that is waiting. This can be obtained from the module call stack, if a dump was taken, or from the partition save area
- For a **LOOP** failure — some indication of the location of the loop. This could be one or more module names taken from the call stack or a description of an interface loop between OGL/370 and a host program. Try to obtain at least a partial trace of the loop
- For **DOC** failures — the location of the error in the documentation.

APARs

An authorized program analysis report (APAR) is prepared only after the diagnostic procedures have been followed and the keyword search has been unsuccessful. This section describes the procedure for preparing an APAR.

If, after you have contacted your IBM Support Center for assistance, a correction is not found for the problem, an IBM specialist will contact you to diagnose the problem. If the specialist determines that the problem is a new one, the specialist prepares an APAR. The specialist will give you an APAR number to identify any material required by IBM to support the correction process.

When submitting material for an APAR to IBM, carefully pack and clearly identify any magnetic tapes. Each magnetic tape submitted must have the following information attached and visible:

- The APAR number assigned by IBM
- A list of files on the tape
- A description of how the tape was made, containing the following information:
 - The syntax needed to get the information from the tape
 - An exact list of the commands used
 - Labelling information for the volume and its files
 - The recording mode and density
 - The record format and block size used for each file.

Each dump and any other printed material must show the APAR number.

Part Two. Diagnosis Reference

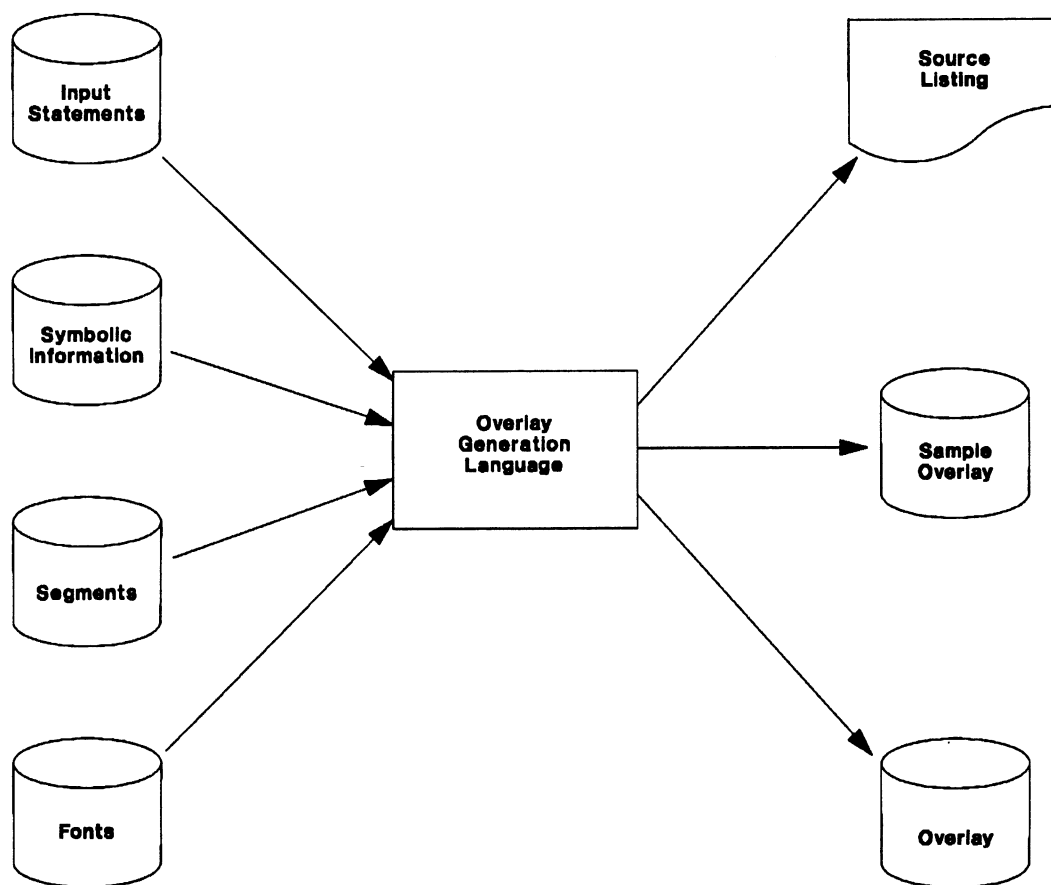


Figure 2. Overlay Generation Language/370

Program Overview

This section provides high-level information about the OGL/370 licensed program.

OGL/370 provides functions that interpret OGL/370 statements and produce a series of advanced function printing (AFP) data stream records that define the overlay.

All OGL/370 modules and macros use the program prefix of **DZI**.

OGL/370 is a batch program that operates in problem program state and performs functions in a manner similar to that of language compilers such as PL/1 and COBOL.

OGL/370 processes the input statements (commands) that you define through a series of internal sub-components and produces a source listing (an input-statement listing, diagnostic messages, and a final disposition report). Optionally, an overlay library member can be created or replaced for subsequent use. Figure 2 on page 30 illustrates the interactions between OGL/370 and its environment.

Figure 2 on page 30 indicates a number of objects in OGL/370's environment with which it interacts to do its job. These are:

- The input data set or file containing the OGL/370 commands
- The symbolic file or data set containing the symbolic information
- The segment library containing page segments
- The font library containing font objects
- The overlay library member in which OGL/370 stores the overlay
- The sample overlay data set or file in which OGL/370 stores the sample overlay
- The destination to which OGL/370 sends the overlay for printing
- The listing data set or file in which OGL/370 stores the source listing.

These objects are created in different ways under each of the operating systems which OGL/370 supports (MVS, VM, and VSE). Appendix C of the *OGL/370: User's Guide and Reference* describes the implementation of each of the objects for each of the operating systems. Appendix B of the same book specifies details of file and data set properties.

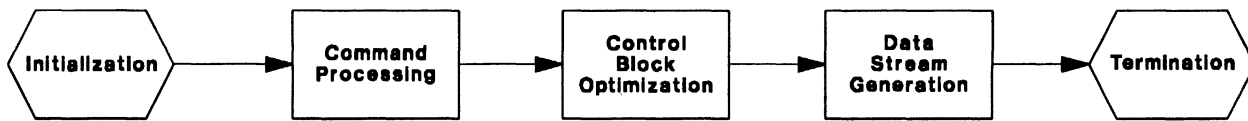


Figure 3. Overlay Generation Language/370 Process Flow

OGI/370 Process Flow

Figure 3 on page 32, Figure 4 on page 34, Figure 5 on page 36, and Figure 6 on page 38 illustrate the relationship between the input statements and the data records that are stored for subsequent use in generating electronic overlays.

These are the steps in the OGI/370 process flow:

1. Initialization

- Obtain and initialize virtual storage for OGI/370 modules.
- Build and initialize the master control block (MCB).
- Build and initialize the trace table.
- Open the listing data set or file.
- Parse the run-time parameters (JCL parameters for MVS, JCS parameters for VSE, or program invocation parameters for VM).
- Load language CSECTS.

2. Command Processing (explained further in "Command Processing" on page 35)

- Open the input data set or file.
- Process the input statements (commands).
 - Process any user errors.
 - Build the control blocks that contain the information to construct the AFP data stream records for the overlay.
 - Create listing data set.
- Close the input data set or file.

3. Image Optimization (explained further in "Image Optimization" on page 37)

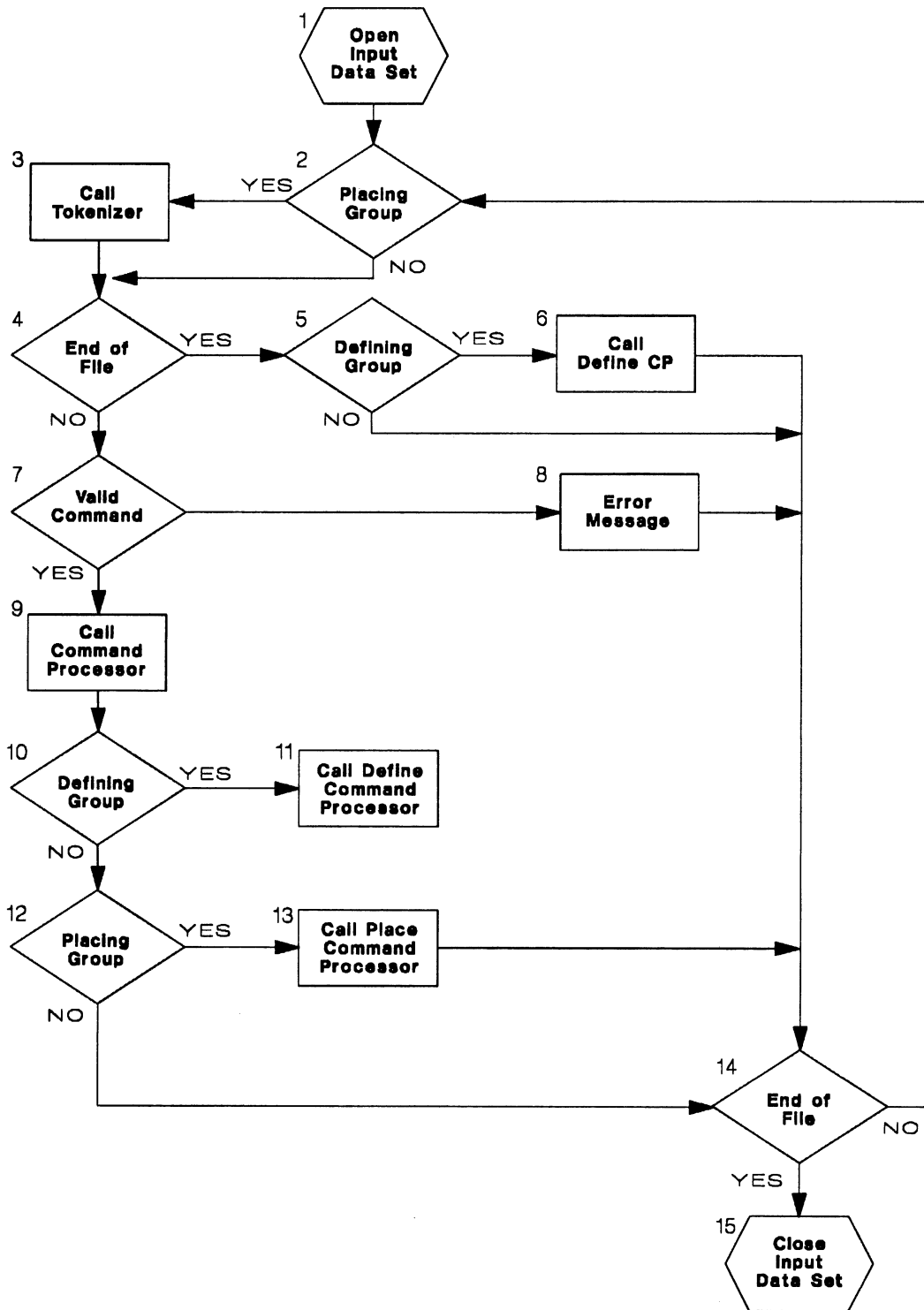
- Print the contents of OGI/370 control blocks in a formatted dump, if specified.
- Split raster images that contain excessive white space into smaller images.
- Remove excessive white space from the edges of the raster images.
- Merge overlapping raster images.

4. AFP Data Stream Generation (explained further in "AFP Data Stream Generation" on page 39)

- Build the AFP data stream records for the overlay.
- Write them to the sample data set or file and if specified, to the overlay library member.

5. Termination

- Close the listing data set or file.
- In VM, view the listing with the editor of your choice.



| Figure 4. Overlay Generation Language/370 (Command Processing)

Detailed Program Structure

OGL/370 statements are read from the input data set or file and processed one at a time. Each statement processed is recorded in an internal storage control block, and control is returned to retrieve another statement until end-of-file is detected. After all the input statements have been internally processed, control is passed to the optimizer, and then to the AFP data stream generation modules. The AFP data stream records are created and spooled for subsequent printing of the form. If the REPLACE or STORE option has been specified, the generated data records are also stored in the overlay library.

Command Processing

Figure 4 on page 34 outlines the flow of control in OGL/370 command processing. The elements of the figure are explained below.

1. Open the input data set or file.
2. Determine whether a group is being placed.
3. Call the tokenizer to read input data (OGL/370 command), write the listing data set, and parse records until the end of a command is recognized. (The parsed command resides in a token table.)
4. Determine whether end-of-file has been reached.
5. Determine whether a group is being defined.
6. Call the DEFINE command processor.
7. Determine whether the command is valid.
8. Write an error message to the source listing.
9. Call the appropriate command processor.
10. Determine whether a group is being defined.
11. Call the DEFINE command processor.
12. Determine whether a group is being placed.
13. Call the PLACE command processor.
14. Determine whether end-of-file has been reached.
15. Close the input data set or file.

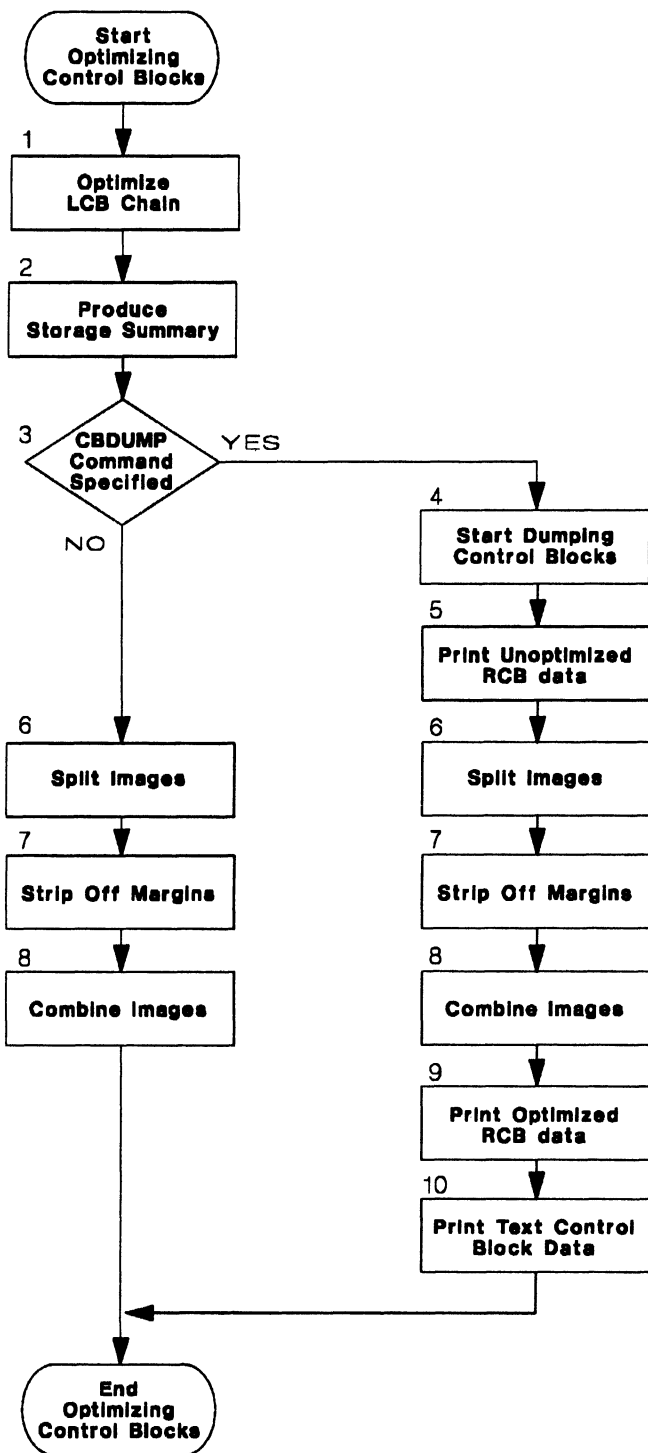


Figure 5. Overlay Generation Language/370 (Image Optimization)

Image Optimization

The purpose of the optimizer is to transform image data created by OGL/370 so that it will be more efficient to print. Figure 5 on page 36 outlines the flow of control in OGL/370 image optimization. The elements of the figure are explained below.

1. The line control block (LCB) chain is optimized. When parallel lines overlap, they are merged into one line.
2. If the overlay definition specified the SUMMARY option for the CONTROL command, a storage summary is printed. This summary can be used for analyzing the printer storage requirements of the overlay.
3. If the CBDUMP command was specified, a control block dump will be printed.
4. Information is dumped from the following data areas:
 - Master control block (MCB)
 - Font control block (FCB)
 - Segment control block (GCB)
 - Group control block (KCB)
 - Line control block (LCB).

Examples of control block dumps are given in "Service Aids" on page 47.

Set flags to ensure that the overlay will not be stored in the overlay library.

5. Raster control block (RCB) data is dumped before it is optimized.
6. Raster images containing large blocks of white space are split into smaller sub-images. These sub-images contain less white space.
7. Any margins of white space (for example, white space appearing along the edges of images) are removed from the image.
8. Raster images that overlap or are very near to one another are combined into a single image.
9. RCB data is dumped after it is optimized.
10. Text control block (XCB) data is dumped.

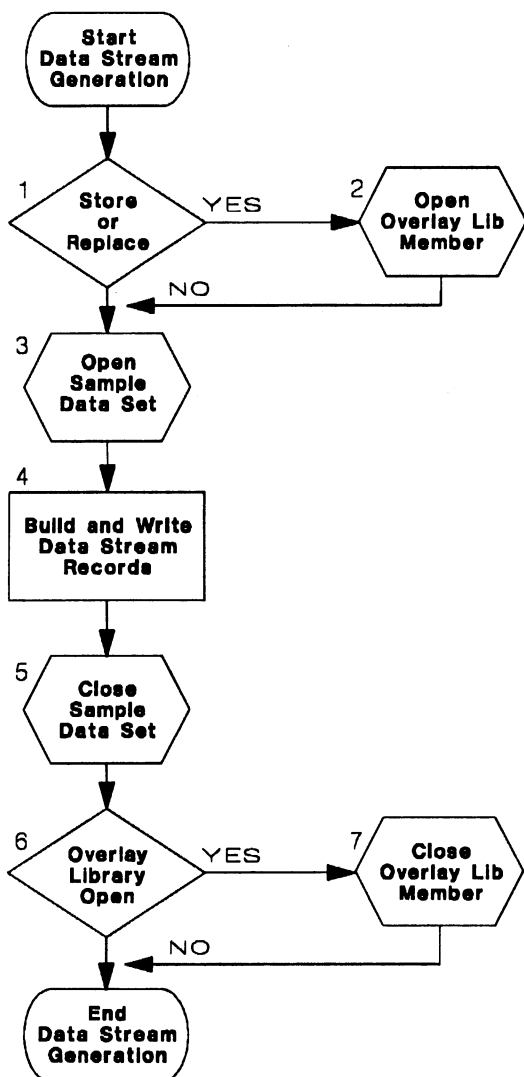


Figure 6. Overlay Generation Language/370 (AFP Data Stream Generation)

AFP Data Stream Generation

Figure 6 on page 38 outlines the flow of control in OGL/370 AFP data stream generation. The elements of the figure are explained below.

1. Determine whether the overlay is to be stored in the overlay library. (STORE or REPLACE was specified.)
2. Open the overlay library member.
3. Open the sample data set or file.
4. Build and write the AFP data stream records based on the information in the control blocks built during the command processing loop.
5. Close the sample data set or file, and in VM, view the listing with the editor of your choice.
6. Close the overlay library member if it was opened.

Module Directory

This module directory is organized alphabetically and provides a summary of the function of each OGL/370 program-logic module that will be encountered by the user in the diagnosis process.

All modules are identical for all the operating systems in which OGL/370 operates (VM, MVS, and VSE).

Table 2 (Page 1 of 4). Module Directory

Module	Description
DZIABEND	DCBabend exit (MVS).
DZICVT00	Get OGL/370's TIOT information.
DZIDTM00	Generate a date/time stamp.
DZILADDP	Add or subtract pel/remainder numbers.
DZILCONP	Convert value/unit to pel/remainder.
DZILCONV	Convert pel/remainder to value/unit.
DZILDBAJ	Adjust SI/SO pairs in mixed DBCS text.
DZILDBAZ	Analyze characters in a mixed text string.
DZILDBGC	Analyze the type of a single mixed text character.
DZILDBGV	Determine byte type in a mixed text string.
DZILDBMV	Move text string to output string area.
DZILDBPP	Pad output text string to required length.
DZILDBTC	Adjust truncated string to valid DBCS boundaries.
DZILDBTT	Translate and test text string.
DZILEXTV	Find amount outside bounds of overlay.
DZILKCNL	CONTROL command processor.
DZILKDBX	DRAWBOX command processor.
DZILKDCR	DRAWCIRCLE command processor.
DZILKDEF	DEFINE command processor.
DZILKDMK	DRAWMASK command processor.
DZILKDPH	DRAWDRAW command processor.
DZILKDRL	DRAWRULE command processor.
DZILKEND	ENDDEF command processor.
DZILKFNT	FONT command interface.
DZILKORN	ORIENTATION command processor.
DZILKOVL	OVERLAY command processor.
DZILKPLC	PLACE command processor.
DZILKPOS	POSITION command processor.
DZILKSEG	SEGMENT command interface.
DZILKSET	SETUNITS command processor.
DZILKTRC	TRACE command processor.
DZILKTXI	SETTEXT command processor.
DZILMSG	Issue messages.
DZILMSGI	Process message inserts.
DZILPARS	Parse the JCL/JCS parameters (MVS,VSE).

Table 2 (Page 2 of 4). Module Directory

Module	Description
DZILPSFC	OGL/PSF interface module (VM).
DZILRAIN	Arc initial angle calculation.
DZILRALI	Find arc/the intersection point.
DZILRANG	Calculate arc angling of dashes.
DZILRANP	Get nex pel on arc.
DZILRARC	Build a raster image of an arc.
DZILRASD	Shade a raster image.
DZILRA1P	Process 1 pel BW arcs.
DZILRBPH	Build DRAWPATH line segments and connections.
DZILRCBS	Shade boxes and circles.
DZILRCEL	Create image cells for dotted, dashed, and solid lines; and shading.
DZILRDBX	Build image of box.
DZILRDCR	Build circle image.
DZILRDGL	Create diagonals for DRAWCIRCLE and DRAWBOX.
DZILRDOP	Optimize diagonal lines.
DZILRDOT	Build a raster image for a single dot.
DZILRDPT	Create user-defined raster patterns.
DZILREPH	Validate DRAWPATH command.
DZILRFCL	Free a cell of memory.
DZILRFPH	Calculate dot/dash segment spacing for DRAWPATH.
DZILRGCL	Get a cell of memory.
DZILRLCN	Build line control block.
DZILRLEP	Locate line width end points.
DZILRMIT	Build miter connection.
DZILRMRG	Merge, clear, or 'and' two raster images.
DZILROPC	Combine raster images to optimize printing.
DZILROPS	Split raster images to optimize printing.
DZILROPT	Control raster image optimization.
DZILRORD	Order end points of an arc.
DZILRPPL	Build point-to-point lines.
DZILRRCB	Build raster control block.
DZILRRPH	Process DRAWPATH REPEAT options.
DZILRSDR	Extract shading parameters.
DZILRSFT	Get font information for text strings.
DZILRSHD	Build LCB shading for rectangular regions.
DZILRSRC	Processor for symbolic search.
DZILRTCN	Text processor.
DZILRTC2	Determine positioning of text.
DZILRWTX	Sub-processor for WITHTEXT.
DZILSCLS	Close an open file or data set.
DZILSCTL	OGL/370 control module.
DZILSDCL	Close an open data set (VSE).
DZILSDCV	Get OGL/370's TIOT information (VSE).
DZILSDDT	Generate a date/time stamp (VSE).

Table 2 (Page 3 of 4). Module Directory

Module	Description
DZILSDFM	Perform a system call to free a block of memory (VSE).
DZILSDGM	Perform a system call to get a block of memory (VSE).
DZILSDKF	FONT command processor (VSE).
DZILSDKS	SEGMENT command processor (VSE).
DZILSDLM	Load a module dynamically.
DZILSDOP	Open a data set (VSE).
DZILSDRD	Read a record from a data set (VSE).
DZILSDSY	DCB synad exit (VSE).
DZILSDWR	Write a record to a data set (VSE).
DZILSD2C	Control building of AFP data stream records (VSE).
DZILSENT	OGL entry module.
DZILSFMN	Perform a system call to free a block of memory.
DZILSGMN	Perform a system call to get a block of memory.
DZILSMCL	Close an open data set (MVS).
DZILSMCV	Get OGL/370's TIOT information (MVS).
DZILSMDT	Generate a date/time stamp (MVS).
DZILSMFM	Perform a system call to free a block of memory (MVS).
DZILSMGM	Perform a system call to get a block of memory (MVS).
DZILSMKF	FONT command processor (MVS).
DZILSMKS	SEGMENT command processor (MVS).
DZILSMOP	Open a data set (MVS).
DZILSMRD	Read a record from a data set (MVS).
DZILSMSY	DCB synad exit (MVS).
DZILSMWR	Write a record to a data set (MVS).
DZILSM2C	Control building of AFP data stream records (MVS).
DZILSOPN	Open a file or data set.
DZILSPRT	Print a dump of control blocks.
DZILSRD	Read a record from a file or data set.
DZILSROP	Optimize the LCB chain.
DZILSUMM	Print storage summary.
DZILSVCL	Close an open file (VM).
DZILSVCV	Get OGL/370's TIOT information (VM).
DZILSVDT	Generate a date/time stamp (VM).
DZILSVFM	Perform a system call to free a block of memory (VM).
DZILSVGM	Perform a system call to get a block of memory (VM).
DZILSVKF	FONT command processor (VM).
DZILSVKS	SEGMENT command processor (VM).
DZILSVOP	Open a file (VM).
DZILSVPR	Parse the invocation (VM).
DZILSVRD	Read a record from a file (VM).
DZILSVWR	Write a record to a file (VM).
DZILSV2C	Control building of AFP data stream records (VM).
DZILSWRT	Write a record to a file or data set.
DZILSYNB	Process syntax of a command portion.

Table 2 (Page 4 of 4). Module Directory

Module	Description
DZILSYNT	Control syntax processing for a whole command.
DZILTOK	Tokenize input command stream.
DZILTRCE	Create entry in trace table.
DZISYNAD	DCB synad exit.

Control Block Directory

This control block directory is organized alphabetically and provides a summary of the function of each OGL/370 control block which will be encountered by the user in the diagnosis process.

All the control blocks described below — except the master control block — describe one instance of an overlay component (for instance a line control block describes one line). All the individual control blocks for one type of component (for instance, all the line control blocks) are linked together into a chain.

All control blocks are identical for all the operating systems in which OGL/370 operates (VM, MVS, and VSE).

Table 3. Control Block Directory

Control Block	Description
DZIMPCB (Font Control Block)	Holds information about a font used in an overlay including character dimensions, coded font names, and underlining.
DZIMPGCB (Segment Control Block)	Holds information about a page segment.
DZIMPKCB (Group Control Block)	Holds information about a group defined with DEFINE GROUP.
DZIMPLCB (Line Control Block)	Holds information about a line or shading.
DZIMPMCB (Master Control Block)	Contains global variables, information about the overlay, and pointers to other control blocks.
DZIMPRCB (Raster-Pattern Control Block)	Holds information about a raster pattern in the overlay. Includes the size and location of the pattern, among other data.
DZIMPXCB (Text Control Block)	Holds information about a text string. Includes location, underlining, font, and other properties.

Service Aids

This section describes procedures, service aids, and internal and external conditions that simplify problem determination—particularly OGL/370's own internal trace facility.

CBDUMP Command

The syntax rules for writing commands, as explained in the *Overlay Generation Language User's Guide and Reference*, are applicable to the CBDUMP command. See Figure 7 for syntax.

The CBDUMP Command Statement

The diagnostician invokes the optional CBDUMP command to generate a formatted listing of OGL/370 control blocks. This may help detect an internal OGL/370 problem. Optionally, a request can be made to create a trace of modules visited and to have their corresponding return codes displayed at the exit of each module. This helps the diagnostician to determine whether the correct execution paths were taken.

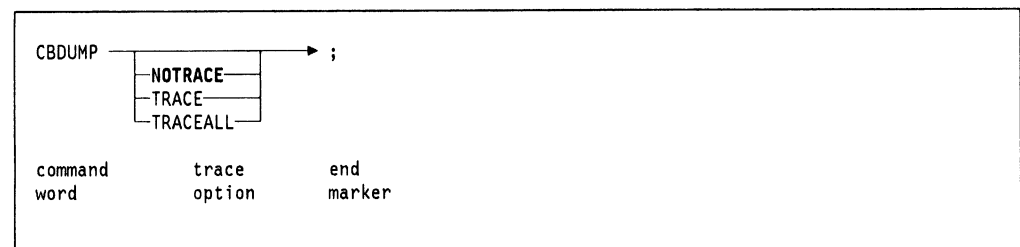


Figure 7. Specifying the CBDUMP Command

- **CBDUMP**—This is the command invocation. No abbreviation is provided. When the CBDUMP command is issued, the overlay is not stored.
- **NOTRACE/TRACE/TRACEALL** —This keyword is optional, and NOTRACE is the default.
 - If TRACE ALL is specified, a trace of all the modules visited is listed.
 - If TRACE is specified, the trace list will not contain the modules DZILADDP, DZILRANP, DZILRFCL and DZILSFMN. These modules are called a great number of times when processing a typical overlay.
 - If NOTRACE is specified, a trace is not listed.
- The semicolon (;) is the delimiter (end marker) for this command.

Using CBDUMP

If you are having problems running OGL/370 or using overlays that were created earlier, you might consider using the CBDUMP command to print the control-block information that was used in building the overlay. The following example uses an actual overlay.

The Scenario

Using the line control block (LCB) as an example, suppose a particular rule was misplaced. For purposes of this scenario, assume this is not a user error.

Narrowing the Problem

1. Add the CBDUMP command to the input stream, and rerun the job. See Figure 7 on page 47 for CBDUMP syntax.
2. Find the OGL/370 command that you used to create the rule in the input stream.
3. Locate the LINE# of the command.
4. Find the same LINE# in the line control block information in the formatted dump under the LINE# label.
5. Locate the x-origin and y-origin.
 - If the values agree with the input, the error occurred in the generation of OGL/370 AFP data stream records.
Note: When checking that the values agree with the input, also check whether the rule is top-left or center positioned. The x and y origins of the rule are always the top-left of the pel of the rule.
 - If the values do not agree, carefully recheck the input.
 - If the input values are correct, the command processor logic for that command caused the problem.

The CBDUMP command can be placed in the command stream to turn tracing on and off. In the above example, CBDUMP TRACE could be placed before the failing command and CBDUMP NOTRACE immediately after. A trace will contain only the activity generated by that OGL/370 command.

Output from the CBDUMP Command

Samples of the actual output generated by the CBDUMP command are shown on the following pages. Some of the samples have been abridged to remove large volumes of similar material. This is indicated by three dots (...).

A SOURCE listing consists of:

1. OGL/370 commands, user-supplied parameters, and inline generated messages
2. The CBDUMP command specifying:
 - Trace option
 - Entry of each module visited
 - Exit of each module visited with its associated exit return code.
3. Control-block information

Information is gathered, formatted, and printed for the following OGL/370 control blocks:

- Master control block
- Font control block
- Segment control block
- Group control block
- Line control block
- Unoptimized raster-pattern control block
- Optimized raster-pattern control block
- Text control block.

A final disposition report is then printed, giving the resultant status of the overlay.

```

*****
*
PROGRAM INVOCATION:

OVERLAY DGCBDUMP EXAMPLE A1 ( SYMBOLIC ( OPALR20 SYMBOLIC
*****
*

- OVERLAY GENERATION LANGUAGE 370 - R1.00 -
----- TIME 15:18 ---- DATE 90.186 1990-07-05 ---- PAGE 1

LINE          SOURCE      INPUT      STATEMENTS
NUM.  -----1-----2-----3-----4-----5-----6-----7--

0001
0002      - '*****'
0003      - '*'
0004      - '*' OVERLAY GENERATION LANGUAGE / 370 (5688-191) '*'
0005      - '*' EXAMPLE NAME: CBDUMP EXAMPLE '*'
0006      - '*' (C) COPYRIGHT BY IBM 1990 '*'
0007      - '*'
0008      - '*****'
0009      - 'GETTING STARTED'
0010      CONTROL REPLACE ALL;
0011      SETUNITS .25 IN .25 IN LINESP .15 IN;
0012      OVERLAY DEM01 SIZE 29 41 OFFSET 0.75 IN 0.5 IN;
0013      ORIENT 0;
0014
0015      - 'FONTS AND SEGMENTS'
0016      FONT FONT1 BRTR FILETYPE FONT3820;
0017      FONT FONT2 B1TR FILETYPE FONT3820;
0018      FONT FONT3 GT10 FILETYPE FONT3820;
0019      FONT FONT4 GT12 FILETYPE FONT3820;
0020      FONT FONT5 GT15 FILETYPE FONT3820;
0021      SEGMENT PALM PALM2;
0022
0023      -LINES
0024      POSITION ABSOLUTE 1 ABSOLUTE 10;
0025      DRAWRULE ACROSS 14 MEDIUM SOLID;
0026      POSITION ABSOLUTE 9 ABSOLUTE 9;
0027      DRAWRULE DOWN 1.5 BOLD SOLID
0028      REPEAT LOCATION 14 9;
0029
0030      POSITION ABSOLUTE 16 ABSOLUTE 10;
0031      DRAWRULE ACROSS 11 MEDIUM SOLID;
0032      POSITION ABSOLUTE 17 ABSOLUTE 9;
0033      DRAWRULE DOWN 1 MEDIUM SOLID

```

Figure 8 (Part 1 of 2). An Example of a Source Listing

"Restricted Materials of IBM"
 Licensed Materials – Property of IBM

- O V E R L A Y G E N E R A T I O N L A N G U A G E 3 7 0 - R 1 . 0 0 -
 DEM01----- TIME 15:18 ---- DATE 90.186 1990-07-05 ---- PAGE 2

LINE NUM.	SOURCE	INPUT	STATEMENTS
	1	2	3
0034	REPEAT ACROSS 9 SPACED 1;		
0035	POSITION ABSOLUTE 19.25 ABSOLUTE 9.5;		
0036	DRAWRULE ACROSS .5 MEDIUM SOLID		
0037	REPEAT ACROSS 1 SPACED 2.5;		
0038			
0039	POSITION ABSOLUTE 1 ABSOLUTE 12;		
0040	DRAWRULE ACROSS 17 MEDIUM SOLID;		-ADDRESS
0041			
0042	POSITION ABSOLUTE 19 ABSOLUTE 12;		
0043	DRAWRULE ACROSS 5 MEDIUM SOLID		-CITY
0044	REPEAT LOCATION 1 14		- 'ZIP CODE '
0045	LOCATION 16 14		- 'HOME PHONE'
0046	LOCATION 22 14;		- 'WORK PHONE'
0047			
0048	POSITION ABSOLUTE 25 ABSOLUTE 12;		
0049	DRAWRULE ACROSS 2 MEDIUM SOLID;		-STATE
0050			
0051			
0052	- 'BOXES AND TEXT'		
0053	POSITION ABSOLUTE 0 ABSOLUTE 4;		
0054	DRAWBOX 28 24 MEDIUM SOLID;		-FRAME
0055			
0056	POSITION ABSOLUTE 1 ABSOLUTE 9;		
0057	DRAWBOX 14 1.5 MEDIUM SOLID;		-NAME
0058	POSITION RIGHT .7 IN DOWN .37 IN;		
0059	SETTEXT 0 MODERN LEFT		
0060	LINE FONT5 NOUNDERLINE CHAR 'LAST NAME';		
0061	POSITION RIGHT 1.6 IN DOWN 0;		
0062	SETTEXT 0 MODERN LEFT		
0063	LINE FONT5 NOUNDERLINE CHAR 'FIRST NAME';		
0064	POSITION RIGHT 1 IN DOWN 0;		
0065	SETTEXT 0 MODERN LEFT		
0066	LINE FONT5 NOUNDERLINE CHAR 'MI';		
0067			
0068	POSITION ABSOLUTE 16 ABSOLUTE 9;		
0069	DRAWBOX 11 1.5 MEDIUM SOLID;		- 'SOC. SEC. NO. '
0070	POSITION RIGHT .6 IN DOWN .37 IN;		
0071	SETTEXT 0 MODERN LEFT		
0072	LINE FONT5 NOUNDERLINE CHAR 'SOCIAL SECURITY NUMBER';		
0073			
0074	POSITION ABSOLUTE 1 ABSOLUTE 11;		
0075	DRAWBOX 17 1.5 MEDIUM SOLID;		-ADDRESS
0076	POSITION RIGHT .9 IN DOWN .37 IN;		

Figure 8 (Part 2 of 2). An Example of a Source Listing

Logic Dump

Starting at line 112 of the source-input statements is a list of the modules entered and exited, the module descriptions, and the return codes.

LINE NUM.	SOURCE	INPUT	STATEMENTS
	1	2	3
0077	SETTEXT 0 MODERN LEFT		
0078	LINE FONT5 NOUNDERLINE CHAR 'STREET ADDRESS OR '		
0079	FONT5 NOUNDERLINE CHAR 'BOX NUMBER';		
0080			
0081	POSITION ABSOLUTE 19 ABSOLUTE 11;		
0082	DRAWBOX 5 1.5 MEDIUM SOLID		-CITY
0083	REPEAT LOCATION 1 13		- 'ZIP CODE'
0084	LOCATION 16 13		- 'HOME PHONE'
0085	LOCATION 22 13;		- 'WORK PHONE'
0086	POSITION RIGHT 2 DOWN .37 IN;		
0087	SETTEXT 0 MODERN LEFT		-CITY
0088	LINE FONT5 NOUNDERLINE CHAR 'CITY';		
0089	POSITION ABSOLUTE .6 IN ABSOLUTE 3.62 IN;		
0090	SETTEXT 0 MODERN LEFT		- 'ZIP CODE'
0091	LINE FONT5 NOUNDERLINE CHAR 'ZIP CODE';		
0092	POSITION ABSOLUTE 4.3 IN ABSOLUTE 3.62 IN;		
0093	SETTEXT 0 MODERN LEFT		- 'HOME PHONE'
0094	LINE FONT5 NOUNDERLINE CHAR 'HOME PHONE';		
0095	POSITION ABSOLUTE 5.8 IN ABSOLUTE 3.62 IN;		- 'WORK PHONE'
0096	SETTEXT 0 MODERN LEFT		
0097	LINE FONT5 NOUNDERLINE CHAR 'WORK PHONE';		
0098			
0099	POSITION ABSOLUTE 25 ABSOLUTE 11;		
0100	DRAWBOX 2 1.5 MEDIUM SOLID;		-STATE
0101	POSITION RIGHT .09 IN DOWN .37 IN;		
0102	SETTEXT 0 MODERN LEFT		
0103	LINE FONT5 NOUNDERLINE CHAR 'STATE';		
0104			
0105	POSITION ABSOLUTE 23 ABSOLUTE 15;		
0106	DRAWBOX 1.02 IN 8 MEDIUM SOLID;		-RESIDENCE
0107	POSITION RIGHT .1 IN DOWN .2 IN;		
0108	SETTEXT 0 MODERN CENTER SPACED 1		
0109	LINE FONT4 NOUNDERLINE CHAR 'LEGAL'		
0110	LINE FONT4 NOUNDERLINE CHAR 'RESIDENCE';		
0111	POSITION LEFT .1 IN DOWN .2 IN;		
0112	CBDDUMP TRACEALL;		
***	FLOW TRACE ***	DZILKTRC EXIT	RC = 0000
***	FLOW TRACE ***	DZILTOK NTRYTOKENIZER	
***	FLOW TRACE ***	DZILRFCL NTRYFREE MEMORY CELL	
***	FLOW TRACE ***	DZILSFMN NTRYRELEASE MEMORY	
***	FLOW TRACE ***	DZILSVFM NTRYFREEMAIN ROUTINE	
***	FLOW TRACE ***	DZILSVFM EXIT	RC = 0000
***	FLOW TRACE ***	DZILSFMN EXIT	RC = 0000
***	FLOW TRACE ***	DZILRFCL EXIT	RC = 0000
***	FLOW TRACE ***	DZILRGCL NTRYGET MEMORY CELL	
***	FLOW TRACE ***	DZILSGMN NTRYOBTAIN STORAGE	
***	FLOW TRACE ***	DZILSVGM NTRYGETMAIN ROUTINE	
***	FLOW TRACE ***	DZILSVGM EXIT	RC = 0000
***	FLOW TRACE ***	DZILSGMN EXIT	RC = 0000
***	FLOW TRACE ***	DZILRGCL EXIT	RC = 0000
***	FLOW TRACE ***	DZILSRD NTRYREAD A RECORD	
***	FLOW TRACE ***	DZILSVRD NTRYREAD REC FILE	
***	FLOW TRACE ***	DZILSVRD EXIT	RC = 0000
***	FLOW TRACE ***	DZILSRD EXIT	RC = 0000
***	FLOW TRACE ***	DZILSWRT NTRYWRITE A RECORD	
***	FLOW TRACE ***	DZILSVWR NTRYWRT REC TO FILE	

Figure 9 (Part 1 of 2). An Example of a Logic Dump

```

0113          DRAWRULE ACROSS 1.02 IN MEDIUM SOLID;
*** FLOW TRACE *** DZILSVWR EXIT          RC = 0000
*** FLOW TRACE *** DZILSWRT EXIT          RC = 0000
*** FLOW TRACE *** DZILTKO  EXIT          RC = 0000
*** FLOW TRACE *** DZILKDRL NTRYDRAWRULE CMD PRCR
*** FLOW TRACE *** DZILSYNT NTRYSYNTAX PR CNTL MOD
*** FLOW TRACE *** DZILSYNB NTRYSYNTAX CNTL BLK
*** FLOW TRACE *** DZILRGCL NTRYGET MEMORY CELL
*** FLOW TRACE *** DZILRGCL EXIT          RC = 0000
*** FLOW TRACE *** DZILRGCL NTRYGET MEMORY CELL
*** FLOW TRACE *** DZILRGCL EXIT          RC = 0000
*** FLOW TRACE *** DZILRGCL NTRYGET MEMORY CELL
*** FLOW TRACE *** DZILRGCL EXIT          RC = 0000
*** FLOW TRACE *** DZILSYNB NTRYSYNTAX CNTL BLK
*** FLOW TRACE *** DZILRGCL NTRYGET MEMORY CELL
*** FLOW TRACE *** DZILRGCL EXIT          RC = 0000
*** FLOW TRACE *** DZILRGCL NTRYGET MEMORY CELL
*** FLOW TRACE *** DZILRGCL EXIT          RC = 0000
*** FLOW TRACE *** DZILRFCL NTRYFREE MEMORY CELL
*** FLOW TRACE *** DZILRFCL EXIT          RC = 0000
*** FLOW TRACE *** DZILSYNB EXIT          RC = 0000
.
.
.
0114          POSITION RIGHT .05 IN DOWN .3 IN;
*** FLOW TRACE *** DZILSVWR EXIT          RC = 0000
.
.
.

```

Figure 9 (Part 2 of 2). An Example of a Logic Dump

Master Control Block

- O V E R L A Y G E N E R A T I O N L A N G U A G E 3 7 0 - R 1 . 0 0 -
DEM01----- TIME 15:18 ---- DATE 90.186 1990-07-05 ---- PAGE 11

```
*****
**          MASTER CONTROL BLOCK INFORMATION          **
*****
OVERLAY ID  XSIZE YSIZE  XORIG YORIG  ORIENT   DATE    TIME
DEM01      01740 02460   00180 00120   000     90-07-05  15:18
```

Figure 10. An Example of a Master Control Block Dump

OVERLAY ID	Local name of overlay.
XSIZE	Overlay horizontal size, in pels.
YSIZE	Overlay vertical size, in pels.
XORIG	Overlay horizontal origin, in pels, from the top left corner of the paper.
YORIG	Overlay vertical origin, in pels, from the top left corner of the paper.
ORIENT	Overlay orientation.
DATE	Date in the form yy-mm-dd.
TIME	Time-of-day in the form hh:mm.

Font Control Block

- OVERLAY GENERATION LANGUAGE 370 - R1.00 -
DEM01----- TIME 15:18 ---- DATE 90.186 1990-07-05 ---- PAGE 12

```
*****
**          FONT CONTROL BLOCK INFORMATION          **
*****
FCB#  LOGO  ID NAME  MEMBER  LINKED TO  FORMAT  ORIENT  UNIF  DOUB  UNDER  UWID  ULOC  ASCDR  DESDR  DBI  INCR  VSI

0001  FCB   FONT1    X1BRTR  FONT2    MODERN    0      NO    NO    NO    NO    003  003  030  010  040  028  020
        X2BRTR    MODERN    90      NO    NO    003  003  030  010  040  028  020
        X3BRTR    MODERN   180      NO    NO    003  003  030  010  040  028  020
        X4BRTR    MODERN   270      NO    NO    003  003  030  010  040  028  020
        XEBRTR    COLUMN    0      YES    NO    003  003  014  012  040  040  020
        XFBRTR    COLUMN    90      YES    NO    003  003  014  012  040  040  020
        XGBRTR    COLUMN   180      YES    NO    003  003  014  012  040  040  020
        XDBRTR    COLUMN   270      YES    NO    003  003  014  012  040  040  020
0002  FCB   FONT2    X1BITR  FONT3    MODERN    0      NO    NO    NO    003  003  030  010  040  028  020
        X2BITR    MODERN    90      NO    NO    003  003  030  010  040  028  020
        X3BITR    MODERN   180      NO    NO    003  003  030  010  040  028  020
        X4BITR    MODERN   270      NO    NO    003  003  030  010  040  028  020
        XEBITR    COLUMN    0      YES    NO    003  003  014  013  040  040  020
        XFBITR    COLUMN    90      YES    NO    003  003  014  013  040  040  020
        XGBITR    COLUMN   180      YES    NO    003  003  014  013  040  040  020
        XDBITR    COLUMN   270      YES    NO    003  003  014  013  040  040  020
0003  FCB   FONT3    X1GT10  FONT4    MODERN    0      NO    NO    NO    003  003  030  010  040  024  024
        X2GT10    MODERN    90      NO    NO    003  003  030  010  040  024  024
        X3GT10    MODERN   180      NO    NO    003  003  030  010  040  024  024
        X4GT10    MODERN   270      NO    NO    003  003  030  010  040  024  024
        XEGT10    COLUMN    0      YES    NO    003  003  013  011  040  040  024
        XFGT10    COLUMN    90      YES    NO    003  003  013  011  040  040  024
        XGGT10    COLUMN   180      YES    NO    003  003  013  011  040  040  024
        XDGT10    COLUMN   270      YES    NO    003  003  013  011  040  040  024
0004  FCB   FONT4    X1GT12  FONT5    MODERN    0      NO    NO    NO    003  003  024  006  030  024  020
        X2GT12    MODERN    90      NO    NO    003  003  024  006  030  024  020
        X3GT12    MODERN   180      NO    NO    003  003  024  006  030  024  020
        X4GT12    MODERN   270      NO    NO    003  003  024  006  030  024  020
        XEGT12    COLUMN    0      YES    NO    003  003  011  009  030  030  020
        XFGT12    COLUMN    90      YES    NO    003  003  011  009  030  030  020
        XGGT12    COLUMN   180      YES    NO    003  003  011  009  030  030  020
        XDGT12    COLUMN   270      YES    NO    003  003  011  009  030  030  020
0005  FCB   FONT5    X1GT15    MODERN    0      NO    NO    NO    003  003  024  006  030  024  016
        X2GT15    MODERN    90      NO    NO    003  003  024  006  030  024  016
        X3GT15    MODERN   180      NO    NO    003  003  024  006  030  024  016
        X4GT15    MODERN   270      NO    NO    003  003  024  006  030  024  016
        XEGT15    COLUMN    0      YES    NO    003  003  009  007  030  030  016
        XFGT15    COLUMN    90      YES    NO    003  003  009  007  030  030  016
        XGGT15    COLUMN   180      YES    NO    003  003  009  007  030  030  016
        XDGT15    COLUMN   270      YES    NO    003  003  009  007  030  030  016
```

Figure 11. An Example of a Font Control Block Dump

FCB#	FCB identification number.
LOGO	Logo to identify FCBs in a dump.
ID NAME	Font name as specified by the user.
MEMBER	Coded font member name.
LINKED TO	Font name for the next font on the FCB chain.
FORMAT	Format of the character progression.
ORIENT	Character orientation in degrees.
UNIF	Does the font have a uniform character increment?
DOUB	Is this a double-byte font?
UNDER	Does the font include underline in the character box?
UWID	Width of the underline ² .
ULOC	Distance below baseline for positioning the underline ² .
ASCDR	Maximum ascender from baseline.
DESDR	Maximum descender from baseline.
DBI	Default baseline increment.
INCR	Uniform character increment.
VSI	Variable space increment.

² This information is used for underlining text that does not contain the underline in the raster pattern. Although a font may include this information, OGL/370 does not allow underscore in the COLUMN or TATE formats.

Segment Control Block

```
- O V E R L A Y   G E N E R A T I O N   L A N G U A G E   3 7 0 - R 1 . 0 0 -  
DEM01----- TIME 15:18 ---- DATE 90.186 1990-07-05 ---- PAGE 13  
  
*****  
**          SEGMENT CONTROL BLOCK INFORMATION          **  
*****  
GCB#  LOGO  ID NAME  MEMBER  LINKED TO  #PLACINGS  XORIG  YORIG  
  
0001  GCB   PALM    S1PALM2      0001    00990  00258
```

Figure 12. An Example of a Segment Control Block Dump

GCB#	Segment identification number.
LOGO	Logo to identify GCBs in a dump.
ID NAME	Segment name as specified by the user.
MEMBER	Library member name.
LINKED TO	Local ID for the next segment on the GCB chain.
#PLACINGS	Number of times the segment was placed by the user.
XORIG	Horizontal coordinate of the segment, in pels.
YORIG	Vertical coordinate of the segment, in pels.

Group Control Block

- O V E R L A Y G E N E R A T I O N L A N G U A G E 3 7 0 - R 1.00 -
DEM01----- TIME 15:18 ---- DATE 90.186 1990-07-05 ---- PAGE 14

```
*****
**          GROUP CONTROL BLOCK INFORMATION          **
*****
KCB#  LOGO  ID NAME  LINKED TO  #COMMANDS  LINE#  COMMAND

0001  KCB   SCHED           0037    00169  SETUNITS
                                00172  POSITION
                                00173  SETTEXT
                                00176  POSITION
                                00177  SETTEXT
                                00181  POSITION
                                00182  DRAWBOX
                                00183  POSITION
                                00184  SETTEXT
                                00187  POSITION
                                00188  SETTEXT
                                00191  POSITION
                                00192  SETTEXT
                                00194  POSITION
                                00195  SETTEXT
                                00197  POSITION
                                00198  SETTEXT
                                00200  POSITION
                                00201  DRAWBOX
                                00215  POSITION
                                00216  DRAWRULE
                                00217  POSITION
                                00218  DRAWRULE
                                00221  POSITION
                                00222  DRAWRULE
                                00227  POSITION
                                00228  DRAWRULE
                                00232  POSITION
                                00233  DRAWBOX
                                00235  POSITION
                                00236  DRAWRULE
                                00238  POSITION
                                00239  SETTEXT
                                00241  POSITION
                                00242  SETTEXT
                                00244  POSITION
                                00245  SETTEXT
```

Figure 13. An Example of a Group Control Block Dump

KCB#	Group identification number.
LOGO	Logo to identify group control blocks in a dump.
ID NAME	Group name as specified by the user.
LINKED TO	Group name for the next group on the KCB chain.
#COMMANDS	Number of commands in the group definition.
LINE#	Source-input-statement line numbers of commands in the group definition.
COMMAND	Commands specified in the group definition.

Line Control Block

- OVERLAY GENERATION LANGUAGE 370 - R1.00 -
 DEMO1----- TIME 15:18 ---- DATE 90.186 1990-07-05 ---- PAGE 15

LINE CONTROL BLOCK INFORMATION											
LCB#	LOGO	LINE#	XSIZE	YSIZE	DIRECT	TYPE	XORIG	YORIG	XCELL	YCELL	RASTER IMAGE
0001	LCB	00025	00840	00004	ACROSS	SOLID	00060	00600	N/A	N/A	N/A
0002	LCB	00027	00006	00090	DOWN	SOLID	00540	00540	N/A	N/A	N/A
0003	LCB	00027	00006	00090	DOWN	SOLID	00840	00540	N/A	N/A	N/A
0004	LCB	00031	00660	00004	ACROSS	SOLID	00960	00600	N/A	N/A	N/A
0005	LCB	00033	00004	00060	DOWN	SOLID	01020	00540	N/A	N/A	N/A
0006	LCB	00033	00004	00060	DOWN	SOLID	01080	00540	N/A	N/A	N/A
0007	LCB	00033	00004	00060	DOWN	SOLID	01140	00540	N/A	N/A	N/A
0008	LCB	00033	00004	00060	DOWN	SOLID	01200	00540	N/A	N/A	N/A
0009	LCB	00033	00004	00060	DOWN	SOLID	01260	00540	N/A	N/A	N/A
0010	LCB	00033	00004	00060	DOWN	SOLID	01320	00540	N/A	N/A	N/A
0011	LCB	00033	00004	00060	DOWN	SOLID	01380	00540	N/A	N/A	N/A
0012	LCB	00033	00004	00060	DOWN	SOLID	01440	00540	N/A	N/A	N/A
0013	LCB	00033	00004	00060	DOWN	SOLID	01500	00540	N/A	N/A	N/A
0014	LCB	00033	00004	00060	DOWN	SOLID	01560	00540	N/A	N/A	N/A
0015	LCB	00036	00030	00004	ACROSS	SOLID	01155	00570	N/A	N/A	N/A
0016	LCB	00036	00030	00004	ACROSS	SOLID	01335	00570	N/A	N/A	N/A
0017	LCB	00040	01020	00004	ACROSS	SOLID	00060	00720	N/A	N/A	N/A
0018	LCB	00043	00300	00004	ACROSS	SOLID	01140	00720	N/A	N/A	N/A
0019	LCB	00043	00300	00004	ACROSS	SOLID	00060	00840	N/A	N/A	N/A
.										N/A	N/A
.										N/A	N/A
.										N/A	N/A
0055	LCB	00100	00124	00004	ACROSS	SOLID	01500	00660	N/A	N/A	N/A
0056	LCB	00100	00004	00094	DOWN	SOLID	01620	00660	N/A	N/A	N/A
0057	LCB	00100	00124	00004	ACROSS	SOLID	01500	00750	N/A	N/A	N/A
0058	LCB	00100	00004	00094	DOWN	SOLID	01500	00660	N/A	N/A	N/A
0059	LCB	00106	00248	00004	ACROSS	SOLID	01380	00900	N/A	N/A	N/A
0060	LCB	00106	00004	00484	DOWN	SOLID	01624	00900	N/A	N/A	N/A
0061	LCB	00106	00248	00004	ACROSS	SOLID	01380	01380	N/A	N/A	N/A
0062	LCB	00106	00004	00484	DOWN	SOLID	01380	00900	N/A	N/A	N/A
0063	LCB	00113	00244	00004	ACROSS	SOLID	01380	00996	N/A	N/A	N/A
0064	LCB	00116	00052	00004	ACROSS	SOLID	01392	01068	N/A	N/A	N/A
0065	LCB	00116	00004	00052	DOWN	SOLID	01440	01068	N/A	N/A	N/A
0066	LCB	00116	00052	00004	ACROSS	SOLID	01392	01116	N/A	N/A	N/A
0067	LCB	00116	00004	00052	DOWN	SOLID	01392	01068	N/A	N/A	N/A
0068	LCB	00116	00052	00004	ACROSS	SOLID	01392	01164	N/A	N/A	N/A
0069	LCB	00116	00004	00052	DOWN	SOLID	01440	01164	N/A	N/A	N/A
0070	LCB	00116	00052	00004	ACROSS	SOLID	01392	01212	N/A	N/A	N/A
0071	LCB	00116	00004	00052	DOWN	SOLID	01392	01164	N/A	N/A	N/A
0072	LCB	00116	00052	00004	ACROSS	SOLID	01392	01260	N/A	N/A	N/A
0073	LCB	00116	00								

Figure 14. An Example of a Line Control Block Dump

LCB#	Line-control-block number.
LOGO	Logo to identify LCBs in a dump.
LINE#	Source-input-statement line number.
XSIZE	Horizontal size of input rectangle, in pels.
YSIZE	Vertical size of input rectangle, in pels.
DIRECT	Line direction with respect to the overlay.
TYPE	Line type specified by the user (solid, dashed, dotted, or shaded).
XORIG	Horizontal coordinate of output rectangle, in pels.
YORIG	Vertical coordinate of output rectangle, in pels.
XCELL	Horizontal size of repeat image cell, in pels.
YCELL	Vertical size of repeat image cell, in pels.
RASTER IMAGE	Hexadecimal representation of the raster image in relationship to the top of the printed page.

Unoptimized Raster Pattern Control Block

- O V E R L A Y G E N E R A T I O N L A N G U A G E 3 7 0 - R 1 . 0 0 -
DEM01----- TIME 15:18 ---- DATE 90.186 1990-07-05 ---- PAGE 16

[illegible]

Figure 15. An Example of an Unoptimized Raster Pattern Control Block Dump

RCB#	Raster-pattern identification number.
LOGO	Logo to identify RCBs in a dump.
ID NAME	Pattern name. May be a name defined by the user, or a name beginning with '\$' for OGL/370-generated patterns. Appendix D, "OGL/370 Storage Requirements" on page 77 contains more information about OGL/370-generated patterns.
LINKED TO	Pattern name for the next raster pattern on the RCB chain.
XSIZE	Horizontal size of the raster pattern, rounded up to the nearest 8-pel boundary. ³
YSIZE	Vertical size of the raster pattern, rounded up to the nearest 8-pel boundary. ³
XACT	Horizontal size of the raster pattern.
YACT	Vertical size of the raster pattern.
#BYTES	Total number of bytes in the pattern (including padding to 8-pel boundary).
#PLACINGS	Number of times the pattern is placed.
XORIG	Horizontal coordinate of the pattern placement, in pels.
YORIG	Vertical coordinate of the pattern placement, in pels.
ORIENT	Orientation of this placement of the pattern with respect to the overlay.
SHADE#	Shading level number (0 to 32) of this placement of the pattern.
MIRR?	Is this placement of the pattern to be mirrored?
NEG?	Is this placement of the pattern to have black and white reversed?

For each RCB#, a hexadecimal representation of the raster pattern including any padding is given. In the above example, the pattern for RCB# 001 is shown as:

0001 FFFFFFFFFFFFFFFFFFFFFFFFFFFFFF etc. ...

³ Patterns are padded with blanks to a multiple of 8 in each direction. An informational message is issued to tell the user both the specified size and the padded size.

Optimized Raster Pattern Control Block

- OVERLAY GENERATION LANGUAGE 370 - R1.00 -
DEM01----- TIME 15:18 ---- DATE 90.186 1990-07-05 ---- PAGE 17

[illegible]

Figure 16. An Example of an Optimized Raster Pattern Control Block Dump

In the example in Figure 16 on page 64, the optimizer has made no changes to the RCB chain. If the optimizer has made changes, then any patterns which have been affected will have an ID NAME of \$OPTMZER.

RCB#	Raster-pattern identification number.
LOGO	Logo to identify RCBs in a dump.
ID NAME	Pattern name. May be a name defined by the user, or a name beginning with '\$' for OGL/370-generated patterns. Appendix D, "OGL/370 Storage Requirements" on page 77 contains more information about OGL/370-generated patterns.
LINKED TO	Pattern name for the next raster pattern on the RCB chain.
XSIZE	Horizontal size of the raster pattern, rounded up to the nearest 8-pel boundary. ⁴
YSIZE	Vertical size of the raster pattern, rounded up to the nearest 8-pel boundary. ⁴
XACT	Horizontal size of the raster pattern.
YACT	Vertical size of the raster pattern.
#BYTES	Total number of bytes in the pattern (including padding to 8-pel boundary).
#PLACINGS	Number of times the pattern is placed.
XORIG	Horizontal coordinate of the pattern placement, in pels.
YORIG	Vertical coordinate of the pattern placement, in pels.
ORIENT	Orientation of this placement of the pattern with respect to the overlay.
SHADE#	Shading level number (0 to 32) of this placement of the pattern.
MIRR?	Is this placement of the pattern to be mirrored?
NEG?	Is this placement of the pattern to have black and white reversed?

For each RCB#, a hexadecimal representation of the raster pattern including any padding is given. In the above example, the pattern for RCB# 001 is shown as:

0001 FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF etc. ...

⁴ Patterns are padded with blanks to a multiple of 8 in each direction. An informational message is issued to tell the user both the specified size and the padded size.

Text Control Block

- O V E R L A Y G E N E R A T I O N L A N G U A G E 3 7 0 - R 1.00 -
DEM01----- TIME 15:18 ---- DATE 90.186 1990-07-05 ---- PAGE 18

```
*****
**          TEXT CONTROL BLOCK INFORMATION          **
*****
XCB#  LOGO  FONT MEM  LINE#  TXLEN  XORIG  YORIG  UNDER?  UWID  ULOC  INCR  ILINE  BLINE

0001  XCB   X1GT15   00059  00009  00228  00628    NO                0000  000  090
0002  XCB   X1GT15   00062  00010  00612  00628    NO                0000  000  090
0003  XCB   X1GT15   00065  00002  00852  00628    NO                0000  000  090
.
.
0064  XCB   X1BRTR   00173  00003  00120  01338    NO                0000  000  090
0065  XCB   X1BRTR   00173  00003  00180  01338    YES  0003  0003  0000  000  090
0066  XCB   X1BITR   00158  00008  01358  00300    NO                0000  000  090
0067  XCB   X1BITR   00158  00017  01260  00348    NO                0000  000  090
0068  XCB   XEGT10   00177  00008  00072  00948    NO                0000  090  180
0069  XCB   XEGT10   00177  00008  00072  01338    NO                0000  090  180

0001  'LAST NAME'
0001  X'D3C1E2E340D5C1D4C5'
0002  'FIRST NAME'
0002  X'C6C9D9E2E340D5C1D4C5'
0003  'MI'
0003  X'D4C9'
.
.
0064  '198'
0064  X'F1F9F8'
0065  ' '
0065  X'404040'
0066  'TropiCal'
0066  X'E399969789C38193'
0067  'Community College'
0067  X'C3969494A49589A3A840C3969393858785'
0068  'SEMESTER'
0068  X'E2C5D4C5E2E3C5D9'
0069  'SEMESTER'
0069  X'E2C5D4C5E2E3C5D9'
```

Figure 17. An Example of a Text Control Block Dump

XCB#	Text-array identification number.
LOGO	Logo to identify XCBs in a dump.
FONT MEM	Coded-font member name for text segment.
LINE#	Source-input-statement line number.
TXLEN	Length of text segment, in bytes.
XORIG	Horizontal coordinate of text segment.
YORIG	Vertical coordinate of text segment.
UNDER?	Is the text to be underlined?
UWID	Width of underline (if the text is to be underlined).
ULOC	Distance of underline below the baseline (if the text is to be underlined).
INCR	Additional character increment for balanced text.
ILINE	Inline orientation.
BLINE	Baseline orientation.

For each XCB#, the text is shown in the dump in both character and hexadecimal format. In the preceding example, XCB# 0001 includes these entries:

```
0001  'LAST NAME'  
0001  X'D3C1E2E340D5C1D4C5'
```

Final Disposition Report

- O V E R L A Y G E N E R A T I O N L A N G U A G E 3 7 0 - R1.00 -
DEM01----- TIME 15:18 ---- DATE 90.186 1990-07-05 ---- PAGE 20

DZI0708I FINAL DISPOSITION:

OVERLAY FILE: NOT CREATED
SAMPLE-OVERLAY FILE: CREATED AND PRINTED
IMAGE OPTIMIZATION: PERFORMED

MESSAGE SEVERITY SUMMARY:
0 INFORMATIONAL MESSAGES WERE SUPPRESSED.
3 INFORMATIONAL MESSAGES WERE PRINTED.
0 WARNING MESSAGES WERE SUPPRESSED.
0 WARNING MESSAGES WERE PRINTED.
0 ERROR MESSAGES WERE PRINTED.

FINAL RETURN CODE: 0

*** FLOW TRACE *** DZILMSG EXIT RC = 0000
*** FLOW TRACE *** DZILMSG NTRYMESSAGE ID: 702

----- END OVERLAY GENERATION LANGUAGE SOURCE LISTING -----

*** FLOW TRACE *** DZILMSG EXIT RC = 0000
*** FLOW TRACE *** DZILSCLS NTRYCLOSE A FILE
*** FLOW TRACE *** DZILSVCL NTRYCLOSE FILE ROUTINE

Figure 18. An Example of a Final Disposition Report (VM)

Appendix A. Message-to-Module Cross-Reference

Table 4 lists, by identification number, all the messages issued by OGL/370 with the module or modules that issue the message.

Table 4 (Page 1 of 4). Message-to-Module Cross-Reference

Message Number	Module Name(s)
DZI0101I	DZILSYNT
DZI0102I	DZILSYNT
DZI0103I	DZILSYNT
DZI0104I	DZILSYNB
DZI0106I	DZILSYNB
DZI0107I	DZILTOK
DZI0109I	DZILTOK
DZI0111I	DZILTOK
DZI0112I	DZILTOK
DZI0201I	DZILSCTL
DZI0202I	DZILSCTL
DZI0203I	DZILSCTL
DZI0204I	DZILSCTL
DZI0205I	DZILSCTL
DZI0206I	DZILSCTL
DZI0301I	DZILSMOP
DZI0302I	DZILSD2C DZILSM2C DZILSV2C
DZI0303I	DZILSMOP
DZI0304I	DZILSMOP
DZI0305I	DZILSMOP
DZI0306I	DZILSD2C DZILSM2C DZILSV2C
DZI0307I	DZILSMOP
DZI0309I	DZILSMOP
DZI0310I	DZILSMOP
DZI0311I	DZILSMOP
DZI0312I	DZIABEND
DZI0315I	DZILSMKS
DZI0316I	DZILPARS
DZI0317I	DZILPARS
DZI0318I	DZILPARS
DZI0319I	DZILSCTL
DZI0335I	DZILSVKS DZILSVOP
DZI0336I	DZILSVOP
DZI0337I	DZILSVCL
DZI0338I	DZILSVCL
DZI0341I	DZILSVWR
DZI0342I	DZILSVOP DZILSVRD DZILSVWR
DZI0343I	DZILSVWR

Table 4 (Page 2 of 4). Message-to-Module Cross-Reference

Message Number	Module Name(s)
DZI0345I	DZILSVOP
DZI0346I	DZILSVOP
DZI0347I	DZILSVOP
DZI0348I	DZILSVCL
DZI0349I	DZILSVOP
DZI0353I	DZILSDKS
DZI0354I	DZILSDSY
DZI0355I	DZILSDCL DZILSDOP DZILSDRD DZILSDWR
DZI0356I	DZILSDOP
DZI0357I	DZILSDOP
DZI0358I	DZILSDOP
DZI0359I	DZILSDOP
DZI0360I	DZILSDOP
DZI0361I	DZILSDOP
DZI0372I	DZILSVPR
DZI0373I	DZILSVPR
DZI0374I	DZILSVPR
DZI0375I	DZILSVPR
DZI0376I	DZILSVPR
DZI0377I	DZILSVPR
DZI0378I	DZILSVPR
DZI0379I	DZILSVOP
DZI0380I	DZILSVPR
DZI0381I	DZILSVPR
DZI0382I	DZILSVPR
DZI0383I	DZILSVPR
DZI0384I	DZILSVPR
DZI0385I	DZILSVPR
DZI0401I	DZILCONP
DZI0402I	DZILRDPT
DZI0403I	DZILCONP
DZI0404I	DZILRDPT
DZI0405I	DZILRDPT
DZI0406I	DZILKDRL
DZI0407I	DZILKDRL
DZI0408I	DZILRSRC
DZI0409I	DZILRDBX DZILRDGL
DZI0410I	DZILRSRC
DZI0415I	DZILRDPT
DZI0416I	DZILRDBX
DZI0417I	DZILRDBX
DZI0418I	DZILRSRC
DZI0419I	DZILKDRL
DZI0420I	DZILRSRC

Table 4 (Page 3 of 4). Message-to-Module Cross-Reference

Message Number	Module Name(s)
DZi0425i	DZILRSRC
DZi0430i	DZILRTCN
DZi0435i	DZILRSRC
DZi0440i	DZILRTCN
DZi0451i	DZILRDBX
DZi0453i	DZILRSRDR
DZi0455i	DZILKTXt DZILRWtX
DZi0456i	DZILKTXt DZILRWtX
DZi0457i	DZILKTXt DZILRWtX
DZi0460i	DZILRDCR
DZi0461i	DZILKDCR
DZi0464i	DZILKDCR
DZi0468i	DZILRDCR
DZi0469i	DZILRDCR
DZi0501i	DZILKORN DZILKPLC DZILKTXt DZILRWtX
DZi0502i	DZILKDEF DZILRDPT DZILSDKF DZILSDKS DZILSMKF DZILSMKS DZILSVKF DZILSVKS
DZi0503i	DZILKDMK
DZi0504i	DZILKDEF
DZi0505i	DZILKDEF
DZi0506i	DZILKEND
DZi0507i	DZILKOVl
DZi0508i	DZILKPOS
DZi0509i	DZILKPLC
DZi0510i	DZILKDBX DZILKDRL DZILREPH DZILKDCR DZILRRPH DZILRSRDR
DZi0511i	DZILKDEF
DZi0512i	DZILRSRC
DZi0513i	DZILRSFT DZILRWtX
DZi0514i	DZILRSRDR DZILRSHD DZILRRPH
DZi0519i	DZILSMKF
DZi0520i	DZILKDBX DZILKDCR
DZi0522i	DZILRSRC
DZi0523i	DZILKPLC
DZi0524i	DZILKPLC
DZi0525i	DZILSDKF DZILSMKF DZILSVKF
DZi0526i	DZILSDKF DZILSMKF DZILSVKF
DZi0528i	DZILRSRC
DZi0529i	DZILKDBX DZILKDCR
DZi0530i	DZILRTCN
DZi0531i	DZILRTC2
DZi0532i	DZILRTCN
DZi0533i	DZILRSFT
DZi0534i	DZILRTCN
DZi0535i	DZILRTCN

Table 4 (Page 4 of 4). Message-to-Module Cross-Reference

Message Number	Module Name(s)
DZI0536I	DZILRSFT
DZI0537I	DZILRTC2
DZI0538I	DZILRWTX
DZI0540I	DZILRTCN
DZI0542I	DZILRTCN
DZI0543I	DZILRSFT
DZI0544I	DZILRTCN
DZI0545I	DZILRTCN
DZI0546I	DZILRSFT
DZI0548I	DZILKTXI
DZI0559I	DZILSDKF DZILSMKF DZILSVKF
DZI0560I	DZILSD2C DZILSM2C DZILSV2C
DZI0561I	DZILKTXI DZILRWTX
DZI0562I	DZILKTXI DZILRWTX
DZI0563I	DZILKTXI DZILRWTX
DZI0580I	DZILREPH
DZI0582I	DZILREPH
DZI0584I	DZILREPH
DZI0586I	DZILKDPH
DZI0587I	DZILKDPH
DZI0588I	DZILKDPH
DZI0589I	DZILKDPH
DZI0601I	DZILSD2C DZILSM2C DZILSV2C
DZI0703I	DZILSCTL
DZI0708I	DZILSCTL
DZI0996I	DZILMSG
DZI0997I	DZILMSGI
DZI0998I	DZILMSGI

Appendix B. Listing an Overlay under MVS

Using IDCAMS, the following example prints an individual overlay in hexadecimal format.

```
//PRINT      EXEC PGM=IDCAMS,REGION=350K
//SYSPRINT   DD   SYSOUT=A
//DDXXX      DD   DSN=OVERLIB.NAME(01YYYYYY),DISP=SHR
//SYSIN      DD   *
              PRINT INFILE(DDXXX)

//*****
//*
//* OVERLIB.NAME  The name of the overlay library
//* 01YYYYYY     The member name of the overlay as
//*              specified in the OVERLAY command
//*
//*****
```

Appendix C. Listing an Overlay under VSE

This appendix lists the VSE/SP job control statements (JCS) necessary to print an overlay member in hexadecimal.

```
// JOB      LISTMEM
// EXEC     LIBR
        ACCESS  SUBLIB=lib.sublib
        LIST 01nnnnnn.PHASE
/*
/ &
```

where:

lib is the overlay library ID

sublib is the sublibrary name

nnnnnn is the overlay member name

PHASE is the member type (The member type must be "PHASE.")

Appendix D. OGL/370 Storage Requirements

This appendix may help you to estimate OGL/370's virtual storage requirements.

Because the logic used by OGL/370 to process overlays is complex, it is not possible here to give an exact method of predicting the requirements, but it is possible to give broad guidelines. The guidelines given here are conservative in almost all cases. Exceptional overlays — for instance an overlay containing a very large number of commands but a small amount of raster image data — may not fit the guidelines.

OGL/370 requires 720k of memory to process the simplest overlay. It needs more memory to process more complex overlays.

Many factors in the overlay definition influence the amount of storage OGL/370 needs. Generally the one which overshadows the others is the amount of raster image data in the overlay.

Each object you specify in an overlay definition — apart from solid horizontal and vertical rules, and pieces of text — is a raster pattern or is composed with smaller raster patterns. OGL/370 needs about 10k of storage per square inch of raster pattern.

You can calculate OGL/370's approximate storage requirement for most overlays as:

$$720k + 10k * (\text{area of raster patterns in square inches})$$

The next section gives you more detailed guidance about calculating the area of the raster patterns on your overlay.

Area of Raster Patterns

Raster patterns are always rectangular, and the amount of storage for a given pattern is dependent on the area of the pattern rectangle. The pattern rectangle is the smallest rectangle (with vertical sides) you can draw around the pattern that entirely encloses the pattern.

The following items are *not* raster patterns:

- Pieces of text
- Solid horizontal or vertical rules — including solid vertical or horizontal path segments and the straight segments of solid box borders
- Page segments.

The following items are all generated as raster patterns by OGL/370:

- Images defined via DEFINE PATTERN.
- Dotted and dashed lines: OGL/370 creates a raster pattern for any dotted or dashed line — a box side or diagonal, a path segment, or a circle diagonal. This includes vertical and horizontal dotted and dashed lines.

- Oblique lines: OGL/370 creates a raster pattern for each box diagonal and circle diagonal and for each segment of a path which is neither vertical nor horizontal.
- Circles and partial circles: OGL/370 creates a raster pattern for each circle or partial circle for each rounded box corner.
- Path connections: Each miter, bevel or rounded connection is a raster pattern; with the exception of a right-angle miter connection between a solid vertical and a solid horizontal rule.
- Shaded areas: OGL/370 creates a raster pattern for the shaded area of a figure:
 - One raster pattern is created for a shaded path.
 - For square boxes with no diagonals, no raster pattern is generated. OGL/370 uses a printer feature to generate shading for these boxes.
 - For a rounded-corner box without diagonals, one raster pattern the size of the corner is created for each rounded corner. Again, a printer feature is used to print the rectangular areas of the boxes between the corners.
 - For a box with diagonals, one raster image is normally created for each area of the box. Each raster image is the size of the box. If SHADE WHOLE is specified in a box, the shade areas are treated the same way as a box without diagonals.
 - For a whole shaded circle without diagonals, or for a shaded partial circle, one raster pattern is created for each quadrant.
 - A circle with diagonals is treated in the same way as a box with diagonals.

To calculate the area of raster patterns in order to estimate OGL/370's virtual memory requirements, you should:

1. Identify all the raster patterns on the overlay.
2. Calculate the area of each raster pattern.
3. Add all the areas together. It makes no difference if two raster patterns overlap; OGL/370 will need the same amount of virtual storage whether they overlap or not.

Appendix E. National Language Support

OGL/370 can generate messages in three languages (English, German and Japanese). The default and alternate languages are defined at installation. However, you can change these definitions at any time, providing the appropriate language modules have been installed.

Notes:

1. If the appropriate language modules have not been installed, you must reinstall OGL/370 from scratch.
2. The language codes, which are the same in all systems, are:
 ENGLISH
 GERMAN
 JAPANESE.

Defining the Languages in MVS

To change the default and alternate languages in MVS, perform the following steps:

1. Modify the source of the sample usermod (supplied on the installation tape), in member UMODSRC of hvr100.install.cntl, as required. The relevant statement is:

```
DZILPARM CSECT
          DZIMPARM DEFLANG=def,ALTLANG=alt
          END
```

2. Receive the usermod. A sample job is in member UMODRCV of hvr100.install.cntl. The contents of this member are:

```
//UMODRCV JOB (account info),'RECEIVE OGL0001'
//*-----
//* RECEIVE OGL0001 USERMOD
//*-----
//SMPE EXEC SMPPROC
//SMPPTFIN DD DSN=hvr100.install.cntl(UMODSRC),DISP=SHR
//SMPCTL DD *
          SET BDY(GLOBAL).
          REJECT SELECT(OGL0001) BYPASS(APPLYCHECK).
          RESETRC.
          RECEIVE SELECT(OGL0001) SYSMODS LIST.
/*
```

Note: The REJECT and RESETRC statements are included in order to allow reprocessing of the usermod at any time. The first time that the usermod is received, an error message and a return code of 12 (hexadecimal '0C') may be expected in response to the REJECT. Any subsequent runs of the job should give a return code of 0 from the REJECT. A return code of 0 should always be received from the RESETRC and RECEIVE commands.

Remember to change the usermod name from OGL0001 in the above RECEIVE and the following APPLY if you changed it in the ++USERMOD statement in the usermod source.

3. Apply the usermod. A sample job is in member UMODAPP of hvrl100.install.cntl. The contents of this member are:

```
//UMODAPP JOB (account info),'APPLY OGL0001'  
/*-----  
/*  APPLY OGL0001 USERMOD  
/*-----  
//SMPE      EXEC SMPPROC  
//SYSLIB    DD DSN=SYS1.ADZISRC0,DISP=SHR  
//SMPCTL    DD *  
            SET BDY(tgtzone).  
            APPLY SELECT(OGL0001) RED0.  
/*
```

Note: The DZIMPARM macro invoked by the DZILPARM source was stored in the SMPMTS data set after the APPLY of OGL/370 and in ADZISRC0 after the ACCEPT. When the above usermod is being applied, the assembly process will require that a library in the SYSLIB concatenation contain the DZIMPARM macro. This is why the sample UMODAPP job contains a DD statement for SYSLIB pointing to SYS1.ADZISRC0.

A return code of 0 is expected from this step.

As is generally the case with usermods, an ACCEPT should **not** be performed.

Defining the Languages in VM

To change the default and alternate languages in VM, perform the following steps:

1. Edit the file DZILPARM ASSEMBLE (supplied on the installation tape) and change the languages as necessary:

```
DZILPARM DEFLANG=def,                                     +  
            ALTLANG=alt  
END
```

2. Perform the assembly by:

```
GLOBAL MACLIB OGLMAC  
ASSEMBLE DZILPARM
```

Check that the return code was zero. If it is not, check that you have specified the languages correctly and re-assemble.

3. Place the resultant DZILPARM TEXT file on the same minidisk as the OVERLAY module, as this is required during OVERLAY processing.

Defining the Languages in VSE

To change the default and alternate languages in VSE, submit the following job:

```
* $$ JOB JNM=DZILPARM
// JOB DZILPARM < accounting information >
// DLBL PRD2,'VSE.PRD2.LIBRARY',,VSAM
// LIBDEF PROC,SEARCH=PRD2.AFP
// EXEC PROC=DZILPARM,                                +
    SEARCH='PRD2.AFP',                                +
    CATALOG='PRD2.AFP',                                +
    DEFLANG=def,                                        +
    ALTLANG=alt
/*
/&
* $$ E0J
```


Appendix F. Related Publications

The following publications, all of which are mentioned in the text of this book, contain related information.

The titles and the order numbers for IBM publications can change from time to time. To verify the current title or order number for a publication, contact your IBM marketing representative.

Table 5. Related Publications

Title	Order Number
<i>Advanced Function Printing: Data Stream Reference</i>	S544-3417
<i>Advanced Function Printing: Diagnosis Guide</i>	LH40-0201
<i>CP General User Command Reference</i>	SC19-6211
<i>IBM System/370, 30xx, 4300, and 9370 Processors Bibliography</i>	GC20-0001
<i>Overlay Generation Language/370: User's Guide and Reference</i>	S544-3702
<i>Overlay Generation Language/370: Getting Started</i>	G544-3691

Glossary

Many of these terms have meanings other than the ones presented here. The definitions in this glossary are only for those meanings related directly or indirectly to OGL/370.

advanced function printing data stream. (Also *AFP data stream* or *data stream*.) (1) Generally, the data format described in the *Advanced Function Printing: Data Stream Reference*. (2) Specifically, the data produced by OGL/370 to specify a particular overlay, which conforms to the *advanced function printing data stream* format.

AFP data stream. See *advanced function printing data stream*.

APAR. authorized program analysis report

BALANCE. An OGL/370 keyword that spaces characters evenly in a line.

baseline. An invisible line on which the characters are aligned.

character box. The boundary completely surrounding the character pattern.

COLUMN. An OGL/370 keyword that arranges the characters of a line from top to bottom and the lines from left to right.

comment. Descriptions added to an overlay definition that describe what the commands are doing but do not affect the way the overlay is printed.

coordinate. The horizontal or vertical distance from an established point to the origin of an overlay element. The established point could be the overlay origin, a group origin, or the point specified in the last POSITION command.

CPU. central processing unit

data set. A major unit of data storage and retrieval in an operating system, consisting of a collection of data in one of several prescribed arrangements and described by control information to which the system has access.

data stream. When used without qualification, *data stream* refers to the *advanced function printing data stream* produced by OGL/370. See *advanced function printing data stream*.

DBCS. See *double-byte character set*.

default. A value, keyword, or unit of measurement that is used when none is specified in a command.

definition. See *overlay definition*.

delimiter. The semi-colon (;) that must appear at the end of each command.

double-byte character set (DBCS). A character set, such as Kanji, requiring two bytes to identify each character.

electronic overlay. An overlay that is stored in a library and that can be requested for a printing job.

end marker. See *delimiter*

entry. A keyword, name, or value that is part of a command.

error message. A message saying the overlay definition contains an error that might cause a command to be ignored or the overlay not to be printed.

EWS. Early Warning System

file. A major unit of storage and retrieval in an operating system, consisting of a collection of data in one of several prescribed arrangements and described by control information to which the system has access.

font. (1) A collection of characters of a certain typeface and size. (2) Used generally, the collection of coded fonts, font character sets, and code pages.

form. (1) An overlay. (2) A physical sheet of paper on which data is printed. Synonymous with *medium*, *physical page*, *sheet*.

forms flash. An overlay in the form of a photographic negative to be flashed on the paper, as in the IBM 3800 series printers.

frame. A border around an overlay.

fullword. Four bytes of main storage in the computer.

graphic. Image, text, or a combination of both that can be placed on an overlay by name.

grid. Horizontal and vertical lines printed on an overlay to help in the design of the overlay. Synonym for *mask*.

group. A named collection of commands that can be placed on the overlay by name.

hexadecimal. A way of representing data using numbers to the base 16.

image. A pattern of toned and untoned pels that form a picture.

informational message. A message providing the user with useful information. The appearance of the overlay is not affected.

invocation. The statement that activates a command or program.

I/S. Information/System

JCL. See *Job Control Language*.

JCS. See *job control statement(s)*.

Job Control Language (JCL). A language that provides MVS with information about a job being run.

job control statement(s) (JCS). Statements that provide VSE with information about a job being run.

Kanji. The non-phonetic writing system used in Japan. Kanji characters are traditionally printed in the tate format.

keyword. (1) A word in OGL/370 that must be entered exactly as shown and that may not be used as a name for a font, segment, definition, or overlay. (2) A word that is used to describe an aspect of an OGL/370 failure to IBM.

keyword string. A sequence of keywords which is used to describe an OGL/370 failure to IBM.

library. A collection of related files. For example, font files are stored in the font library.

mask. See *grid*.

member. A file in a library. For example, font X1S0BITR is a member of the font library.

member name. The 8-character name under which a file is stored in a library. For example, X1S0BITR is the member name of a font in the font library.

member ID. The member name of a file less the first two characters. For example, S0BITR is the member ID of the font whose member name is X1S0BITR.

MODERN. An OGL/370 keyword that arranges the characters of a line from left to right and the lines from top to bottom.

Multiple Virtual Storage (MVS). An operating system.

MVS. See *Multiple Virtual Storage*.

orientation. The rotation of an overlay element relative to a fixed reference. For example, text can be rotated to 0°, 90°, 180°, or 270° relative to the top of the overlay.

origin. (1) The point in an overlay element that is used for positioning the element. (2) The upper left corner of an overlay in its 0° orientation.

overlay. A collection of predefined data — such as lines, shading, text, boxes, circles, paths, or logos — that can be merged with variable data on a sheet while printing. An overlay can be either electronic or forms flash.

overlay definition. The collection of commands that define an overlay.

page segment. A collection of images or text prepared before printing and stored in a library. A page segment can be requested by name for use in an overlay during printing.

paper origin. The upper left corner of the paper as it goes through the printer.

parse. To analyze the syntax of a command or statement to extract the parts.

pattern. A combination of toned and untoned pels that make up an image. Synonymous with *raster pattern*.

pel. Picture element; the smallest area that can be individually toned by the printer. On printers supported by OGL/370, that area is approximately 1/240 of an inch square.

PMF. See *Print Management Facility*.

point. A unit of measurement for fonts, approximately 1/72 of an inch.

printable area. The area of the paper on which images, text, and rules can be printed.

Print Management Facility (PMF). A program that can create fonts, segments, and structures that control the placement of data on pages and the placement of pages and overlays on the paper.

Print Services Facility (PSF). A program that produces printer instructions from the data sent to it.

PSF. See *Print Services Facility*.

PTF. program temporary fix

raster pattern. See *pattern*.

rule. A solid or patterned, horizontal or vertical, straight line of any width.

section. See *overlay section*.

segment. See *page segment*.

SBCS. See *single-byte character set*.

shade. An OGL/370 option to tint part of the overlay with a selected intensity of gray.

Shift In (SI). Character used to identify the end of DBCS characters in a mixed DBCS/SBCS text string.

Shift Out (SO). Character used to identify the start of DBCS characters in a mixed DBCS/SBCS text string.

SI. See *Shift In*.

single-byte character set (SBCS). A character set whose codes require a single byte of data (for example, the character set used in English).

SMP. System Modification Program

SO. See *Shift Out*.

software support data base. I/S or EWS.

source listing. A listing of the overlay definition and messages after OGL/370 has processed the definition.

string. See *text string*.

subcommand. A keyword that introduces a distinct part of a command. For example, the REPEAT

subcommand of the DRAWBOX, DRAWCIRCLE, and DRAWRULE commands.

syntax. The rules and keywords of OGL/370.

syntaxer. A group of modules that ensures correct construction of a given OGL/370 command.

TATE. An OGL/370 keyword that arranges the characters of a line from top to bottom and the lines from right to left.

text string. Text, in the form of a string of characters, that is to be printed as part of the overlay. A text string is enclosed in apostrophes.

tokenizer. An OGL/370 module that parses the input stream from the command name to the first single semicolon (;).

typeface. The style of the font. For example, bold italic.

Virtual Machine (VM). An operating system.

Virtual Storage Extended (VSE). An operating system.

VM. See *Virtual Machine*.

VSE. See *Virtual Storage Extended*.

warning message. A message saying the overlay definition contains an error that will probably result in undesirable output.

white space. A portion of a raster pattern in which no pels are toned.

Reader's Comments

Overlay Generation Language/370 Diagnosis Guide and Reference

Publication No. LH40-0208-00

Use this form to provide comments about this publication, its organization, or subject matter. Understand that IBM may use the information any way it believes appropriate, without incurring any obligation to you. Your comments will be sent to the author's department for the appropriate action. Comments may be written in your language.

Note: IBM publications are not stocked at the location to which this form is addressed. Direct requests for publications or for assistance in using your IBM system, to your IBM representative or local IBM branch office.

- | | Yes | No |
|---|--------------------------|--------------------------|
| • Does the publication meet your needs? | <input type="checkbox"/> | <input type="checkbox"/> |
| • Did you find the information: | | |
| Accurate? | <input type="checkbox"/> | <input type="checkbox"/> |
| Easy to read and understand? | <input type="checkbox"/> | <input type="checkbox"/> |
| Easy to retrieve? | <input type="checkbox"/> | <input type="checkbox"/> |
| Organized for convenient use? | <input type="checkbox"/> | <input type="checkbox"/> |
| Legible? | <input type="checkbox"/> | <input type="checkbox"/> |
| Complete? | <input type="checkbox"/> | <input type="checkbox"/> |
| Well illustrated? | <input type="checkbox"/> | <input type="checkbox"/> |
| Written for your technical level? | <input type="checkbox"/> | <input type="checkbox"/> |
| • Do you use this publication: | | |
| As an introduction to the subject? | <input type="checkbox"/> | <input type="checkbox"/> |
| As a reference manual? | <input type="checkbox"/> | <input type="checkbox"/> |
| As an instructor in class? | <input type="checkbox"/> | <input type="checkbox"/> |
| As a student in class? | <input type="checkbox"/> | <input type="checkbox"/> |
| • What is your occupation? | | |

Thank you for your input and cooperation.

Note: If mailed in the U.S.A., no postage stamp is necessary. For residents outside the U.S.A., your local IBM office or representative will forward your comments.

Comments:[illegible]

Name _____

Address

Company or Organization

Phone No. _____

Fold and Tape

Please do not staple

Fold and Tape



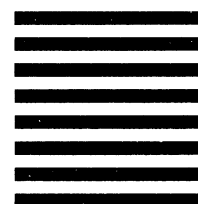
BUSINESS REPLY MAIL

FIRST CLASS MAIL PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

International Business Machines Corporation
Information Development
Department 588
PO BOX 1900
BOULDER CO 80301-9191

NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES



Fold and Tape

Please do not staple

Fold and Tape

Reader's Comments

**Overlay Generation Language/370
Diagnosis Guide and Reference
Publication No. LH40-0208-00**

Use this form to provide comments about this publication, its organization, or subject matter. Understand that IBM may use the information any way it believes appropriate, without incurring any obligation to you. Your comments will be sent to the author's department for the appropriate action. Comments may be written in your language.

Note: IBM publications are not stocked at the location to which this form is addressed. Direct requests for publications or for assistance in using your IBM system, to your IBM representative or local IBM branch office.

- | | Yes | No |
|---|--------------------------|--------------------------|
| • Does the publication meet your needs? | <input type="checkbox"/> | <input type="checkbox"/> |
| • Did you find the information: | | |
| Accurate? | <input type="checkbox"/> | <input type="checkbox"/> |
| Easy to read and understand? | <input type="checkbox"/> | <input type="checkbox"/> |
| Easy to retrieve? | <input type="checkbox"/> | <input type="checkbox"/> |
| Organized for convenient use? | <input type="checkbox"/> | <input type="checkbox"/> |
| Legible? | <input type="checkbox"/> | <input type="checkbox"/> |
| Complete? | <input type="checkbox"/> | <input type="checkbox"/> |
| Well illustrated? | <input type="checkbox"/> | <input type="checkbox"/> |
| Written for your technical level? | <input type="checkbox"/> | <input type="checkbox"/> |
| • Do you use this publication: | | |
| As an introduction to the subject? | <input type="checkbox"/> | <input type="checkbox"/> |
| As a reference manual? | <input type="checkbox"/> | <input type="checkbox"/> |
| As an instructor in class? | <input type="checkbox"/> | <input type="checkbox"/> |
| As a student in class? | <input type="checkbox"/> | <input type="checkbox"/> |
| • What is your occupation? | | |

Thank you for your input and cooperation.

Note: If mailed in the U.S.A., no postage stamp is necessary. For residents outside the U.S.A., your local IBM office or representative will forward your comments.

Comments:[illegible]

Name _____

Address

Company or Organization

Phone No. _____



Fold and Tape

Please do not staple

Fold and Tape



NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS MAIL PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

International Business Machines Corporation
Information Development
Department 588
PO BOX 1900
BOULDER CO 80301-9191



Fold and Tape

Please do not staple

Fold and Tape