GC28-0633-1
File No. S370-37

Systems

OS/VS Service Aids

VS1 Release 2
VS2 Release 1

IBM

Second Edition (September, 1972)

This is a major revision of, and obsoletes, GC28-0633-0. See the Summary of Amendments following the Contents. Changes or additions to the text and illustrations are indicated by a vertical bar to the left of the change.

This edition, as updated by GN28-2540, applies to release 2 of OS/VS1 and release 1 of OS/VS2 and to all subsequent releases until otherwise indicated in new editions or Technical Newsletters. Changes are continually made to the information herein; before using this publication in connection with the operation of IBM systems, consult the latest IBM System/360 and System/370 Bibliography, Order No. GA22-6822, and the current SRL Newsletter, Order No. GN20-0360, for the editions that are applicable and current.

Requests for copies of IBM publications should be made to your IBM representative or to the IBM branch office serving your locality.

A form for readers' comments is provided at the back of this publication. If the form has been removed, comments may be addressed to IBM Corporation, Publications Development, Department D58, Building 706-2, PO Box 390, Poughkeepsie, N.Y. 12602. Comments become the property of IBM.

How to Use This Book

This publication is for system programmers and IBM programming systems representatives. It explains when, why, and how to use IBM service aids to diagnose and fix failures in system or application programs.

Each service aid is described in a separate chapter. The chapters are arranged so that the corresponding index tabs will appear in alphabetical order. The index tabs show the names of the programs minus the three-character component identifier (such as HMD or AMD). The form of the name shown on the index tab also appears in the index to help you locate the chapter you want.

Please note that throughout the text each service aid is referred to by its abbreviated name, except where the full name of the program is necessary for technical accuracy. This means that you should expect to see HMDPRDMP or AMDPRDMP referred to as simply PRDMP, except in JCL examples and other situations where the full name is necessary. Although you may be confused by the abbreviations at first, you will soon find that the shorter names are easier to remember, because they remind you of the functions that the service aids perform.

Think of the abbreviated names as acronyms, like this:

GTF -- Generalized Trace Facility.

JOBQD -- Job Queue Dump Program (VS1 only).

LIST -- Module Listing Program.

OSJQD -- Job Queue Dump Program (VS2 only).

PRDMP -- Print Dump Program.

PTFLE -- Program Temporary Fix Link Edit Program.

SADMP -- Stand-Alone Dump Program.

SPZAP -- Superzap (Data Checker and Modifier).

Two hardware-oriented service aids, IFCDIP00 and IFCEREP0, are not documented in this publication, but in two new publications:

- OS/VS SYS1.LOGREC Error Recording, GC28-0638 -- describes how IFCDIP00 and IFCEREP0 can be used to initialize and record data from the SYS1.LOGREC data set.

- OS/VS SYS1.LOGREC Error Recording Logic, SY28-0639 -- describes the internal logic of IFCDIP00 and IFCEREP0 (how they work).

Some information about other service aids is not included in this publication, but is covered in the following publications:

- OS/VS Service Aids Logic, SY28-0635 -- describes the internal logic of the service aid programs (how they work).

- OS/VS1 Debugging Guide, GC24-5093 -- describes the dump-type output of the service aids in VS1.

- OS/VS2 Debugging Guide, GC28-0632 -- describes the dump-type output of the service aids in VS2.

- OS/VS Message Library: OLTEP and Service Aids Messages, GC38-1006 -- describes the numbered messages issued by the service aids.

You should also be familiar with the following publications:

- OS/VS Utilities, GC35-0005 -- describes how to use utility programs to print certain types of service aid output.

- Operator's Library: OS/VS1 Reference, GC38-0110 -- describes how to perform certain basic operations in VS1, such as loading a stand-alone program.

- Operator's Library: OS/VS2 Reference, GC38-0210 -- describes how to perform certain basic operations in VS2, such as loading a stand-alone program.

- OS/VS JCL Reference, GC28-0618 -- describes how to use job control statements to override default parameters, use cataloged procedures, allocate space for data sets, etc.

Contents Directory

Each chapter has its own table of contents.

Summary of Amendments for GC28-0633-1 VS1 Release 2

Information in this manual applies to both VS1 Release 2 and VS2 Release 1. This summary of amendments describes those chapters which were changed to support VS1 Release 2.

Chapter 1: GTF (Generalized Trace Facility)

This chapter contains a single change which indicates a restriction on starting GTF using the START command.

Chapter 5: PRDMP

This chapter was changed to include a description of the new USR parameter, DMA1, which has been created for EDIT.

This chapter was changed to include a FORMAT control statement restriction which occurs when input is an SVC dump.

Chapter 6: PTFLE

This chapter was changed to add a specification that the PCHF DD statement must describe Stage 1 output from the generation of the system to be updated.

Two figures were deleted because LINKS and ASMS catalogued procedures are the same for VS1 and VS2.

6.2 OS/VS Service Aids (VS1 Release 2 and VS2 Release 1)

Summary of Amendments for GC28-0633-1 VS2 Release 1

Information in this manual applies to both VS1 Release 1 and VS2 Release 1. This summary of amendments describes those chapters that were changed to support VS2.

General Comments

IFDIP00 and IFCEREP0 are no longer documented in this publication, but have been moved to a new publication, OS/VS SYS1.LOGREC Error Recording, GC28-0638.

Chapter 1: GTF (Generalized Trace Facility)

Minor changes have been made to this chapter to describe storage requirements for VS2 and a new format for timestamp records.

Chapter 3: LIST

This chapter has been changed to include the new program name for VS2, AMBLIST, wherever references to HMBLIST occur, and to show a new output format for VS2 LISTLPA.

Chapter 4: OSJQD

This is a new chapter that describes IMCOSJQD, a new service aid that operates as a problem program to dump the system job queue in VS2.

Chapter 5: PRDMP

This chapter has been extensively changed, as follows:

- The new program name for VS2, AMDPRDMP, has been included wherever references to HMDPRDMP occur.
- A new section has been added to describe printing dumps of the TSO system in VS2.
- A new section has been added to describe printing storage belonging to TSO users.
- A new section has been added to describe the PRINT PAGE facility in VS2, which differs from the VS1 facility as follows:

  a. Supports 4K page size.

  b. Permits selectivity by slot group number rather than by relative track address.

  c. Permits selectivity by device number as well as device address.
- The discussion of the LPAMAP facility has been changed: only active modules will be formatted.
- References to DAR dumps and the PRINT F03 facility have been deleted for VS2.
- New output format are shown for TCB, RB, SPCT, TJB, TJBX, and SPCA.

- Information has been added to support time-of-day-clock.
- Information has been added to support local time.

Chapter 6: PTFLE

This chapter has been changed as follows:

- The new program name for VS2, AMAPTFLE, has been included wherever references to HMAPTFLE occurs.
- New information has been added to support the PARMLIB control statement.
- New information has been added to support SYSGEN cataloged procedures.
- New information has been added to support independent component releases -- the generate function recognizes assembler steps in the SYSGEN Stage I output.
- IEHIOSUP support has been deleted for VS2.
- New information has been added to support a listing facility in the generate function.

Chapter 7: SADMP

This chapter has been changed as follows:

- The new program name for VS2, AMDSADMP, has been added wherever references to HMDSADMP occur.
- Support has been withdrawn in VS2 for shared direct access devices for real page dumping.

Chapter 8: SPZAP

This chapter has been changed to include the new program name for VS2, AMASPZAP, wherever references to HMASPZAP occur.

Appendix B: SADMP Wait State Codes

This section, formerly a part of chapter 7, lists and explains the wait state codes that the stand-alone dump program uses to communicate with the operator during execution.

Introduction

Service aids are programs designed to help system programmers and IBM programming system representatives diagnose and fix failures in system or application programs. Service aids have three general functions:

Information Gathering

- To dump real storage, use the stand-alone program SADMP. To dump the page data set, use the high-speed version of SADMP. SADMP's output can be formatted and printed using PRDMP.
- To trace system events such as SVC and I/O interruptions, use GTF (the Generalized Trace Facility). Its output can be formatted and printed using the EDIT function of PRDMP.

Formatting and Printing: Mapping

- To summarize and print records in the SYS1.LOGREC data set, use IFCEREP0, which is described in the publication OS/VS SYS1.LOGREC Error Recording, GC28-0638.
- To format and print load modules, object modules and CSECT identification records, or to map the reenterable load module area or the link pack area, use LIST.
- To format and print the system job queue, use JOBQD in VS1, and OSJQD in VS2.
- To format and print SADMP output, other system dumps, and GTF trace output, use PRDMP.

Generating and Applying Fixes

- To apply a PTF or an ICR, use PTFLE.
- To verify and/or replace instructions in a load module, or data on a direct access device, use SPZAP.
- To initialize the SYS1.LOGREC data set, use IFCDIP00, which is described in the publication OS/VS SYS1.LOGREC Error Recording, GC28-0638.

For more detailed information about choosing a service aid, refer to the table in Figure INTRO-1.

The numbers in this table refer to the explanatory notes on the accompanying sheets. For each symptom, read from left to right to find out which functions of these service aids you should use to diagnose and fix the problem. For complete information about IFCDIP00 and IFCEREPO, see OS/VS SYS1. LOGREC Error Recording, GC28-0638. For complete information about the other service aids, see OS/VS Service Aids, GC28-0633.

SYMPTOM	INFORMATION GATHERING		MAPPING, FORMATTING, AND PRINTING				PATCHING		
	SADMP	GTF	PRDMP	LIST	EREP0	JOBQD OSJQD	PTFLE	SPZAP	DIP00
Warm Start Failure	1	-	5c-e	-	-	15	-	21	-
Scheduler ABEND	-	2	6	8,9	-	15	-	21	-
Writer ABEND	-	2	6	-	-	15	-	21	-
Problem Program ABEND	-	4	6	9	-	-	-	21	-
Recursive ABEND	1	2	5a,5c-d,6	9,12	-	16	-	21	-
Disabled Loop	1	2	5c-e,6	-	-	-	-	-	-
Problem Program Loop	-	4	6	9	-	-	-	-	-
Large Loop with I/O	1	2	5a,5c-e,6b-d	12	-	-	-	21	-
DAR Loop (VS1 Only)	1	2	5c,5e,6	9	13	-	-	-	24
Hard Wait	1	2	5c-e	8,9	13	-	-	-	24
Enabled Wait	1	2	5b,6	9	13	-	-	-	24
Reader/Interpreter Failure	-	-	-	-	-	15	-	21	-
I/O Failure (e.g. console)	1	3	5a-e,6b-d	12	13,14	-	-	21,23	-
Allocation Failure	1	-	5b-d	9	-	-	-	21	-
Enqueued Job Lost	-	3	-	-	-	17	-	-	-
Chain Scheduling Problem	1	3	5a,5c-e,6b-d	12	-	-	-	-	-
Access Method Failure	-	3	6	-	14	-	-	21	-
Data Management Program Chk	-	2,4	6	9	-	-	-	-	-
Module Level Unknown	-	-	-	10	-	-	-	22	-
User Modification Unknown	-	-	-	11	-	-	-	22	-
Applying PTF	-	-	-	-	-	-	18	20	-
Applying ICR	-	-	-	-	-	-	19	-	-
Applying Local Fix	-	-	-	-	-	-	18	20	-
APAR Documentation	1	2,4	5a,5c-e,6	10,12	-	15	-	22	-
Print SYS1.DUMP	-	-	5b-d,6	-	-	-	-	-	-
Capturing System Before Re-IPL	1	-	5a-e,6	12	-	-	-	-	-
TSO Failure	1	-	5a-e,6,7	12	-	-	-	-	-

INFORMATION GATHERING

SADMP

1. Dumps the contents of real or virtual storage to a tape, which can be formatted and printed using PRDMP. (Note that SADMP output may also be directed to a printer.)

GTF

2. Traces all system events.
3. Traces selected events, such as I/O interruptions, SIO operations, etc.
4. Traces user programs with GTRACE macro instruction.

MAPPING, FORMATTING, AND PRINTING

PRDMP

5. Formats and prints the following from SADMP high-speed output:
   a. Link pack area.
   b. Queue control block trace.
   c. Major system data areas.
   d. Selected areas of storage by virtual or real address.
   e. Operating system nucleus.

6. Formats and prints selected records from the GTF trace data set or from trace buffers in a SYS1.DUMP or SADMP output data set. Records are selected by keywords such as:
   a. JOBNAME.
   b. I/O.
   c. SVC.
   d. SIO.

7. Formats and prints TSO data areas and storage and TSO user data areas and storage.

LIST

8. Lists specific object modules, load modules, or load modules in a data set.
9. Maps control sections and overlay structure and lists cross-references within a load module.
10. Lists CSECT identification records for specific load modules.
11. Lists translation data, linkage editor modification data, or SPZAP modifications to control sections in a load module.
12. Maps reenterable load module area (VS1) or link pack area (VS2).

EREP0

13. Selects, formats and prints records from the SYS1.LOGREC data set, by record type:
    a. Machine check and/or inboard.
    b. Outboard.
14. Selects records by device type or device address.

JOBQD (VS1) or OSJQD (VS2)

15. Dumps entire SYS1.SYSJOBQE data set.
16. Selects, formats, and prints job queue records associated with a specific job.
17. Selects, formats, and prints job queue records associated with a specific work queue.

PATCHING

PTFLE

18. Generates control statements and JCL needed to apply PTFs; the application function also invokes the linkage editor.
19. Generates control statements and JCL needed to apply ICRs.

SPZAP

20. Modifies data in a load module.
21. Sets traps by inserting invalid instructions or user-written SVCs.
22. Dumps load modules by CSECT to allow examination of the text.
23. Dumps selected data to verify the count, key and contents of the data.

DIP00

24. Reinitializes the SYS1.LOGREC data set if destroyed.

Figure INTRO-1. Service Aids Symptom Table

Chapter 1: GTF (Generalized Trace Facility)

Traces selected system events such as SVC and I/O interruptions.

Contents

GTF

Figures

Introduction

The Generalized Trace Facility (GTF) is a feature of OS/VS that allows you to trace selected system events. It also allows you to create your own user trace records and include them in the trace output, which may be directed to buffers in virtual storage (internal) or to a data set (external). The trace output, when formatted and printed by the EDIT function of PRDMP, is useful in determining and diagnosing problems that may arise while using OS/VS. (For information about EDIT and PRDMP, see Chapter 5 in this publication.)

Features

GTF operates as a system task under OS/VS; it supports a minimum CPU storage size of 144K for internal tracing or 160K for external tracing. If the TRACE option has been selected at system generation, the OS/VS Trace facility will function normally except during GTF processing, when OS/VS Trace processing will be suspended.

GTF can trace any or all of the following system events:

- Input/output interruptions (IO)
- START I/O operations (SIO)
- Supervisor Call interruptions (SVC)
- Program interruptions (PI)
- External interruptions (EXT)
- Dispatcher task-switch operations (DSP)
- User events (USR)
- Events associated with the trace task itself (TRC)

If you choose IO or SIO, you can supply specific device addresses in response to a prompting message. GTF will then selectively trace only those IO or SIO events that are associated with the devices you specified. Similarly, you can cause selective tracing of specific SVC numbers when you choose SVC tracing, and specific program interrupt codes when you choose PI tracing. Events not selected for tracing are filtered out (not traced).

GTF will ordinarily ignore traceable events that are associated with its own task, but you can request that such events be included as part of the trace output (TRC). You can also request that a timestamp be included in each trace record (TIME=YES).

GTF trace output can be maintained in storage (MODE=INT) or directed to a data set on an external storage device (MODE=EXT). The output device may be any magnetic tape or direct access device supported by OS/VS.

If GTF runs out of output space, either in storage or on an output direct access volume, it overlays previously stored or written output beginning at the first buffer or block.

Any abnormally terminating user task that has requested ABEND processing will be supplied with formatted trace data as part of the ABEND dump if GTF was active with MODE=INT when ABEND was given control, and if you had provided a SYSABEND DD statement. Similarly, trace data will be provided for SNAP dumps if the user has included the SDATA=TRT parameter in the SNAP macro.

Starting GTF

You start GTF as a system task by entering a START command from the operator's console. (GTF cannot be started as a job.) By specifying certain optional parameters, you can choose whether the trace records should be recorded internally or externally, whether or not they should be time-stamped, and whether or not GTF should terminate if it encounters errors while gathering trace information. You can also select trace options, either by entering them directly through the console or by retrieving them from SYS1.PARMLIB where you have stored them.

Using the START Command

Figure GTF-1 shows the general format of the START command as it is used to start GTF.

```
START procname[.identifier],[devaddr],[volser],[(parmvalue)]
     [,keyword=option][...,keyword=option]
```

Figure GTF-1. General Format of the Start Command for GTF

The following discussion describes the parameters of the START command as they are used for GTF.

procname.identifier

defines one of the two cataloged procedures (GTF and GTFSNP) described in the next section. The qualifier ".identifier" allows you to specify the partition where you want GTF to execute.

devaddr

indicates the address of the device to which trace output is to be written, if you have specified MODE=EXT. If you have specified MODE=INT, omit this field.

volser

defines the volume serial number of the direct access storage pack to which trace output is to be written, if you have specified MODE=EXT. If you specified MODE=INT, omit this field.

parmvalue

overrides the value specified in the PARM= parameter of the EXEC statement in the cataloged procedure GTF or GTFSNP. This field may contain any combination of the following parameters:

```
MODE=(INT     )
     {EXT     }
     ((INT.S) )
```

defines where the trace data is to be maintained. IF you omit this parameter, GTF will assume the default specified in the cataloged procedure (MODE=EXT) and write the trace data on the SYS1.TRACE data set. When MODE=EXT is in effect, you will be prompted to supply trace options unless you have specified a member of SYS1.PARMLIB where trace options are stored.

When MODE=INT is in effect, the trace data is maintained in main storage, and GTF will not prompt you to supply trace options. It will gather basic data (similar to that contained in the operating system trace table) for the following events:

- Dispatcher entries
- External interrupts
- I/O interrupts, including program-controlled interrupts.
- Program interrupts
- SIO operations
- SVC interrupts.

When any task in the system terminates abnormally and the ABEND routine is invoked, GTF will suspend tracing until the ABDUMP program can format the trace data as part of the dump output. Trace events missed during ABEND processing will be counted in a lost event record that will be included in the trace buffers. If ABEND is not invoked, tracing will continue unaffected. If you specified MODE=(INT,S), GTF will not pause for ABEND or SNAP processing, and the trace buffers will not be formatted.

While MODE=INT does not provide trace information with as much detail as MODE=EXT, you chould consider specifying MODE=INT whenever GTF is to be run for long periods; by eliminating time required to write data to an external device, you can thus reduce GTF's impact on total system processing.

$\text{TIME=}\begin{Bmatrix}\text{YES}\\\text{NO}\end{Bmatrix}$

TIME=YES requests that every logical trace record be timestamped with the time-of-day clock value at the time the record was constructed. This timestamp is in addition to the block timestamp associated with every block of data.

If you code TIME=NO, or if you omit this parameter, GTF will not timestamp individual records.

$\text{DEBUG=}\begin{Bmatrix}\text{YES}\\\text{NO}\end{Bmatrix}$

GTF may encounter errors while attempting to create a trace record. If you specify DEBUG=YES, most errors of this kind will cause GTF to issue an error message and then terminate, so that the contents of the GTF buffers immediately prior to the error will be unchanged. If you have named the GTFSNP procedure in the START command, a SNAP dump will be produced if GTF terminates abnormally.

If you specify DEBUG=NO, or if you omit this parameter, GTF will not terminate immediately, but instead will initiate error recovery procedures. For more information about error recovery procedures, refer to the section "GTF Error Recovery Handling" later in this chapter.

BUF=nnn

This parameter allows you to specify the number of buffers (1 to 255) to be used for recording trace data; thus it overrides the BUFNO= subparameter of the DCB= parameter of the IEFRDER DD statement. If you omit this parameter, GTF will obtain the number of buffers specified in the BUFNO= subparameter. If neither the BUF= parameter nor the BUFNO= parameter is specified, GTF will assume the following default values:

- If MODE=EXT, GTF obtains 3 buffers.
- If MODE=INT, GTF obtains 4 buffers.

Note: the BUF= parameter should be used with caution, since buffers are maintained in fixed storage. The more buffers you request, and the larger they are, the more of your real storage will be fixed and unavailable for paging. For more detailed information concerning GTF fixed storage requirements, see OS/VS1 Storaqe Estimates, GC24-5094; or OS/VS2 Storaqe Extimates, Gc28-0604.

keyword=option

You may use this parameter to override specific parameters in the IEFRDER DD statement in the cataloged procedure. For example:

- To specify a different name for the trace data set, code DSNAME=newname.
- To prevent the system from sending mount messages to the operator's console when specifying MODE=INT, code DSN=NULLFILE.
- To specify an existing data set as the output data set, code DISP=OLD. (Note: If you specify DISP=MOD, GTF will change the data set disposition to OLD.)
- To modify the GTF buffer size code DCB=(BLKSIZE=number). The minimum default block size is 350 bytes. Note that if you intend to trace events associated with the trace task itself (TRC option), you should specify a large blocksize to avoid continuous writing to tape.

  Whenever GTF is to be run for long periods, use this DCB parameter to request buffers as large as the track size on a direct access degice, or as large as is practical for tape. Requesting a few large buffers, rather than many smaller ones, tends to reduce GTF's impact on total system processing.
- To run GTF in a virtual=real address space and thus reduce its impact on total system processing, code ADDRSPEC=REAL,REGION=nnK. (VS2 only)

Do not use this parameter to request DCB=OPTCD=C; GTF does not support chain-scheduling.

Using the GTF Cataloged Procedure

The START command for GTF names one of two cataloged procedures supplied in SYS1.PROCLIB. The first, GTF, contains job control statements as shown in Figure GTF-2. The second, GTFSNP, is identical to cataloged procedure GTF except that the SNAPDUMP DD statement, shown as optional in Figure GTF-2, is supplied.

```
 //GTF          PROC        [&REG=64]
 //IEFPROC      EXEC        PGM=xHLGTF[,REGION=&REG.K],
 //        PARM='MODE=EXT,DEBUG=NO,TIME=NO'
 //IEFRDER      DD          DSNAME=SYS1.TRACE,UNIT=SYSDA,
 //        SPACE=(3500,20),DISP=(NEW,KEEP)
 //SYSPRINT     DD          SYSOUT=A,SPACE=(TRK,(1,1))
[//SYSLIB       DD          DSN=SYS1.PARMLIB (membername),]
[//        DISP=SHR]
[//SNAPDUMP     DD          SYSOUT=A]
```

Note: The GTFSNP cataloged procedure contains a SNAPDUMP DD statement, shown here as an optional statement.

Figure GTF-2. The GTF Cataloged Procedure

PROC Statement

defines the cataloged procedure GTF or GTFSNP. The &REG= parameter applies to VS2 only.

EXEC Statement

calls for the execution of HHLGTF (VS1 only) or AHLGTF (VS2 only). The REGION= parameter applies to VS2 only.

IEFRDER DD Statement

defines the trace output data set, according to the following defaults: the trace output data set will have the name SYS1.TRACE; it will be directed to a direct access device with sufficient allocation to allow the data set to contain twenty 3500-byte physical blocks. Three 3500-byte buffers will be provided to contain these blocks. If you want to establish a new default number of buffers, code the BUF= parameter in the GTF START command.

To reduce GTF's impact on total system processing, consider overriding this statement to define the trace data set as residing on a tape volume rather than on a direct access device.

SYSPRINT DD Statement

defines the GTF message data set.

SNAPDUMP DD Statement (Optional in the cataloged procedure GTF, supplied in GTFSNP.)

causes GTF to issue the SNAP macro to dump the nucleus and the GTF region if an error condition causes GTF to terminate. This statement increases GTF's virtual storage requirements by 4K.

SYSLIB DD Statement (Optional)

defines a member in the SYS1.PARMLIB data set that contains GTF options. If such a member exists, GTF will not prompt you to supply options, but will use the options in the member.

Specifying GTF Trace Options

When you start GTF with MODE=EXT, you will receive the following message:

```
{HHL100A}  SPECIFY TRACE OPTIONS.
{AHL100A}
```

Use the following format to specify the events to be recorded during GTF execution:

```
TRACE=option1[,option2]...[,optionx]
```

You can specify any of the following trace option values:

```
{SYS }
{SYSM}
{SYSP}
```

SYS requests that comprehensive trace data be recorded for the following system events:

- I/O interrupts
- SVC interrupts
- Program interrupts
- External interrupts
- Start I/O operations

Note: Tracing for dispatcher task-switching must be requested separately through the DSP keyword. Similarly, tracing for program-controlled interrupts must be requested separately through the PCI keyword.

SYSM requests that minimal trace data be recorded for all system events listed above. SYSP requests further prompting for IO, SIO, SVC, and PI; that is, if you specify SYSP, GTF will prompt you to supply specific device addresses, SVC numbers, or program interrupt codes. Comprehensive trace data will be recorded for events associated with the devices or interrupts that you specify; all other events will be filtered out and ignored. If SYS and SYSM, or SYS and SYSP, are both specified, SYS will be ignored. Similarly, if SYSP and SYSM are both specified, SYSP will be ignored.

You should consider specifying SYSP, and in subsequent prompting request only a few specific trace events, whenever GTF is to be run for long periods; by reducing the amount of data to be written to an external device, you can thus reduce GTF's impact on total system processing.

```
{SIO }
{SIOP}
```

SIO requests comprehensive recording for system SIO operations on all devices. SIOP requests further prompting for specific devices for which trace data should be recorded.

This keyword will be ignored if SYS, SYSM, or SYSP has also been specified.

$\left\{\begin{array}{l}\text{IO}\\ \text{IOP}\end{array}\right\}$

IO requests comprehensive recording for all I/O interrupts except program-controlled interrupts, which must be requested spearately through the PCI keyword. IOP requests further prompting for specific devices for which I/O interrupts should be recorded.

This keyword will be ignored if SYS, SYSM, or SYSP has also been specified.

$\left\{\begin{array}{l}\text{SVC}\\ \text{SVCP}\end{array}\right\}$

SVC requests comprehensive recording for all SVC interrupts. SVCP requests further prompting for specific SVC numbers for which trace data should be recorded.

This keyword will be ignored if SYS, SYSM, or SYSP has also been specified.

$\left\{\begin{array}{l}\text{PI}\\ \text{PIP}\end{array}\right\}$

PI requests comprehensive recording for all program interrupts. PIP requests further prompting for specific interrupt codes for which trace data should be recorded.

This keyword will be ignored if SYS,SYSM, or SYSP has also been specified.

EXT

requests comprehensive recording for all external interrupts. This keyword will be ignored if SYS, SYSM, or SYSP has also been specified.

DSP

requests that a trace record be created whenever the dispatcher is entered for task switching. The trace data collected will be comprehensive unless you have requested SYSM.

USR

requests that all data passed to GTF via the GTRACE macro be recorded with the system data in the trace data set.

PCI

requests that all program-controlled I/O interrupts be recorded. This keyword will be ignored unless IO, IOP, SYS, SYSM, or SYSP is also specified. If you have specified IOP or SYSP, program-controlled I/O interrupts will be recorded only for those devices that you supplied in response to a prompting message.

TRC

requests tracing of trace events associated with the trace task while operating under GTF's task control block. Such events will be traced according to the GTF trace options selected while starting GTF. If this keyword is not specified, GTF task events will be filtered out and not recorded.

Prompting

When you specify SYSP, IOP, SIOP, SVCP, or PIP as trace options, GTF will prompt you to supply specific values. These values are:

SIO=(devaddr1[,devaddr2][...,devaddr50])

specifies up to 50 device addresses for which you want SIO operations traced. All other SIO operations will be filtered out. If you have specified SIOP or SYSP, and do not specify SIO= in response to the prompting message, no SIO filtering will take place.

IO=(devaddr1[,devaddr2][...,devaddr50])

specifies up to 50 device addresses for which you want I/O interruptions traced. All other IO interruptions will be filtered out. If you have specified IOP or SYSP, and do not specify IO= in response to the prompting messages, no I/O interruption filtering will take place.

IO=SIO=(devaddr1[,devaddr2][...,devaddr50])

specified after requesting SYSP or both IOP and SIOP, names up to 50 device addresses for which you want GTF to trace both IO and SIO events. All other IO and SIO events, except those requested specifically by IO= or SIO=, will be filtered out.

SVC=(svcnum1[,svcnum2][...,svcnum50])

specifies up to 50 SVC numbers that you want traced. All other SVC numbers will be filtered out. If you have specified SVCP or SYSP, and do not specify SVC= in response to the prompting message, no SVC filtering will take place.

PI=(code1[,code2][...,code15,code17,code19]) (VS1)

PI=(code1[,code2][,...,code15,code17,code18,code19,] (VS2)

specifies up to 17 (for VS1) or 18 (for VS2) program interrupt codes that you want traced. (Valid program interrupt codes in VS1 are 1 through 15, 17, and 19. In VS2, code 18 is also valid.) All other program interruptions will be filtered out. If you have specified PIP or SYSP, and do not specify PI= in response to this prompting message, no program interruption filtering will take place.

Note that in each case GTF imposes a limit on the number of specific values you can supply through prompting. If you exceed this limit, GTF will issue a message and you must respecify all values.

Figure GTF-3 shows an example of an exchange between GTF and the operator when GTF is being started. The example applies to VS1; in VS2 the message numbers would begin with AHL, and the START command would read:

```
START GTF,,,(MODE=EXT)
```

```
    START GTF.P3,,,(MODE=EXT)
          .
          .
          .
  00 HHL100A SPECIFY TRACE OPTIONS
r 00,'TRACE=SYSP,USR'
          .
          .
          .
  01 HHL101A SPECIFY TRACE EVENT KEYWORDS--SVC=,IO=,SIO=,PI=
r 01,'SVC=(1,2,3,4,10),IO=(191,192)'
          .
          .
          .
  02 HHL102A CONTINUE TRACE DEFINITION OR REPLY END
r 02,'SIO=282,END'
HHL103I TRACE OPTIONS SELECTED--SYSP,USR
HHL103I SVC=(1,2,3,4,10),IO=(191,192),SIO=(282)
  03 HHL125A RESPECIFY TRACE OPTIONS OR REPLY U
r 03,'U'
```

Figure GTF-3. GTF Messages and Operator Replies While Starting GTF.

Storing Trace Options in SYS1.PARMLIB

You can save time when starting GTF by previously storing one or more set combinations of trace options as members in SYS1.PARMLIB, and including a SYSLIB DD statement in the GTF or GTFSNP cataloged procedures. If you do this, GTF will not prompt you to supply trace options, but will get them from SYS1.PARMLIB.

Figure GTF-4 shows the job control statements and utility control statements needed to add trace options to SYS1.PARMLIB using IEBUPDTE. For full descriptions of the statements, refer to the publications OS/VS Utilities, GC35-0005, and OS/VS JCL Reference, GC28-0618.

```
//GTFPARM     JOB       MSGLEVEL=(1,1)
//            EXEC      PGM=IEBUPDTE,PARM=NEW
//SYSPRINT    DD        SYSOUT=A
//SYSUT2      DD        DSNAME=SYS1.PARMLIB,DISP=SHR
//SYSIN       DD        DATA
./       ADD       NAME=GTFA,LIST=ALL,SOURCE=0
TRACE=SYSP,USR
SVC=(1,2,3,4,10),IO=(191,192),SIO=282,PI=15
./       ADD       NAME=GTFB,LIST=ALL,SOURCE=0
TRACE=IO,SIO,TRC
./       ADD       NAME=GTFC,LIST=ALL,SOURCE=0
TRACE=SYS,PCI
/*
```

Figure GTF-4. Adding Trace Options to SYS1.PARMLIB Using IEBUPDTE.

A sample SYSLIB DD statement to be included in the GTF or GTFSNP cataloged procedure might look like this:

```
    //SYSLIB  DD   DSN=SYS1.PARMLIB(GTFA),DISP=SHR
```

Calculating Storage Requirements

GTF's partition or region size requirements vary according to the GTF options that you specify.

Internal Trace

If you request MODE=INT, you must specify a minimum partition or region size of 64K. For partitions or regions larger than this minimum, use the following formula to calculate your storage requirements. Note that the final partition or region size must be rounded up to the nearest 64K multiple, since virtual storage is assigned in 64K segments. Approximately 30 trace events will fit in a single 1K buffer.

size = 32K + T + a

Where:

32K

the amount of virtual storage needed for GTF initialization routines, trace routines, and control blocks.

T

the amount of storage required for the trace buffers, rounded up to the nearest 2K multiple for VS1, or the nearest 4K multiple for VS2. To calculate T, multiply the number of trace buffers that you intend to request by the size of each trace buffer (1K).

Note that for MODE=INT, the default number of buffers is 4. If you want to request more than 4, you must use the BUF= parameter of the GTF START command. If you do not specify the BUF= parameter, the value of T will default to 4K. (When you specify MODE=INT, the DCB=BUFNO= parameter of the IEFRDER DD statement has no effect. If you attempt to substitute BUFNO= for BUF= in the GTF START command, BUFNO= will be ignored, and you will be assigned the default number of buffers.)

a

the amount of storage required for SNAP processing. If you intend to invoke the GTFSNP cataloged procedure, this value is 4K. If you intend to invoke the GTF cataloged procedure (without a SNAPDUMP DD statement), this value is zero.

External Trace

If you have requested MODE=EXT, you must specify a minimum partition or region size of 64K. For larger partitions or regions, use the following formula to compute your storage requirements. Note that all intermediate values must be rounded up to the nearest 2K multiple for VS1, or the nearest 4K multiple for VS2. The final partition or region size that you calculate must be rounded up to the nearest 64K multiple since virtual storage is assigned in 64K segments. Approximately 60 trace events will fit in a single 3500-byte buffer.

$$\text{size} = 32K + n(b+8) + 88(n) + m\ [+\ p] + a$$

Where:

32K

the amount of virtual storage needed for GTF initialization routines, trace routines, and control blocks.

n

the number of trace buffers that you intend to request in the GTF START command. Note that for MODE=EXT, the default number of buffers is 3. If you want to request more than 3, you can use either the BUF= parameter of the GTF START command or the DCB=BUFNO= parameter of the IEFRDER DD statement. BUF= overrides BUFNO= if both are specified. If you specify neither BUF= nor BUFNO=, this value will be 3.

b

the size of the trace buffers. The default buffer size is 3500 bytes. If you want larger or smaller buffers, override the DCB=BLKSIZE= parameter in the IEFRDER DD statement in the GTF cataloged procedure. The additional 8 bytes are needed for the GTF buffer prefix.

88

the size of the input/output block (IOB); one IOB is required for each buffer.

m

total storage required to process GTF options requested. In some cases, several GTF options are contained within one module. Even if you request two or more GTF function that are contained in the same module, you only need to provide enough space for one copy of the module. Refer to Figure GTF-5 for a summary of GTF options, the modules that contain them, and the amount of storage required for each module.

To calculate m, add together the storage requirements for each module that you will need, and add 1K to the total if you have requested filtering for any option. For example, if you specify EXT, SVCP, and USR:

$$m = 2K + 8K + 1K + 1.5K$$
$$m = 12.5K$$

p (for VS1 only)

the amount of storage required for a pool of control blocks (TIRBs) that GTF uses to schedule asynchronous tracing when it encounters a disabled page fault while tracing an SVC interruption or a user program that specified the GTRACE macro. If you intend to request SYS, SYSP, SVC, SVCP, or USR, this value should be 2K. Otherwise, this value is zero.

a

the amount of storage required for SNAP processing. If you intend to invoke the GTFSNP cataloged procedure, this value should be 4K. If you intend to invoke the GTF cataloged procedure (without the SNAPDUMP DD statement), this value is zero.

GTF OPTIONS SELECTED	MODULES REQUIRED		STORAGE REQUIRED
	VS1	VS2	
SYSM[,DSP][,PCI]	HHLSYSV	AHLSYSV	1.5K
DSP EXT PI PI=	HHLTPED	AHLTPED	2.0K
IO IO= SIO SIO= PCI	HHLTSIO	AHLTSIO	1.5K
SVC SVC=	HHLTSVC	AHLTSVC	8.0K
SYS[,DSP][,PCI]	HHLTPED, HHLTSIO, and HHLTSVC	AHLTPED, AHLTSIO, and AHLTSVC	11.5K
USR	HHLTUSR	AHLTUSR	1.5K
IOP SIOP SVCP PIP	HHLTFIL	AHLTFIL	1.0K

Figure GTF-5. Virtual Storage Requirements for GTF Options, by module. Note that TRC can be considered to require 0 (zero K) bytes of virtual storage.

Recording User Data

If you want your own trace data to be recorded in the GTF trace buffers, you can use the GTRACE macro instruction to define the data. In one invocation of GTRACE, an application program can record up to 256 bytes of data in a GTF trace buffer. Secure data should not be recorded using the GTRACE macro since security protection cannot be guaranteed. Note, however, that GTRACE can record only data that has the same protect key as the GTRACE user.

GTRACE will be effective only when GTF is active, when it is directing its output to an external data set, and when it is accepting user data -- that is, when GTF has been started with MODE=EXT and TRACE=USR.

Printing User Data

Like other trace data, information recorded by the GTRACE macro can be printed by the EDIT function of PRDMP. Usually user data will be printed in hexadecimal, since EDIT cannot format records not created by GTF. However, you can write format appendages to format specific types of user data records. For information about writing EDIT format appendages, see Appendix A: Writing EDIT User Programs. (Note: If your installation has format appendages written for use with OS/MFT or OS/MVT, you can still use them in OS/VS. EDIT recognizes and will accept format appendages named IMDUSRxx as well as those named HMDUSRxx and AMDUSRxx.)

Every time you issue GTRACE to create a user record, you specify which format appendage should process it; you do this by including the optional FID (format identifier) parameter in the GTRACE invocation. The FID corresponds to the last two hexadecimal characters in the name of the format appendage, HMDUSRxx or AMDUSRxx.

Coding the GTRACE Macro

Figure GTF-6 shows the general format of the GTRACE macro, standard form.

```
[symbol] GTRACE    DATA=address,LNG=number,ID=number[,FID=value]
```

Figure GTF-6. General Format of the GTRACE Macro, Standard Form

The parameters in the macro are described below.

DATA=address

gives the main storage address of the data to be recorded.

LNG=number

specifies the number of bytes (1 to 256) to be recorded from the address specified in the DATA= parameter. The number may be specified in decimal or in hexadecimal (as X'number').

ID=value

is the identifier to be associated with the record. ID values are assigned as follows:

0 to 1023 -- user events

1024 to 4095 -- reserved

The value may be specified in decimal or in hexadecimal (as X'value').

FID=value

indicates the format appendage that is to format this record when the trace output is processed by the EDIT function of HMDPRDMP. FID values are assigned as follows:

0 (or FID= parameter omitted) -- record to be dumped in hexadecimal

1 to 80 -- user format identifiers

81 to 255 -- reserved

The value may be specified in decimal or in hexadecimal (as X'value').

Figure GTF-7 shows how the GTRACE macro might be coded to record 200 bytes of data, beginning at the address of AREA, with an event identifier of 37 and to be formatted by the format appendage with the name IMDUSR40.

```
GTRACE   DATA=AREA,LNG=200,ID=37,FID=64
```

Figure GTF-7. An Example of the GTRACE Macro.

For more details about the GTRACE macro instruction, consult the publication OS/VS Supervisor Services and Macro Instructions, GC27-6979.

GTF Error Recovery Handling

GTF recognizes all errors that occur while building a trace record as potentially recoverable. Whether or not recovery is attempted depends on what you specify in the START command.

If you specify DEBUG=YES, GTF will not attempt error recovery. It will issue an error message and then terminate, so that the contents of the GTF buffers immediately prior to the error will be preserved.

If you specify DEBUG=NO, GTF will initiate the following error procedures:

- For minor errors in the routine that builds the trace record (the build routine), GTF flags the field in the trace record that led to the error and continues processing. It does not issue a message to the operator's console or disable the function that caused the error; instead, it proceeds as if no error had occurred. All errors that occur while building an SVC record fall into this category.
- For severe errors in the build routine, GTF flags the entire record that was being built, issues a message to the console, and continues processing with the function that caused the error suppressed.
- For errors in the routine that filters trace events, GTF suppresses filtering for future events of the same type, issues a message to the console, and continues processing.

Errors that occur outside the build and filter routines are not recoverable; they result in immediate abnormal termination of GTF.

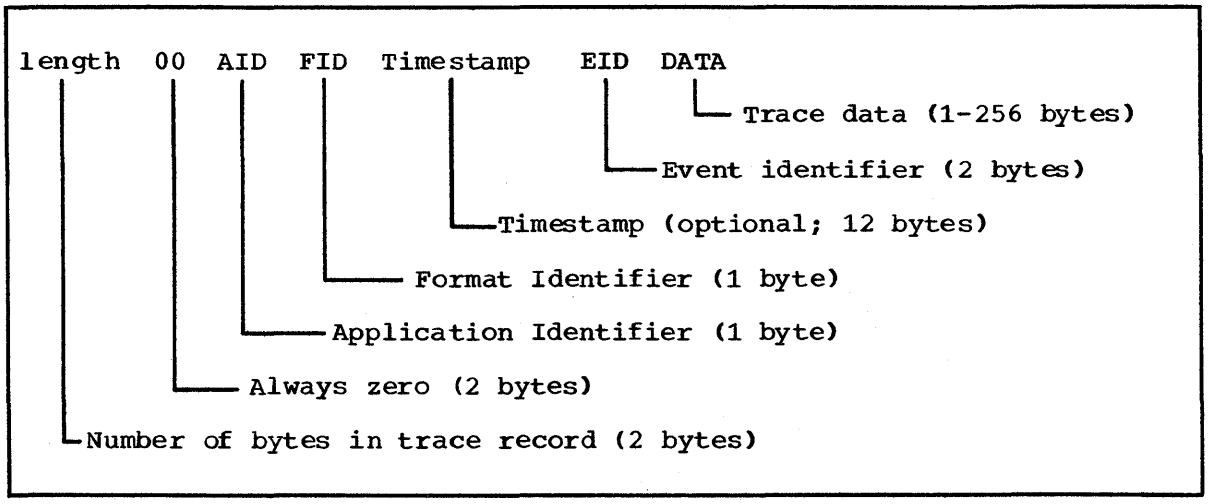
Note that the termination of GTF will never cause termination of a user's task.

GTF Output

GTF creates two kinds of records: trace records and control records.

Trace Records

GTF creates trace records for each system event you select. The records have the general format shown in Figure GTF-8.



length 00 AID FID Timestamp EID DATA

Trace data (1-256 bytes)

Event identifier (2 bytes)

Timestamp (optional; 12 bytes)

Format Identifier (1 byte)

Application Identifier (1 byte)

Always zero (2 bytes)

Number of bytes in trace record (2 bytes)

Figure GTF-8. Fields in a Trace Record.

The fields in the record are described as follows:

length

indicates the total length of the record in bytes.

00

always zero.

AID

defines whether the data record is a trace record or a GTF control record.

X'FF' -- Trace record for data gathered synchronously

X'FE' -- Trace record for data gathered asynchronously

X'FD' -- Indicates a basic SVC trace record when a comprehensive trace record was requested and the trace data could not be gathered asynchronously.

X'00' -- GTF control record

X'01' to X'FC' -- reserved

FID

is the format identifier, a one-byte hexadecimal number that identifies the program that will format the trace record during EDIT execution. (For information on specifying the FID in the GTRACE macro, refer to "Coding the GTRACE Macro" in this chapter.)

If this field is zero, the trace record will not be formatted, but will be dumped in hexadecimal.

timestamp

If TIME=YES was specified in the START command, a timestamp will be included in this twelve-byte field. The value in the low-order 8 bytes of the record will be the clock value at the time the record was constructed. The high-order 4 bytes contain the time zone value, a factor used in converting Greenwich Mean Time to local time.

EID

defines the event that caused the trace record to be created. It is not present in GTF control records. You can determine the EID of a trace record by issuing the IMDMEDIT mapping macro, which is described in the Appendix: Writing EDIT User Programs.

data

This field contains the trace data gathered for the requested event. The length of this field varies according to the event being traced.

Figures GTF-9 through GTF-11 are examples of trace output as processed by the EDIT function of IMDPRDMP. In all the examples, fields flagged with h-----h are hexadecimal representations, and fields flagged with c-----c are alphameric characters. N/A signifies that the field label does not apply to this particular record.

```
I/O   cuu OLD PSW hhhhhhhh hhhhhhhh  JOBN cccccccc  DDNM cccccccc RCSW hhhhhhhh
PCI           CSW hhhhhhhh hhhhhhhh  RQE hhhhhhhh hhhhhhhh hhhhhhhh hhhhhhhh hhhhhhhh SENS hhhhhhhh

SIO   cuu     CC  hh   CAW hhhhhhhh  JOBN cccccccc                VSTRT hhhhhhhh
              CSW hhhhhhhh hhhhhhhh  RQE hhhhhhhh hhhhhhhh hhhhhhhh hhhhhhhh hhhhhhhh

PGM   ddd OLD PSW hhhhhhhh hhhhhhhh  JOBN cccccccc  MODN cccccccc OLTCB hhhhhhhh  VPA hhhhhhhh
          R0  hhhhhhhh  R1 hhhhhhhh  R2   hhhhhhhh  R3   hhhhhhhh R4  hhhhhhhh  R5  hhhhhhhh R6  hhhhhhhh  R7  hhhhhhhh
          R8  hhhhhhhh  R9 hhhhhhhh  R10  hhhhhhhh  R11  hhhhhhhh R12 hhhhhhhh  R13 hhhhhhhh R14 hhhhhhhh  R15 hhhhhhhh

EXT       OLD PSW hhhhhhhh hhhhhhhh  JOBN cccccccc  MODN cccccccc NUTCB hhhhhhhh  TQEFLG/TCB hhhhhhhh      EXIT hhhhhhhh

DSP       RES PSW hhhhhhhh hhhhhhhh  JOBN cccccccc  MODN cccccccc NUTCB hhhhhhhh    PRTY  hh

SVC   nnn OLD PSW hhhhhhhh hhhhhhhh  JOBN cccccccc  MODN cccccccc OLTCB hhhhhhhh  R15/R0  hhhhhhhh hhhhhhhh  R1 hhhhhhhh
                 (some SVCs have continuation lines)
```

Figure GTF-9. Format of Comprehensive Trace records for DSP, IO (including PCI),SIO, PI, EXT, and SVC.

```
I/O     OLD PSW     hhhhhhhh hhhhhhhh   CSW      hhhhhhhh hhhhhhhh   RCSW     hhhhhhhh   RQE TCB   hhhhhhhh
PCI

SIO     CC/DEV/CAW  hhhhhhhh hhhhhhhh   CSW      hhhhhhhh hhhhhhhh   VSTART   hhhhhhhh   RQE TCB   hhhhhhhh

PGM     OLD PSW     hhhhhhhh hhhhhhhh   R15/R1   hhhhhhhh hhhhhhhh   VPA      hhhhhhhh   OLT TCB   hhhhhhhh

EXT     OLD PSW     hhhhhhhh hhhhhhhh   R15/R0   hhhhhhhh hhhhhhhh   R1       hhhhhhhh   TQE TCB   hhhhhhhh

SVC     OLD PSW     hhhhhhhh hhhhhhhh   R15/R0   hhhhhhhh hhhhhhhh   R1       hhhhhhhh   OLD TCB   hhhhhhhh

DSP     NEW PSW     hhhhhhhh hhhhhhhh   R15/R0   hhhhhhhh hhhhhhhh   R1       hhhhhhhh   NEW TCB   hhhhhhhh


        TIME sssss.mmmmmm    (records that have timestamps will have the timestamp printed as a continuation line)
```

Figure GTF-10. Format of Minimal Trace Records for DSP, IO (including PCI), SIO, PI, EXT, and SVC.

```
(HEXFORMAT)  AID hh   FID hh   EID hh   hhhhhhhh hhhhhhhh hhhhhhhh hhhhhhhh hhhhhhhh hhhhhhhh
(USER     )                             hhhhhhhh hhhhhhhh hhhhhhhh hhhhhhhh hhhhhhhh hhhhhhhh
(SYSTEM   )                             hhhhhhhh hhhhhhhh hhhhhhhh hhhhhhhh hhhhhhhh hhhhhhhh
(SUBSYS   )                             hhhhhhhh hhhhhhhh hhhhhhhh hhhhhhhh hhhhhhhh hhhhhhhh
                                        hhhhhhhh hhhh...
```

Figure GTF-11. Hexadecimal Format Records.

Control Records

GTF produces two types of control records: timestamp records and lost data records. The first record in every block of trace output is a timestamp record. A lost data record tabulates trace events that were not recorded because the GTF buffers were full or because GTF has temporarily suspended operations during ABEND or SNAP processing. Figure GTF-12 shows the general format of a timestamp record.

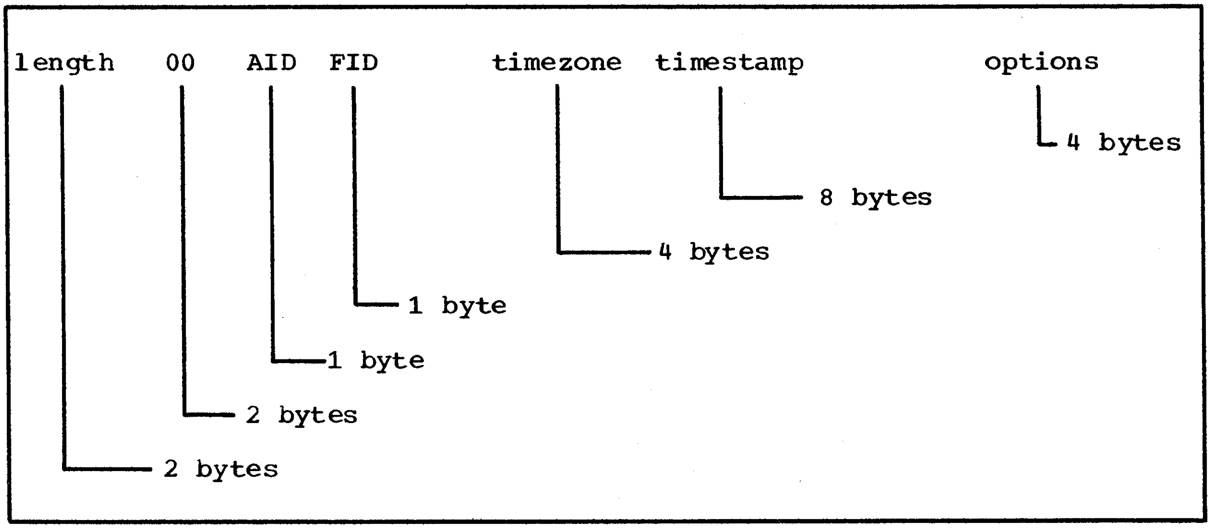


Figure GTF-12. General Format of a Timestamp Control Record.

The fields in the record contain the following information:

length

total length of the record in bytes.

00

always zero.

AID

always zero, for control records.

FID

for timestamp control records, this field is always X'04'.

timezone

factor for converting Greenwich Mean Time to local time.

timestamp

clock value representing the time when the control record was constructed.

options

GTF options in effect. For detailed information about this field, see Figure APNDX-2 in the Appendix.

Figure GTF-13 shows the general format of a lost event record.

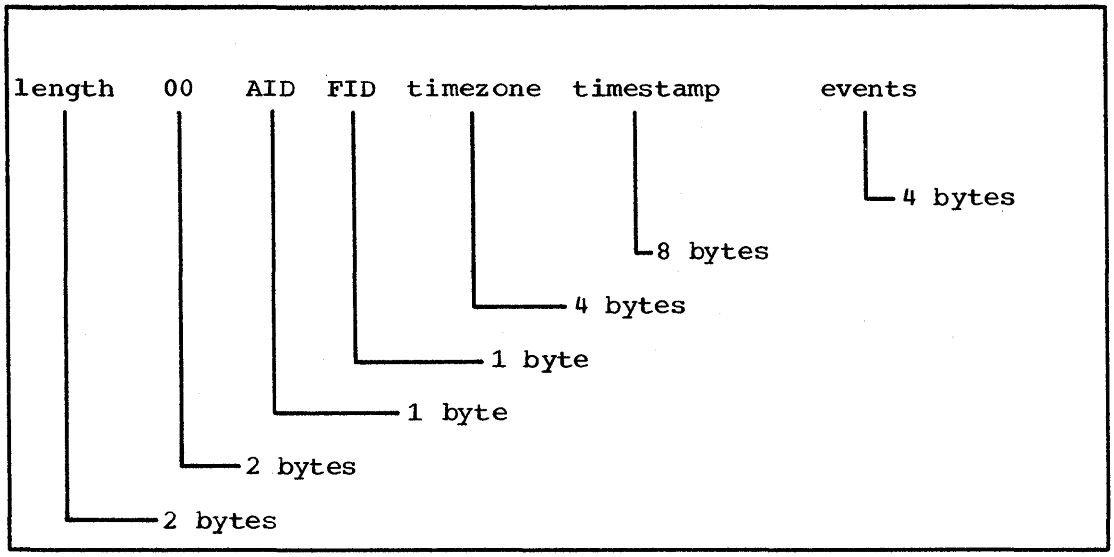


Figure GTF-13. General Format of a Lost Event Record.

The fields in the record contain the following information.

length

total record length in bytes.

00

always zero.

AID

always 00 in control records

FID

format identifier. Valid values are:

X'05' -- events lost because buffers full.

X'06' -- events lost because GTF was temporarily disabled.

timezone

factor for converting Greenwich Mean Time to local time.

timestamp

clock value representing the time when the control record was constructed.

events

number of traceable events lost (in hexadecimal).

Examples

Example 1: Changing the Name of the Trace Data Set

This example shows how to use the START GTF command to change the name of the trace data set from the default name, SYS1.TRACE, to an arbitrary name, OUTPUT.

```
    START GTF,,,(MODE=EXT),DSNAME=OUTPUT
```

Example 2: Directing Trace Output to an Existing Data Set

This example shows how to use the START GTF command to change the disposition of the trace data set from DISP=(NEW,KEEP) to DISP=(OLD,KEEP).

```
    START GTF,,,(MODE=EXT),DISP=OLD
```

If the name of the trace data set is not SYS1.TRACE, specify the name using the DSNAME= parameter, as shown in Example 1.

Example 3: Directing Trace Output to a Tape

This example shows how to use the START GTF command to direct trace output to a data set residing on tape rather than on a direct access device. The DSNAME= parameter changes the name of the output data set from SYS1.TRACE to TPOUTPUT.

```
    START GTF,2400,TRCTAP,(MODE=EXT),DSNAME=TPOUTPUT
```

In this example, the specified tape resides on a 2400 tape drive and has a volume serial of TRCTAP.

Chapter 2: JOBQD

Operates as a stand-alone program to format and print the system job queue. (VS1 Only)

Contents

Figures

Introduction

JOBQD is an OS/VS1 service aid that formats and prints the contents of the system job queue data set (SYS1. SYSJOBQE) and the scheduler work area data sets (SWADS). You may use JOBQD to dump the entire job queue data set, or selected portions of it. You may print out:

- Records in particular work queues, by specifying their QCRs (queue control records).
- Records for a particular job in the input work queue(s), by specifying the job name. To reduce processing time you should specify the QCR as well as the job name if you know the input queue to which the job is assigned.

Similarly you may dump all SWADS, or only selected SWADS by specifying their initiator procedure names.

Detailed descriptions and layouts of the record types found in the job queue data set and SWADS are given in the publication OS/VS1 Job Management Logic, SY24-5161.

The Job Queue Dump program is a stand-alone program. Since this program does not function under OS/VS, it is not enqueued on the job queue and, therefore, does not alter the existing status of the records that are to be dumped. The printed queue records reflect precisely what they contained at the time of malfunction.

You do not need to know the explicit address of the job queue data set (SYS1.SYSJOBQE), you need to know only the address assigned to the direct access device on which the volume containing the job queue is mounted. The dump program finds the address by reading the VTOC (volume table of contents) on the volume mounted on the device you specify. When JOBQD finds the job queue data set, it identifies, formats, and writes the records on the selected output device in accordance with the options you select.

Thus JOBQD is designed to supply you with specialized job queue and scheduler work area dumps:

- Without disturbing their prevailing status
- Whether or not the system is operational
- Without prior knowledge of the exact location of the job queue or scheduler work area data set on the assigned direct access volume
- On a record-by-record basis, according to direct access volume address
- Conveniently formatted for ready access and interpretation

Figure JOBQD-1 shows the flow of processing for JOBQD.

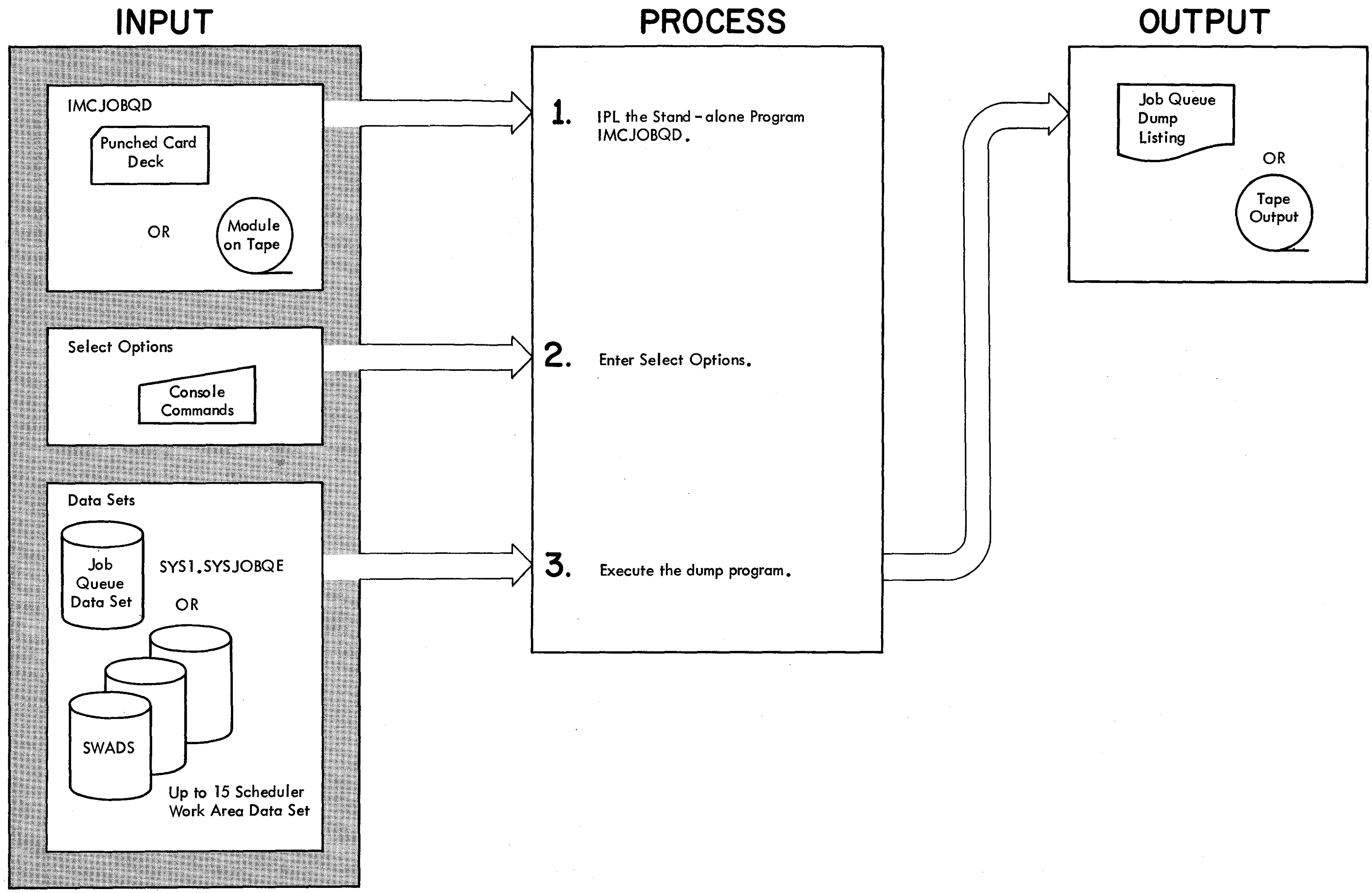


Figure JOBQD-1. Devices and Flow of Processing for JOBQD

Selecting I/O Units

ro execute JOBQD, you will require about 25K bytes of real storage. You will also need the following I/O devices.

- For initial program loading (IPL): a card reader or, optionally, a tape drive (IBM 2400-series or 3420)

- For system-operator communication: a console printer-keyboard (IBM 1052, 3210, or 3215)

- For input: a DASD device (IBM 2305, 2314, or 3330)

- For output: a printer (IBM 1403, 1443, or 3211) or tape drive (IBM 2400-series or 3420). If you need readable output immediately, you should direct JOBQD output to a printer. Otherwise, you should direct output to a tape and have the tape printed later.

Note for VS1: If intervention required is detected on the output device, the message IMC013A is issued. After the device is made ready, I/O is reissued with the possibility that one line of output may be duplicated.

Retrieving JOBQD

You will receive the Job Queue Dump program (JOBQD) in object module form, together with an absolute loader, on Distribution Library pack. It is composed of two members (named IMCJQAPP and IMCJQMCI) in distribution library SYS1.ASAMPLIB. Because this program will be executed as a stand-alone program, you must

- punch it into a card deck

  or

- copy it onto an unlabelled magnetic tape.

Figure JOBQD-2 shows the JCL statements that you can use for the IEBPTPCH utility program, to punch the program from the Distribution library into a card deck.

```
//QDUMP        JOB       MSGLEVEL=(1,1)
//STEP         EXEC      PGM=IEBPTPCH
//SYSPRINT     DD        SYSOUT=A
//SYSUT1       DD        DSN=SYS1.ASAMPLIB,DISP=OLD,
//        DCB=(BLKSIZE=3600,LRECL=80,RECFM=FB)
//SYSUT2       DD        UNIT=2540-2
//SYSIN        DD        *
          PUNCH     TYPORG=PO,MAXNAME=2
          MEMBER    NAME=IMCJQAPP
          MEMBER    NAME=IMCJQMCI
/*
```

Figure JOBQD-2. An Example of JCL Statements for Punching JOBQD From Distribution Library SYS1.ASAMPLIB

SYSUT1 DD statement

defines the distribution library data set and assumes that the distribution libraries are cataloged; if they are not, add these two parameters to the SYSUT1 definition statement:

UNIT=devicetype

VOL=SER=DLIB03

SYSUT2 DD statement

provides for punching the stand-alone JOBQD program into a card deck. If you want to write JOBQD on tape, replace the 2540 designation with:

UNIT=2400

Executing JOBQD

Follow this procedure.

1. Ready the input/output devices:

   - Card reader or tape drive from which JOBQD will be loaded.
   - Console printer-keyboard for system-operator messages.
   - Input DASD device where the job queue data set is stored.
   - Output printer or output tape drive, whichever you are using.

2. If you use output tape, provide for standard labels or for an unlabeled tape.

   For standard labels:

   - Mount a tape volume that already has a standard volume label (VOL1) as the first record on the tpe.
   - Be sure that the density of the volume label is the same as the density set for the tape drive.

   For unlabeled tape:

   - Mount an unlabeled tape if possible,
   - or, if you must use a labeled tape, make sure the recording density set now for the tape drive differs from the density used previously to write the label.

3. Load JOBQD.

4. Press REQUEST on the console printer-keyboard.

5. This message is printed:

   ```
   IMC000A   ENTER 0=XXXD,Q=YYY(,S) OR PRESS INTERRUPT KEY FOR O-00E,
             Q=151
   ```

6. Respond by:

   - Pressing INTERRUPT if you want default devices (disk drive 151 for input, and printer 00E for output) and a full dump. See Example 1.
   - Entering only the device addresses

     ```
     O=xxxd,Q=yyy
     ```

     if you want to specify I/O devices and have a full dump. See Example 2.

- Entering the device addresses and selection

  ```
  O=xxxd,Q=yyy,5
  ```

  if you want a selective dump.

  Example 3 shows the procedure to specify device addresses and dump the records in a specific work queue or input job.

  Example 4 shows the procedure to specify device addresses and dump SWADS records.

  Example 5 shows the procedure to specify device addresses and dump the records in a specific work queue or input job followed by SWADS records.

7. If in step 6 you specified a tape output device , JOBQD checks the mounted tape for IBM standard labels. If standard labels are not found, JOBQD writes an unlabeled tape. If standard labels are found, JOBQD processes standard labels by

   - Requesting a new tape if USASCII has been used for the volume label.

   - Ignoring and destroying any user labels.

   - Requesting a new tape if the mounted tape contains a security-protected data set.

   - Checking the expiration date in the HDR1 label, and requesting permission to use the tape if the data has not yet occurred. If your reply is:

     M (Mount), it requests that you mount a new tape.

     U (Use), it uses the mounted tape, retaining the VOL1 label.

   - Creating and writing standard header and trailer labels, with a data set name of JQDUMP.

8. After all specified records are dumped, the message IMC004I DUMP COMPLETED is printed and program execution ends.

Example 1: Requesting Default Devices and a Full Dump

In this example your reply to message IMC000A will

- specify input queue device 151
- specify output printer 00E
- dump all job queue data set records
- allow you to dump SWADS.

Follow these steps:

1. Reply to message IMC0000A by pressing INTERRUPT on the system control panel.

2. This message is printed:

```
IMC020A   ENTER INITIATOR PROC NAMES FOR WHICH SWADS IS TO BE DUMPED
```

3. Respond by:

   - Entering up to 4 initiator procedure names - if you want any SWADS printed out.

     Example: INIT,INITD

   - Pressing the END or EOB key - if you do not want to dump any SWADS.

4. The job queue data set records are dumped.

5. A mount message (M) tells you to mount the disk pack containing the first SWADS to be dumped:

```
IMC022A   M    ,volid,,initname,qual AND ENTER DEVICE ADDRESS OR
          ENTER CANCEL
```

   The disk pack is identified by:

   volid - DASD volume identifier

   initname - initiator procedure name

   qual - qualifier used in starting the initiator

6. Respond by:

   - Mounting the specified disk pack.

   - Entering 3-position hexadecimal address of the disk drive you used.

     Example: 133

     If you use the IBM 2305, merely enter the hexadecimal address, since there is no mounting to be done.

7. The first SWADS is dumped.

8. The mount message is printed again if JOBQD determines that there is another SWADS to be dumped.

9. Respond:

   - As in step 6 for another SWADS, or
   - By entering C or CANCEL if you do not want this SWADS dumped.

10. Repeat steps 8 and 9.

11. After all SWADS have been printed out, the program gives you the message IMC004I DUMP COMPLETED and terminates.

Example 2. Specifying I/O Devices and Requesting a Full Dump

In this example, you will reply to message IMC000A by:

- first, giving the address of the output device
- second, giving the address of the input queue device.

This will:

- cause a full dump of the job queue data set
- allow you to dump SWADS.

JOBQD

Follow these steps:

1. Respond to message IMC000A by:

   - Entering 0=xxxd for the output device

     Example: 0=282T (for tape unit indicated by T) with address 282.

     0=00F for printer with address 00F.

   - Omitting 0= xxxd if you use the default printer (with address 00E).

   - Entering Q=yyy for the input queue device.

     Example: Q=152 for a disk drive with address 152.

2. This message is printed:

   ```
   IMC020A   ENTER INITIATOR PROC NAMES FOR WHICH SWADS IS TO BE DUMPED
   ```

3. Respond by:

   - Entering up to 4 initiator procedure names if you want any SWADS printed out.

     Example: INIT,INITD

   - Pressing the END or EOB key if you do not want to dump any SWADS.

4. The job queue data set records are dumped.

5. A mount message (M) tells you to mount the disk pack containing the first SWADS to be dumped:

   ```
   IMC022A   M   ,volid,,initname,qual AND ENTER DEVICE ADDRESS OR
             ENTER CANCEL
   ```

   The disk pack is identified by:

   volid - DASD volume identification

   initname - initiator procedure name

   qual - qualifier used in starting the initiator

6. Respond by:

   - Mounting the specified disk pack.
   - Entering the 3-position hexadecimal address of the disk drive you used.

     Example: 133

     If you use the IBM 2305, merely enter the hexadecimal address, since there is no mounting to be done.

7. The first SWADS is dumped.

8. The mount message is printed again, if JOBQD determines that there is another SWADS to be dumped.

9. Respond:

   - As in step 6 for another SWADS, or
   - By entering C or CANCEL if you do not want this SWADS dumped.

10. Repeat steps 8 and 9.

11. After all SWADS have been printed out, the program gives you the message IMC004I DUMP COMPLETED and terminates.

Example 3: Specifying I/O Devices and Dumping Records of a Specific Work Queue or Job

In this example, you will reply to message IMC000A by:

- first, giving the address of the output device
- second, giving the address of the input queue device
- third, indicating that you will select the records to be dumped.

This will allow you to specify the

- particular work queues to be dumped
- particular input jobs to be dumped.

Follow these steps:

1. Respond to message IMC000A by:
   - Entering 0=xxxd for the output device

     Example: 0=282T for tape unit (indicated by T) with address 282

     0=00f for printer with address 00F.
   - Omitting o=xxxd if you use the default printer (with address 00E).
   - Entering Q=yyy for the input queue device.
   - Entering S (or SELECT) for dump selection.
2. This message is printed:

   IMC001 SPECIFY SELECT PARAMETERS
3. Respond by:

Entering	To dump the
QCR=CLASS=y	Records in one of the 15 job class input queues. Replace "y" with the desired job class code (A-O).
QCR=SYSOUT=x	Records in one of the 36 output class work queues. Replace x with the desired output class code (A-Z,0-9)
QCR=FREE	Tracks in the free-track queue
QCR=HOLD	Jobs in the hold queue
JOBNAME=(name1...,name4)	Records for a particular job in the input queues. Specify up to 4 job names. Example: JOBNAME=(TAX,NUMBER, PAYROLL,UPDATE)
QCR=CLASS=y,JOBNAME=name	The records for a particular job, and you know which job class is assigned. Example: QCR=CLASS=J,JOBNAME=(TAX) The job named TAX is assigned to class J input.

4. The records you requested are printed.

5. Message IMC001A SPECIFY SELECT PARAMETERS is printed again.

6. Respond by repeating step 3, to dump records for another queue or job. Steps 4 and 5 are repeated.

7. Repeat steps 4-6 until you have dumped all desired work queues and input jobs.

8. Respond to message IMC001A by entering END to terminate the job. The message IMC004I DUMP COMPLETED is printed.

Example 4: Specifying I/O Devices and Dumping Specific SWADS

In this example, you will reply to message IMC000A by:

- first, giving the address of the output device
- second, giving the address of the input queue device
- third, indicating that you will select the records to be dumped.

This will allow you to specify the particular SWADS to be dumped, by giving the initiator procedure names qualified by the partition identifiers.

Follow these steps:

1. Respond to message IMC000A by:

    - Entering O=xxxd for the output device.

      Example: O=282T for tape unit (indicated by T) with address 282.

      O=00F for printer with address 00F.

    - Omitting O=xxxd if you use the default printer (with address 00E).

    - Entering Q=yyy for the input queue device.

      Example: Q=152 for disk drive with address 152.

    - Entering S (or SELECT) for dump selection.

2. This message is printed:

    IMC001A SPECIFY SELECT PARAMETERS

3. Respond by Entering in a maximum of 80 characters:

    SWADS=(prOCname.IPL,procname.id2...,procname.idn)

4. A mount message (M) tells you to mount the disk pack containing the first SWADS to be dumped.

    Example:

    IMC022A M ,volid,,initname,qual AND ENTER DEVICE ADDRESS OR CANCEL

    The disk pack is identified by:

    volid - DASD volume identification

    initname - initiator procedure name

    qual - qualifier used in starting the initiator

5. Respond by:

    - Mounting the specified disk pack.

    - Entering the 3-position hexadecimal address of the disk drive you used.

      Example: 133

If you use the IBM 2305, merely enter the hexadecimal address, since there is no mounting to be done.

6. The first SWADS you specified (step 3) is dumped.

7. Ihe mount message is printed again, if JOBQD determines that there is another SWADS to be dumped.

8. Respond:

   - as in step 5 for another SWADS, or
   - by entering C or CANCEL if you do not want this SWADS dumped.

9. Repeat steps 7 and 8.

10. If JOBQD fails to find a specified SWADS, this message is printed:

    ```
    IMC021I   UNABLE TO FIND SWADS FOR procname.id
    ```

    and processing continues with the next SWADS.

11. After all the specified SWADS are dumped, the message IMC004I DUMP COMPLETED is printed and the program ends.

Example 5: Specifying I/O Devices and Dumping Records of a Specific Work Queue or Job Followed by SWADS

This example is a combination of example 3 and 4. The only requirement in combining the two is that you must respond to the message

```
        IMC001A  SPECIFY SELECT PARAMETERS
```

by giving each QCR= or JOBNAME= specification before you give any SWADS= specification. The program terminates automatically after the last SWADS you specify is dumped.

Thus you will follow these steps:

- Example 3, steps 1-7
- Example 4, steps 3-11

Printing Dump Tapes

If you specify tape as your output device when executing JOBQD, you may write on either 9-track or 7-track tape. The dump program creates 121-byte unblocked records, each containiing a printer control character in its first byte.

An example of the JCL statements you need for the IEBPTPCH program when you transfer from 9-track tape to printer is shwon in Figure JOBQD-3.

```
//PRINT        JOB        1234,SMITH,MSGLEVEL=(1,1)
//STEP         EXEC       PGM=IEBPTPCH
//SYSPRINT     DD         SYSOUT=A
//SYSUT1       DD         UNIT=2400,LABEL=(,NL),VOL=SER=QDUMPT,
//         DISP=(OLD,KEEP),DCB=(REDFM=F,BLKSIZE=121,LRECL=121),
//SYSUT2       DD         SYSOUT=A
//SYSIN        DD         *
         PRINT      PREFORM=M
/*
```

Figure JOBQD-3. Example of JCL for Printing Records Dumped on 9-Track Tape

If the records are written on 7-track tape, include these additional DCB parameter(s) in the SYSUT1 DD statement:

- DEN=2 — if the output tape control unit did not have the data conversion feature.

- DEN=2,TRTCH=C — if the output tape control unit did have the data conversion feature.

These changes are required because the initial loading of the dump program generated a system reset, which had the following effect on the 7-track tape control unit:

1. Density was set to 800 bytes per inch.

2. If the data conversion feature was present in the control unit, the data converter was turned off.

3. The translator was turned off.

4. Odd parity was established.

JOBQD Output

You control the job queue dump program (see Executing JOBQD) to print:

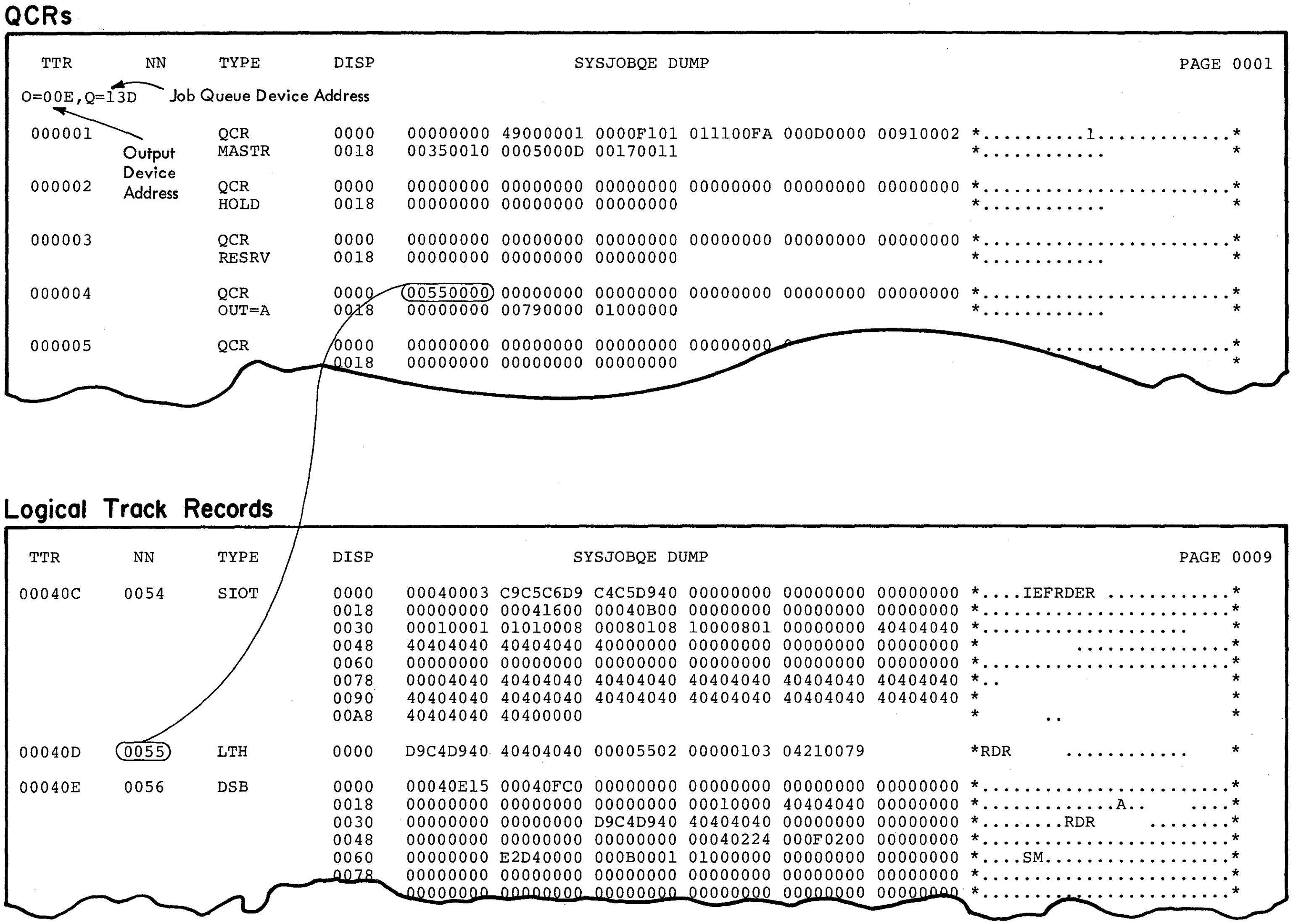
- all records in the job queue data set (SYS1,SYSJOBQE)
- all SWADS records
- selected portions of the job queue data set
- selected SWADS.

If you print the entire job queue data set (Figure JOBQD-4) and SWADS (Figure JOBQD-5), the output is printed in sections:

1st section - all 75 queue control records (QCRs)

2nd section - all job control records in the logical track area of the job queue data set

a section for each SWADS - all records in each SWADS.



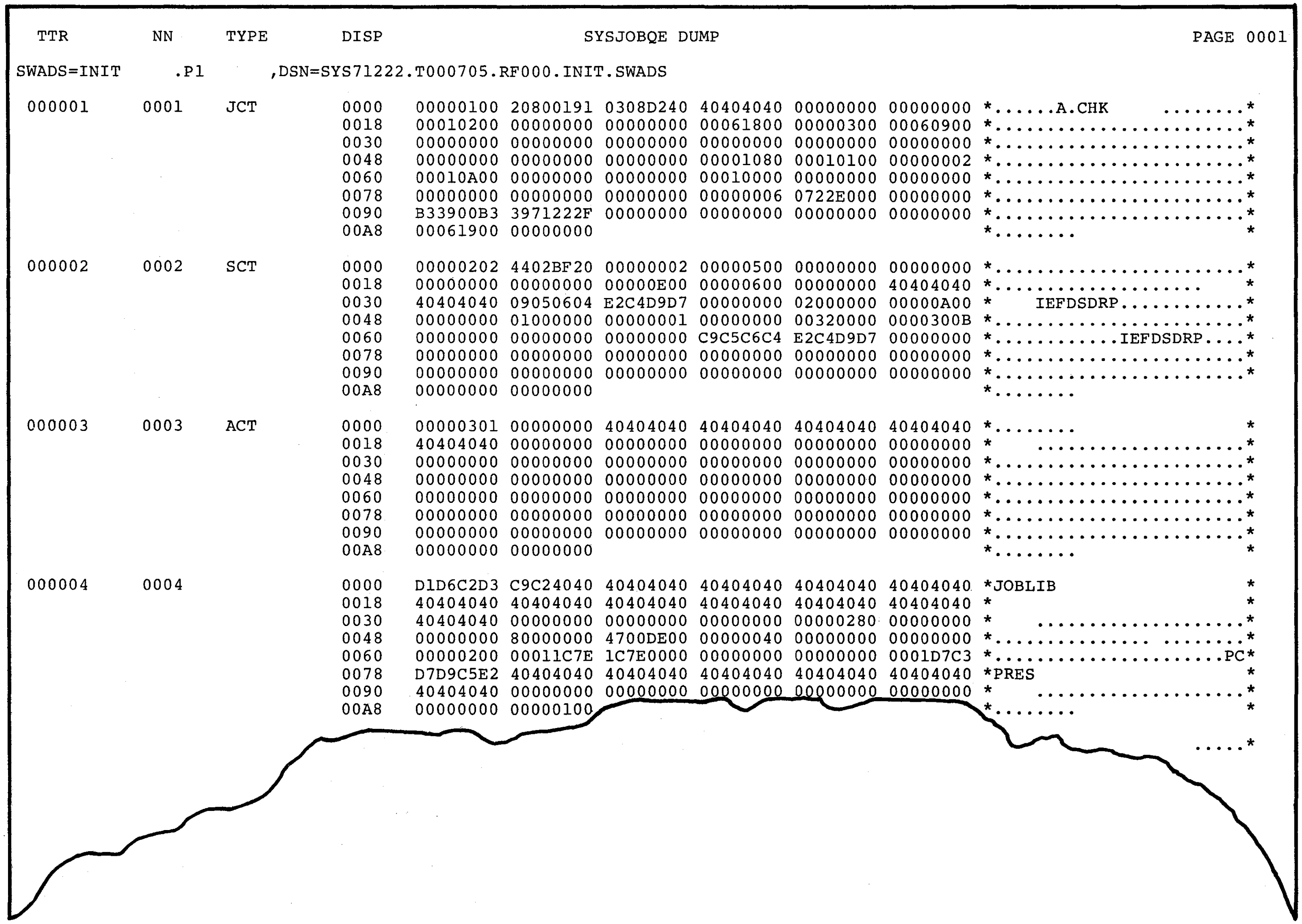
QCRs

```
  TTR        NN      TYPE      DISP                    SYSJOBQE DUMP                                                PAGE 0001
O=00E,Q=13D
 000001              QCR       0000    00000000 49000001 0000F101 011100FA 000D0000 00910002 *..........1.............*
                     MASTR     0018    00350010 0005000D 00170011                            *............            *
 000002              QCR       0000    00000000 00000000 00000000 00000000 00000000 00000000 *........................*
                     HOLD      0018    00000000 00000000 00000000                            *............            *
 000003              QCR       0000    00000000 00000000 00000000 00000000 00000000 00000000 *........................*
                     RESRV     0018    00000000 00000000 00000000                            *............            *
 000004              QCR       0000    00550000 00000000 00000000 00000000 00000000 00000000 *........................*
                     OUT=A     0018    00000000 00790000 01000000                            *............            *
 000005              QCR       0000    00000000 00000000 00000000 0000000                          ...................*
                               0018    00000000 00000000 00000000                                                     *
```

Logical Track Records

```
 TTR       NN       TYPE      DISP                    SYSJOBQE DUMP                                                PAGE 0009
00040C     0054     SIOT      0000    00040003 C9C5C6D9 C4C5D940 00000000 00000000 00000000 *....IEFRDER ............*
                              0018    00000000 00041600 00040B00 00000000 00000000 00000000 *........................*
                              0030    00010001 01010008 00080108 10000801 00000000 40404040 *....................    *
                              0048    40404040 40404040 40000000 00000000 00000000 00000000 *          ..............*
                              0060    00000000 00000000 00000000 00000000 00000000 00000000 *........................*
                              0078    00004040 40404040 40404040 40404040 40404040 40404040 *..                      *
                              0090    40404040 40404040 40404040 40404040 40404040 40404040 *                        *
                              00A8    40404040 40400000                                     *        ..              *
00040D     0055     LTH       0000    D9C4D940 40404040 00005502 00000103 04210079          *RDR     ............    *
00040E     0056     DSB       0000    00040E15 00040FC0 00000000 00000000 00000000 00000000 *........................*
                              0018    00000000 00000000 00000000 00010000 40404040 00000000 *............A..     ....*
                              0030    00000000 00000000 D9C4D940 40404040 00000000 00000000 *........RDR     ........*
                              0048    00000000 00000000 00000000 00040224 000F0200 00000000 *........................*
                              0060    00000000 E2D40000 000B0001 01000000 00000000 00000000 *....SM..................*
                              0078    00000000 00000000 00000000 00000000 00000000 00000000 *........................*
                                      00000000 00000000 00000000 00000000 00000000 00000000 *........................*
```

Figure JOBQD-4. Sample Dump of Entire Job Queue Data Set



```
 TTR         NN      TYPE        DISP                    SYSJOBQE DUMP                                                 PAGE 0001

SWADS=INIT      .P1       ,DSN=SYS71222.T000705.RF000.INIT.SWADS

 000001      0001    JCT         0000    00000100 20800191 0308D240 40404040 00000000 00000000 *......A.CHK     ........*
                                 0018    00010200 00000000 00000000 00061800 00000300 00060900 *........................*
                                 0030    00000000 00000000 00000000 00000000 00000000 00000000 *........................*
                                 0048    00000000 00000000 00000000 00001080 00010100 00000002 *........................*
                                 0060    00010A00 00000000 00000000 00010000 00000000 00000000 *........................*
                                 0078    00000000 00000000 00000000 00000006 0722E000 00000000 *........................*
                                 0090    B33900B3 3971222F 00000000 00000000 00000000 00000000 *........................*
                                 00A8    00061900 00000000                                     *........                *

 000002      0002    SCT         0000    00000202 4402BF20 00000002 00000500 00000000 00000000 *........................*
                                 0018    00000000 00000000 00000E00 00000600 00000000 40404040 *....................    *
                                 0030    40404040 09050604 E2C4D9D7 00000000 02000000 00000A00 *    IEFDSDRP............*
                                 0048    00000000 01000000 00000001 00000000 00320000 0000300B *........................*
                                 0060    00000000 00000000 00000000 C9C5C6C4 E2C4D9D7 00000000 *............IEFDSDRP....*
                                 0078    00000000 00000000 00000000 00000000 00000000 00000000 *........................*
                                 0090    00000000 00000000 00000000 00000000 00000000 00000000 *........................*
                                 00A8    00000000 00000000                                     *........

 000003      0003    ACT         0000    00000301 00000000 40404040 40404040 40404040 40404040 *........                *
                                 0018    40404040 00000000 00000000 00000000 00000000 00000000 *    ....................*
                                 0030    00000000 00000000 00000000 00000000 00000000 00000000 *........................*
                                 0048    00000000 00000000 00000000 00000000 00000000 00000000 *........................*
                                 0060    00000000 00000000 00000000 00000000 00000000 00000000 *........................*
                                 0078    00000000 00000000 00000000 00000000 00000000 00000000 *........................*
                                 0090    00000000 00000000 00000000 00000000 00000000 00000000 *........................*
                                 00A8    00000000 00000000                                     *........                *

 000004      0004                0000    D1D6C2D3 C9C24040 40404040 40404040 40404040 40404040 *JOBLIB                  *
                                 0018    40404040 40404040 40404040 40404040 40404040 40404040 *                        *
                                 0030    40404040 00000000 00000000 00000000 00000280 00000000 *    ....................*
                                 0048    00000000 80000000 4700DE00 00000040 00000000 00000000 *............... ........*
                                 0060    00000200 00011C7E 1C7E0000 00000000 00000000 0001D7C3 *......................PC*
                                 0078    D7D9C5E2 40404040 40404040 40404040 40404040 40404040 *PRES                    *
                                 0090    40404040 00000000 00000000 00000000 00000000 00000000 *    ....................*
                                 00A8    00000000 00000100                                     *........                *

                                                                                                                   .....*
```

Figure JOBQD-5. Sample Dump of SWADS

If you print selectively, you may request specific work queues (such as an input queue) within the job queue data set, specific job names, or specific scheduler work areas.

For a specific work queue (specified by QCR=), the printed output is the:

- master QCR
- QCR you specified
- the work queue records

For a specific job name (specified by JOBNAME=), the printed output includes the:

- job name you specified
- master QCR
- Input QCRs up to and including the QCR for the input class of the job requested
- records associated with that job, which are collected and printed together

If the job(s) you request cannot be found, a message (IMC006I) and the job name(s) are printed.

For a specific scheduler work area (specified by SWADS=), the printed output includes:

- a header line that gives the initiator identifier and the system-assigned data name of SWADS
- records containing scheduler work area tables

JOBQD prints the records in hexadecimal representation, with six 4-byte words appearing in a line of printed output. It also prints EBCDIC letters and digits in a one-character-per-byte format at the right end of the printline. Other EBCDIC characters (except blanks) are represented by periods. Records in each logical track are read and dumped sequentially. When a record contains binary zeroes only, its TTR and NN positions are given, but the record is not dumped. Instead, the comment

```
    ENTIRE RECORD CONTAINS BINARY ZEROES
```

is printed on the listing. Any succeeding zero records are bypassed until a nonzero record or a logical track header, whichever occurs first, is encountered. Then the TTR and NN of the last zero-filled record and the message

```
    ZERO RECORDS SUPPRESSED
```

are printed.

The headings at the top of the listing identify the records (in either a job queue or SWADS dump) as follows:

TTR

The direct access address (track and record) relative to the beginning of the job queue data set or a SWADS. It is supplied for both QCR and logical track records.

NN

The record number assigned relative to the beginning of the logical track area. The first logical track header (LTH) record is numbered 1.

TYPE

The type of job queue record. Each queue control record (QCR) is further identified with the type of work queue with which it is associated. Details about record types and the types of work queues are given in the OS/VS1 Debugging Guide, GC24-5093.

Finding the Reason for an Abnormal Termination of JOBQD

If the job queue dump program enters the wait state before its normal completion, and no error message is issued, the error condition may be caused by:

- Unrecoverable I/O error. JOBQD issues an SVC. This may be the same error condition as the one that caused the system to malfunction in the first place. That is, it may be an I/O error on the queue device or an invalid chaining of queue records.
- Invalid data detected by the job queue dump program. JOBQD issued an SVC.
- Program check caused by invalid data.
- Machine check.

Determine the type of error and its cause by examining the content of the PSW (Figure JOBQD-6) that was stored when the error condition was discovered. To do this:

1. Refer to the wait lights on the system control panel to find out which old PSW is involved. The two low-order bytes contain 0Dnn, where nn is the address, in hex, of the doubleword where the old PSW is stored.

   The address of the old PSW and the error condition that was detected are:

Address	Error Condition
0D20	Unrecoverable I/O error or invalid data detected by JOBQD, and SVC issued
0D28	Program check caused by invalid data
0D30	Machine check

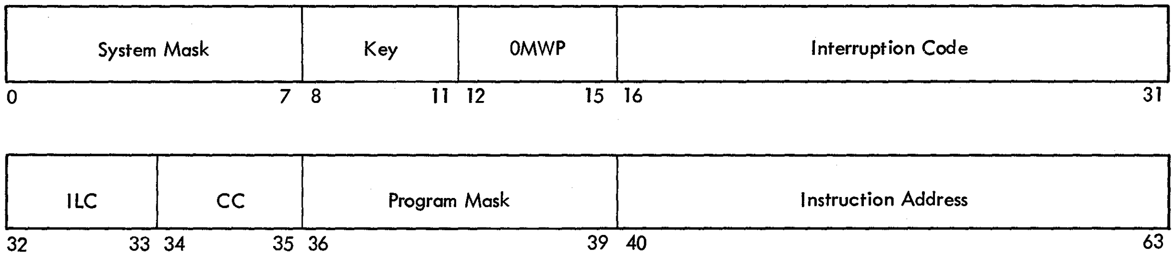


Figure JOBQD-6. Format of PSW (Program Satus Word)

2. For a program-detected error condition (lights contain 0D20) locate the nature of the error by inspecting the interruption code in the old PSW (at location hex 20). The possible codes and their meanings are:

Interruption Code	Error Cause
X'00'	Channel end, device end, and unit check bits are all off in a stored channel status word (CSW).
X'02'	Invalid track-per-cylinder count in the format 4 DSCB (data set control block) of the queue volume.
X'03'	I/O error during write operation to output device or system console. The number of retries for recoverable tape I/O errors is set at 20.
X'20'	I/O error during read operation from SYS1.SYSJOBQE data set or SWADS. The number of retries for recoverable DASD I/O errors is set at 16.
X'26'	I/O error during read operation from system console.

Message	Request	Your Response
IMC000A	Devices and selection	$0=\left\{\begin{matrix}\text{00E}\\ \text{xxx[T]}\end{matrix}\right\}\text{,Q=}\left\{\begin{matrix}\text{151}\\ \text{yyy}\end{matrix}\right\}\left[\begin{matrix}\text{,S}\\ \text{,SELECT}\end{matrix}\right]$
IMC020A	Initiator procedure names (1-4 names)	[procname1,procname2,procname3,procname4]
IMC022A	Device address of SWADS pack	$\left\{\begin{matrix}\text{cuu}\\ \text{C}\\ \text{CANCEL}\end{matrix}\right\}$
IMC001A	Selection(S) parameters	One of the following: QCR=CLASS=y (y is input job class A-O) QCR=SYSOUT=x(x is output class A-Z,0-9) QCR=FREE QCR=HOLD
	(1-4 names)	JOBNAME=(name1,name2,name3,name4)
	(1-4 names)	QCR=CLASS=y,JOBNAME=(name1,name2, name3,name4)
		END
	(total chars= 80)	SWADS=(procname.id1 . . .,procname.idn)

Figure JOBQD-7. Message Response Reference

Chapter 3: LIST

Formats and prints object modules, load modules, and CSECT identification records. Maps nucleus and link pack area.

Contents

Figures

Introduction

LIST is a service aid that operates as a problem program under OS/VS. It produces several kinds of output that you need to perform certain diagnostic functions, such as:

Verifying an object module. LIST produces a formatted listing that contains the external symbol dictionary (ESD), the relocation dictionary (RLD), the text of the program containing instructions and data, and the END record.

Mapping CSECTs in a load module. LIST produces a listing of the load module along with its module map and cross-reference listing, which you can examine to determine the organization of CSECTs within the load module, the overlay structure, and the cross-references for each CSECT.

Verifying the contents of the nucleus. LIST can produce a map and cross-reference listing of a nucleus.

Tracing modifications to the executable code in a CSECT. LIST produces a formatted listing of all information in a load module's CSECT identification records (IDRs). An IDR provides the following information:

- It identifies the version and modification level of the language translator and the date that each CSECT was translated. (Translation data is available only for CSECTs that were produced by a translator that supports IDR generation.)

- It identifies the version and modification level of the linkage editor that built the load module and gives the date the load module was created.

- It identifies by date modifications to the load module that may have been performed by SPZAP.

An IDR may also contain optional user-supplied data associated with the executable code of the CSECTs.

Mapping the link pack area. LIST produces a map of all modules in the reenterable load module area or the link pack area.

JCL Statements

LIST requires the following JCL statements:

JOB Statement

initiates the job.

EXEC Statement

calls for the execution of HMBLIST (on VS1) or AMBLIST (in VS2).

SYSPRINT DD Statement

defines the message data set.

anyname DD Statement

defines an input data set.

SYSIN DD Statement

defines the data set (in the input stream) that contains LIST control statements.

Control Statements

You control LIST processing by supplying control statements in the input stream. You must code the control statements according to the following rules:

- Leave column 1 blank, unless you want to supply an optional symbolic name. A symbolic name must be terminated by one or more blanks.

- If a complete control statement will not fit on a single card, end the first card with a comma or a non-blank character in column 72 and continue on the next card. Begin all continuation cards in columns 2 - 16. You must not split parameters between two cards; the only exception is the MEMBER parameters, which may be split at any internal comma.

The control statements and their parameters are:

```
LISTLOAD [OUTPUT={MODLIST}] [,TITLE=('title',position)]
         [       { XREF  }]
         [       { BOTH  }]
    [,DDN= ddname] [,MEMBER= {(list,...) }] [,RELOC=hhhhhh]
                   [         { membername}]
```

OUTPUT=type

specifies the type of load module listing to be produced. OUTPUT=MODLIST requests a formatted listing of the control and text records of a load module, including its External Symbol Dictionary and Relocation Dictionary Records. OUTPUT=XREF requests a module map and cross-reference listing for the load module. OUTPUT=BOTH requests both a formatted listing of the load module and its map and cross-references. If this parameter is omitted, OUTPUT=BOTH will be assumed.

TITLE=('title',position)

specifies a title, from one to forty characters long, to be printed below the heading line on each page of output. (The heading line identifies the page number and the type of listing being printed, and is not subject to user control.) The position subparameter specifies whether or not the title should be indented; if TITLE=('title',1) is specified, or if the position parameter is omitted, the title will be printed flush left, that is, starting in the first column. If you want the title indented from the margin, use the position parameter to specify the number of characters that should be left blank before the title. Note: Do not punctuate your title with commas: since LIST recognizes a comma as a delimiter, anything that follows an embedded comma in a title will be ignored.

DDN=ddname

identifies the DD statement that defines the data set containing the input module. If the DDN= parameter is omitted, LIST will assume SYSLIB as the default ddname.

```
MEMBER=⎧(member1,...membern)⎫
       ⎩member              ⎭
```

identifies the input load module(s) by membername or alias name. To specifiy more than one load module, enclose the list of names in parentheses and separate the names with commas. If you omit the MEMBER= parameter, LIST will print all modules in the data set.

RELOC=hhhhhh

specifies a relocation or base address in hexadecimal of up to six characters. When the relocation address is added to each relative map and cross-reference address, it gives the absolute main storage address for each item on the output listing. If you omit the RELOC= parameter no relocation is performed.

```
LISTOBJ  [TITLE=('title',position)
         [,DDN=ddname]⎡,MEMBER=⎧(member1,...membern)⎫⎤
                      ⎣        ⎩   member           ⎭⎦
```

TITLE=('title',position)

specifies a title, from one to forty characters long, to be printed below the heading line on each page of output. (The heading line identifies the page number and the type of listing being printed, and is not subject to user control.) The position parameter specifies whether or not the title should be indented; if TITLE=('title',1) is specified, or if the position parameter is omitted, the title will be printed flush left, that is, starting in the first column. If you want the title indented from the margin, use the position parameter to specify the number of characters that should be left blank before the title. Note: Do not punctuate your title with commas: since LIST recognizes a comma as a delimiter, anything that follows an embedded comma in a title will be ignored.

DDN=ddname

identifies the DD statement that defines the data set containing the input module. If the DDN= parameter is omitted, LIST will assume SYSLIB as the default ddname.

```
MEMBER=⎧(member1,...membern)⎫
       ⎩member              ⎭
```

identifies the input object module(s) by membername or alias name. To specify more than one object module, enclose the list of names in parentheses and separate the names with commas. CAUTION: You must include the MEMBER= parameter if the input object modules exist as members in a partitioned data set. If you do not include the MEMBER= parameter, LIST will assume that the input data set is organized sequentially and that it contains a single, continuous object module.

```
LISTIDR [OUTPUT=(IDENT)][,TITLE=('title',position)]
        [       (ALL  )]
        [,DDN=ddname][,MEMBER=((member1,...membern))]
                              (  member            )
```

OUTPUT= type

specifies whether LIST should print all CSECT identification records or only those containing HMASPZAP data and user data. If you specifiy OUTPUT=ALL, all IDRs associated with the module will be printed. If you specify OUTPUT=IDENT, LIST will print only those IDRs that contain SPZAP data or user-supplied data. If you omit this parameter, LIST will assume a default of OUTPUT=ALL.

TITLE=('title',position)

specifies a title, from one to forty characters long, to be printed below the heading line on each page of output. (The heading line identifies the page number and the type of listing being printed, and is not subject to user control.) The position parameter specifies whether or not the title should be indented; if TITLE=('title',1) is specified, or if the position parameter is omitted, the title will be printed flush left, that is, starting in the first column. If you want the title indented from the margin, use the position parameter to specify the number of characters that should be left blank before the title. Note: Do not punctuate your title with commas: since LIST recognizes a comma as a delimiter, anything that follows an embedded comma in a title will be ignored.

DDN=ddname

identifies the DD statement that defines the data set containing the input module. If you omit the DDN= parameter, LIST will assume SYSLIB as the default ddname.

```
MEMBER= ((member1,...membern))
        (  member            )
```

identifies the input load module(s) by membername or alias name. To specify more than one load module, enclose the list of names in parentheses and separate the names with commas. If you omit the MEMBER= parameter, LIST will print all modules in the data set.

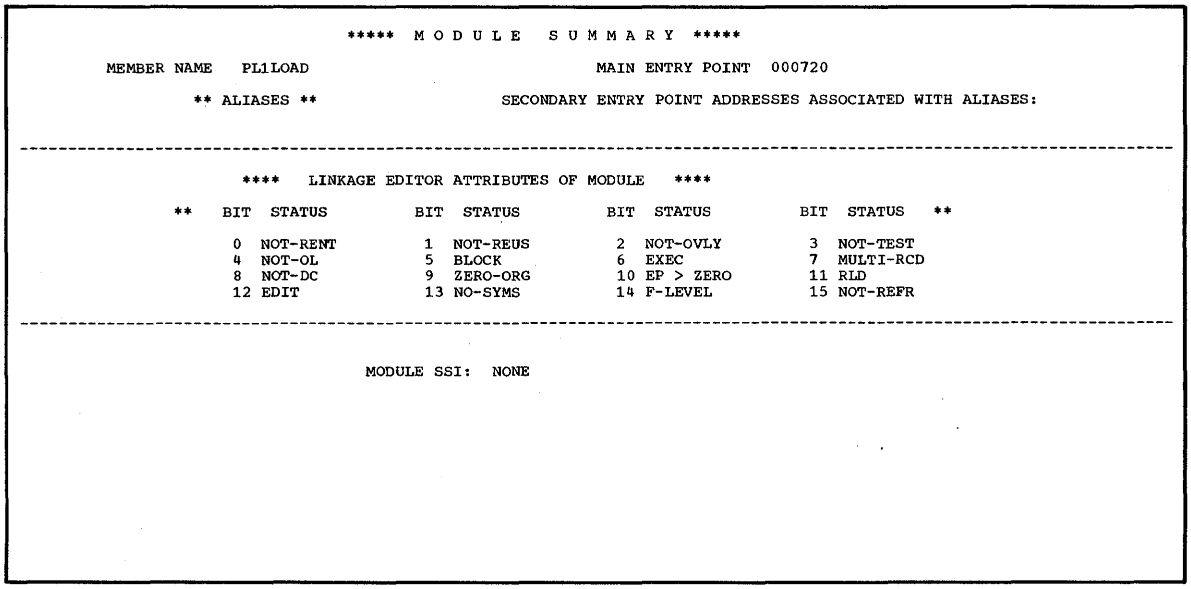
```
LISTLPA
```

Note that no operands are needed.

Output

LIST produces a separate listing for each control statement that you specify. The first page of each listing always shows the control statement as you entered it. The second page of the listing is a module summary, unless you requested LISTOBJ or LISTLPA; in that case, no module summary will be produced, and the second page of the listing will be the beginning of the formatted output.

The module summary gives the member name (with aliases), the entry point, the linkage editor attributes, system status index information (SSI) for the module being formatted, and APF code (if the module was link-edited with a VS2 linkage editor). Figure LIST-1 shows a typical module summary.



```
                              *****  M O D U L E   S U M M A R Y  *****
          MEMBER NAME   PL1LOAD                                MAIN ENTRY POINT  000720
                   ** ALIASES **                    SECONDARY ENTRY POINT ADDRESSES ASSOCIATED WITH ALIASES:

----------------------------------------------------------------------------------------------------------------------

                         ****   LINKAGE EDITOR ATTRIBUTES OF MODULE    ****
                  **   BIT  STATUS          BIT  STATUS          BIT  STATUS          BIT  STATUS   **

                        0  NOT-RENT           1  NOT-REUS          2  NOT-OVLY          3  NOT-TEST
                        4  NOT-OL             5  BLOCK             6  EXEC              7  MULTI-RCD
                        8  NOT-DC             9  ZERO-ORG          10 EP > ZERO         11 RLD
                        12 EDIT               13 NO-SYMS           14 F-LEVEL           15 NOT-REFR
----------------------------------------------------------------------------------------------------------------------

                                MODULE SSI:   NONE
```

Figure LIST-1. Sample Module Summary for LISTLOAD

The third page of the listing (or, for LISTOBJ and LISTLPA, the second page) is the beginning of the formatted output itself.

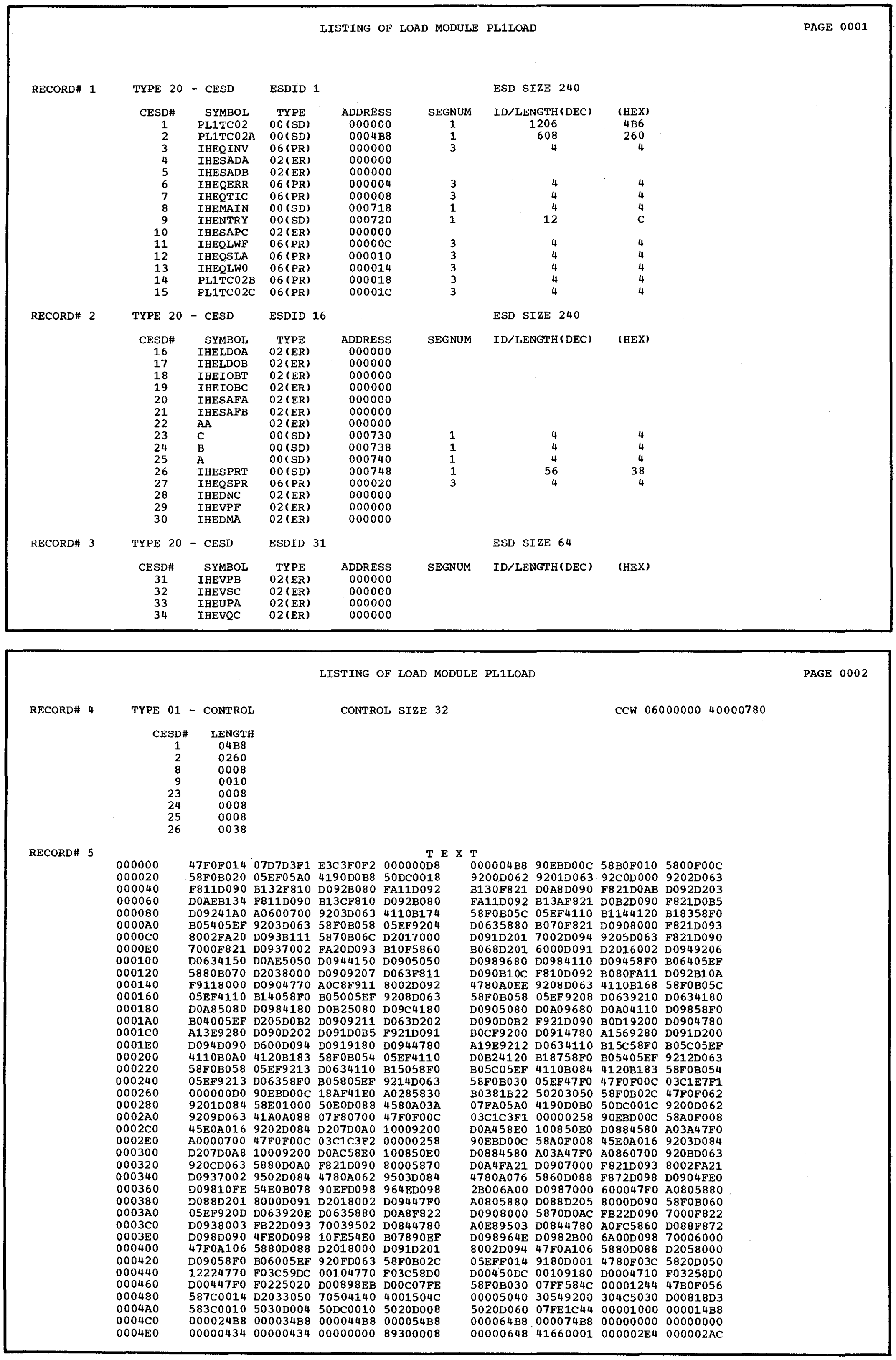
For LISTLOAD, this consists of the load module and/or the module map and cross-reference listing. Figure LIST-2 shows an example of LISTLOAD module map output. Figure LIST-3 shows an example of the cross-reference listing for the same module.

For LISTOBJ, the body of the listing consists of the object module listing, the module's external symbol dictionary, and its relocation dictionary. Figure LIST-4 shows an example of LISTOBJ output.

For LISTIDR, the third page of the listing begins a complete list of all CSECT identification records for the module. Figure LIST-5 shows an example of LISTIDR output.

For LISTLPA, the second page of the listing is a map of the link pack area, with modules ordered numerically by location. The third page is a similar map, with modules ordered alphabetically by name. Figure LIST-6 shows an example of LISTLPA output for VS1, and Figure LIST-7 shows an example of LISTLPA output for VS2. Note that in VS2 LISTLPA output consists of an alphabetical listing only.

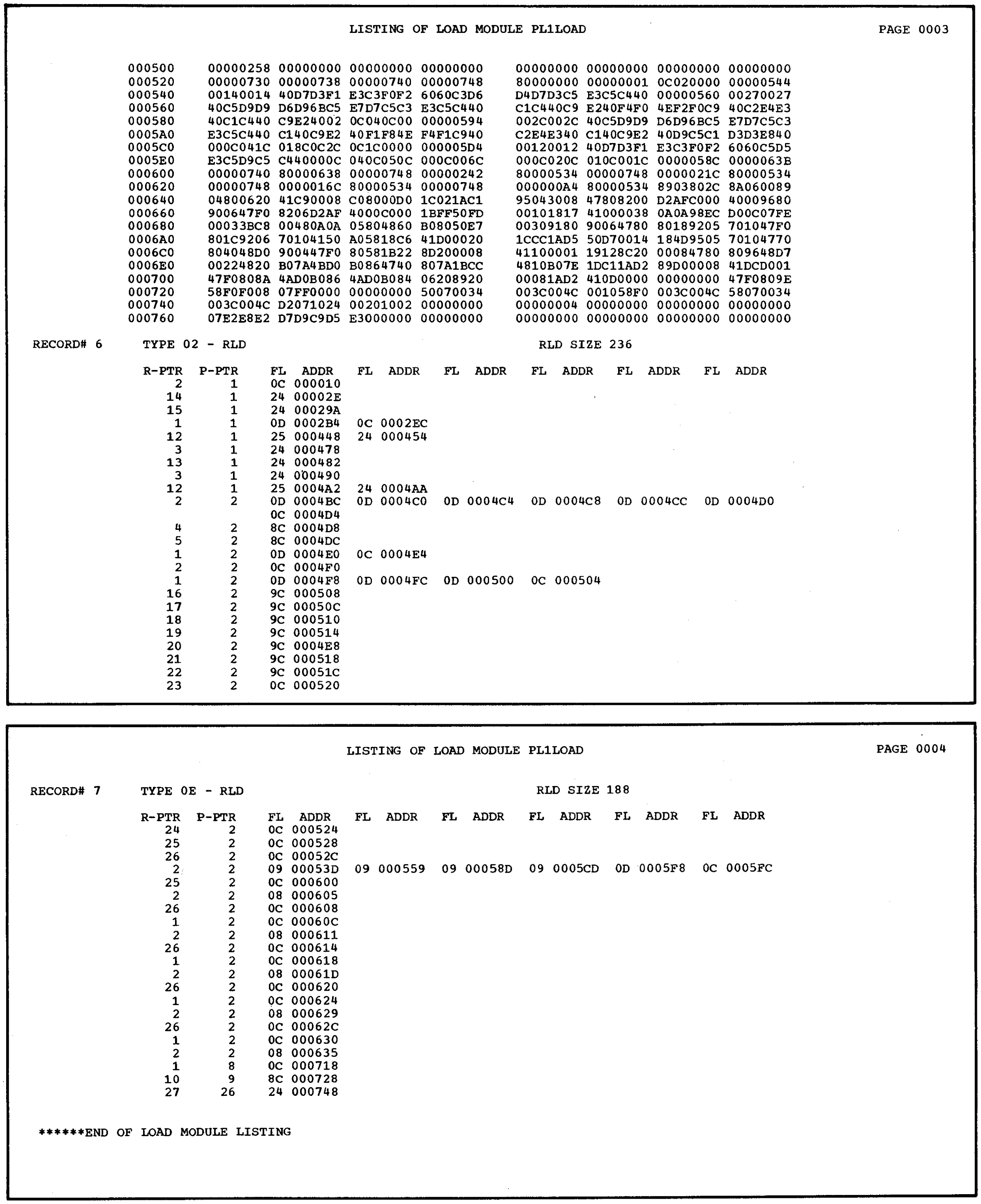
For complete descriptions of the fields in the formatted output listings, refer to the publications OS/VS1 Debugging Guide, GC24-5093, and OS/VS2 Debugging Guide, GC28-0632.



```
                                  LISTING OF LOAD MODULE PL1LOAD                               PAGE 0001


RECORD# 1     TYPE 20 - CESD     ESDID 1                         ESD SIZE 240

                CESD#    SYMBOL     TYPE      ADDRESS     SEGNUM    ID/LENGTH(DEC)     (HEX)
                  1      PL1TC02    00(SD)     000000       1          1206            4B6
                  2      PL1TC02A   00(SD)     0004B8       1           608            260
                  3      IHEQINV    06(PR)     000000       3             4              4
                  4      IHESADA    02(ER)     000000
                  5      IHESADB    02(ER)     000000
                  6      IHEQERR    06(PR)     000004       3             4              4
                  7      IHEQTIC    06(PR)     000008       3             4              4
                  8      IHEMAIN    00(SD)     000718       1             4              4
                  9      IHENTRY    00(SD)     000720       1            12              C
                 10      IHESAPC    02(ER)     000000
                 11      IHEQLWF    06(PR)     00000C       3             4              4
                 12      IHEQSLA    06(PR)     000010       3             4              4
                 13      IHEQLWO    06(PR)     000014       3             4              4
                 14      PL1TC02B   06(PR)     000018       3             4              4
                 15      PL1TC02C   06(PR)     00001C       3             4              4

RECORD# 2     TYPE 20 - CESD     ESDID 16                        ESD SIZE 240

                CESD#    SYMBOL     TYPE      ADDRESS     SEGNUM    ID/LENGTH(DEC)     (HEX)
                 16      IHELDOA    02(ER)     000000
                 17      IHELDOB    02(ER)     000000
                 18      IHEIOBT    02(ER)     000000
                 19      IHEIOBC    02(ER)     000000
                 20      IHESAFA    02(ER)     000000
                 21      IHESAFB    02(ER)     000000
                 22      AA         02(ER)     000000
                 23      C          00(SD)     000730       1             4              4
                 24      B          00(SD)     000738       1             4              4
                 25      A          00(SD)     000740       1             4              4
                 26      IHESPRT    00(SD)     000748       1            56             38
                 27      IHEQSPR    06(PR)     000020       3             4              4
                 28      IHEDNC     02(ER)     000000
                 29      IHEVPF     02(ER)     000000
                 30      IHEDMA     02(ER)     000000

RECORD# 3     TYPE 20 - CESD     ESDID 31                        ESD SIZE 64

                CESD#    SYMBOL     TYPE      ADDRESS     SEGNUM    ID/LENGTH(DEC)     (HEX)
                 31      IHEVPB     02(ER)     000000
                 32      IHEVSC     02(ER)     000000
                 33      IHEUPA     02(ER)     000000
                 34      IHEVQC     02(ER)     000000
```

```
                                  LISTING OF LOAD MODULE PL1LOAD                               PAGE 0002

RECORD# 4     TYPE 01 - CONTROL            CONTROL SIZE 32                  CCW 06000000 40000780

                  CESD#    LENGTH
                    1       04B8
                    2       0260
                    8       0008
                    9       0010
                   23       0008
                   24       0008
                   25       0008
                   26       0038

RECORD# 5                                                T E X T
             000000     47F0F014 07D7D3F1 E3C3F0F2 000000D8    000004B8 90EBD00C 58B0F010 5800F00C
             000020     58F0B020 05EF05A0 4190D0B8 50DC0018    9200D062 9201D063 92C0D000 9202D063
             000040     F811D090 B132F810 D092B080 FA11D092    B130F821 D0A8D090 F821D0AB D092D203
             000060     D0AEB134 F811D090 B13CF810 D092B080    FA11D092 B13AF821 D0B2D090 F821D0B5
             000080     D09241A0 A0600700 9203D063 4110B174    58F0B05C 05EF4110 B1144120 B18358F0
             0000A0     B05405EF 9203D063 58F0B058 05EF9204    D0635880 B070F821 D0908000 F821D093
             0000C0     8002FA20 D093B111 5870B06C D2017000    D091D201 7002D094 9205D063 F821D090
             0000E0     7000F821 D0937002 FA20D093 B10F5860    B068D201 6000D091 D2016002 D0949206
             000100     D0634150 D0AE5050 D0944150 D0905050    D0989680 D0984110 D09458F0 B06405EF
             000120     5880B070 D2038000 D0909207 D063F811    D090B10C F810D092 B080FA11 D092B10A
             000140     F9118000 D0904770 A0C8F911 8002D092    4780A0EE 9208D063 4110B168 58F0B05C
             000160     05EF4110 B14058F0 B05005EF 9208D063    58F0B058 05EF9208 D0639210 D0634180
             000180     D0A85080 D0984180 D0B25080 D09C4180    D0905080 D0A09680 D0A04110 D09858F0
             0001A0     B04005EF D205D0B2 D0909211 D063D202    D090D0B2 F921D090 B0D19200 D0904780
             0001C0     A13E9280 D090D202 D091D0B5 F921D091    B0CF9200 D0914780 A1569280 D091D200
             0001E0     D094D090 D600D094 D0919180 D0944780    A19E9212 D0634110 B15C58F0 B05C05EF
             000200     4110B0A0 4120B183 58F0B054 05EF4110    D0B24120 B18758F0 B05405EF 9212D063
             000220     58F0B058 05EF9213 D0634110 B15058F0    B05C05EF 4110B084 4120B183 58F0B054
             000240     05EF9213 D06358F0 B05805EF 9214D063    58F0B030 05EF47F0 47F0F00C 03C1E7F1
             000260     000000D0 90EBD00C 18AF41E0 A0285830    B0381B22 50203050 58F0B02C 47F0F062
             000280     9201D084 58E01000 50E0D088 4580A03A    07FA05A0 4190D0B0 50DC001C 9200D062
             0002A0     9209D063 41A0A088 07F80700 47F0F00C    03C1C3F1 00000258 90EBD00C 58A0F008
             0002C0     45E0A016 9202D084 D207D0A0 10009200    D0A458E0 100850E0 D0884580 A03A47F0
             0002E0     A0000700 47F0F00C 03C1C3F2 00000258    90EBD00C 58A0F008 45E0A016 9203D084
             000300     D207D0A8 10009200 D0AC58E0 100850E0    D0884580 A03A47F0 A0860700 920BD063
             000320     920CD063 5880D0A0 F821D090 80005870    D0A4FA21 D0907000 F821D093 8002FA21
             000340     D0937002 9502D084 4780A062 9503D084    4780A076 5860D088 F872D098 D0904FE0
             000360     D09810FE 54E0B078 90EFD098 964ED098    2B006A00 D0987000 600047F0 A0805880
             000380     D088D201 8000D091 D2018002 D09447F0    A0805880 D088D205 8000D090 58F0B060
             0003A0     05EF920D D063920E D0635880 D0A8F822    D0908000 5870D0AC FB22D090 7000F822
             0003C0     D0938003 FB22D093 70039502 D0844780    A0E89503 D0844780 A0FC5860 D088F872
             0003E0     D098D090 4FE0D098 10FE54E0 B07890EF    D098964E D0982B00 6A00D098 70006000
             000400     47F0A106 5880D088 D2018000 D091D201    8002D094 47F0A106 5880D088 D2058000
             000420     D09058F0 B06005EF 920FD063 58F0B02C    05EFF014 9180D001 4780F03C 5820D050
             000440     12224770 F03C59DC 00104770 F03C58D0    D00450DC 00109180 D0004710 F03258D0
             000460     D00447F0 F0225020 D00898EB D00C07FE    58F0B030 07FF584C 00001244 47B0F056
             000480     587C0014 D2033050 70504140 4001504C    00005040 30549200 304C5030 D00818D3
             0004A0     583C0010 5030D004 50DC0010 5020D008    5020D060 07FE1C44 00001000 000014B8
             0004C0     000024B8 000034B8 000044B8 000054B8    000064B8 000074B8 00000000 00000000
             0004E0     00000434 00000434 00000000 89300008    00000648 41660001 000002E4 000002AC
```

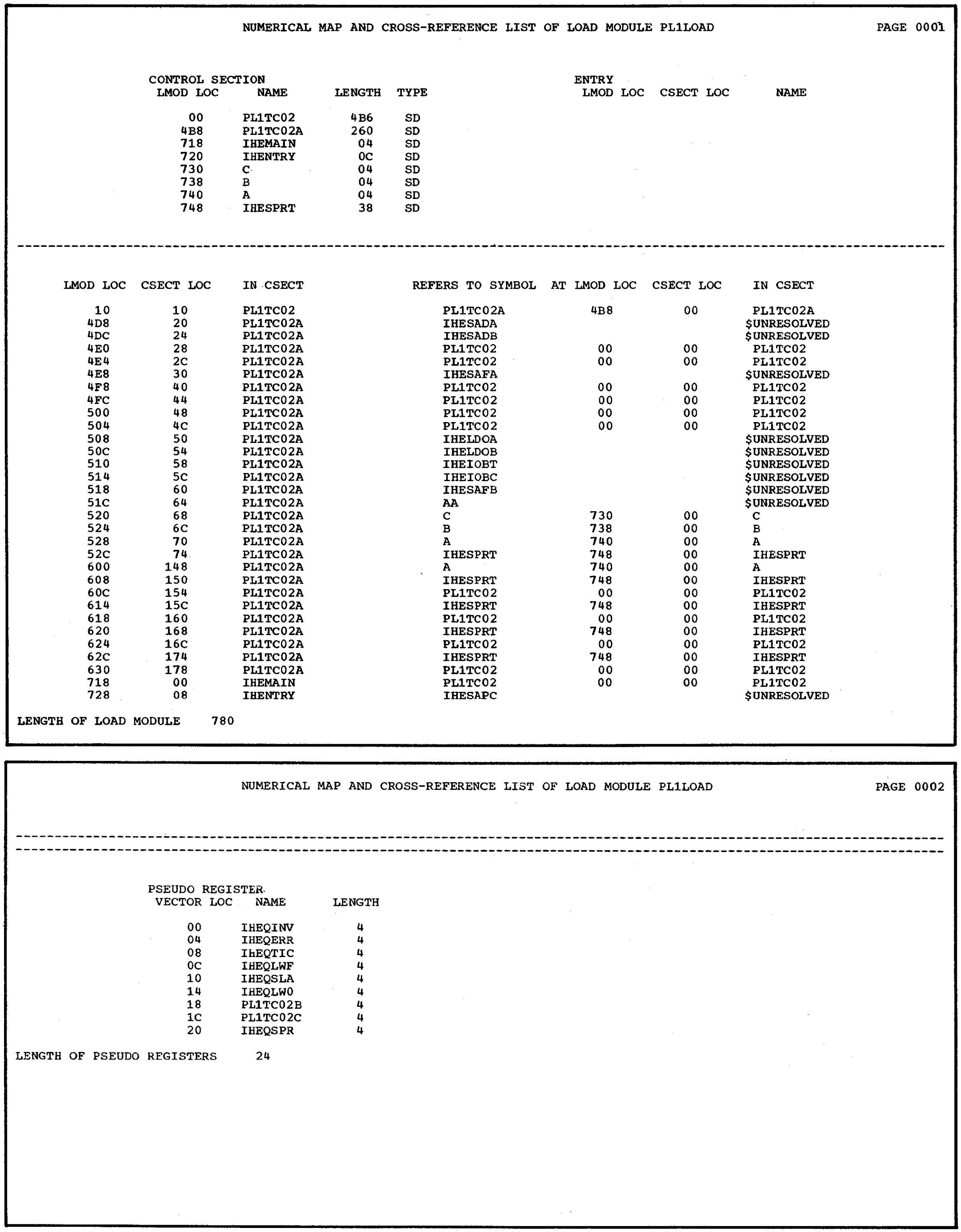
Figure LIST-2. Sample LISTLOAD Output - Load Module Map (Part 1 of 2)



```
                                   LISTING OF LOAD MODULE PL1LOAD                                    PAGE 0003

          000500      00000258 00000000 00000000 00000000    00000000 00000000 00000000 00000000
          000520      00000730 00000738 00000740 00000748    80000000 00000001 0C020000 00000544
          000540      00140014 40D7D3F1 E3C3F0F2 6060C3D6    D4D7D3C5 E3C5C440 00000560 00270027
          000560      40C5D9D9 D6D96BC5 E7D7C5C3 E3C5C440    C1C440C9 E240F4F0 4EF2F0C9 40C2E4E3
          000580      40C1C440 C9E24002 0C040C00 00000594    002C002C 40C5D9D9 D6D96BC5 E7D7C5C3
          0005A0      E3C5C440 C140C9E2 40F1F84E F4F1C940    C2E4E340 C140C9E2 40D9C5C1 D3D3E840
          0005C0      000C041C 018C0C2C 0C1C0000 000005D4    00120012 40D7D3F1 E3C3F0F2 6060C5D5
          0005E0      E3C5D9C5 C440000C 040C050C 000C006C    000C020C 010C001C 0000058C 0000063B
          000600      00000740 80000638 00000748 00000242    80000534 00000748 0000021C 80000534
          000620      00000748 0000016C 80000534 00000748    000000A4 80000534 8903802C 8A060089
          000640      04800620 41C90008 C08000D0 1C021AC1    95043008 47808200 D2AFC000 40009680
          000660      900647F0 8206D2AF 4000C000 1BFF50FD    00101817 41000038 0A0A98EC D00C07FE
          000680      00033BC8 00480A0A 05804860 B08050E7    00309180 90064780 80189205 701047F0
          0006A0      801C9206 70104150 A05818C6 41D00020    1CCC1AD5 50D70014 184D9505 70104770
          0006C0      804048D0 900447F0 80581B22 8D200008    41100001 19128C20 00084780 809648D7
          0006E0      00224820 B07A4BD0 B0864740 807A1BCC    4810B07E 1DC11AD2 89D00008 41DCD001
          000700      47F0808A 4AD0B086 4AD0B084 06208920    00081AD2 410D0000 00000000 47F0809E
          000720      58F0F008 07FF0000 00000000 50070034    003C004C 001058F0 003C004C 58070034
          000740      003C004C D2071024 00201002 00000000    00000004 00000000 00000000 00000000
          000760      07E2E8E2 D7D9C9D5 E3000000 00000000    00000000 00000000 00000000 00000000

RECORD# 6    TYPE 02 - RLD                                         RLD SIZE 236

             R-PTR  P-PTR    FL  ADDR   FL  ADDR   FL  ADDR   FL  ADDR   FL  ADDR   FL  ADDR
                2      1     0C 000010
               14      1     24 00002E
               15      1     24 00029A
                1      1     0D 0002B4  0C 0002EC
               12      1     25 000448  24 000454
                3      1     24 000478
               13      1     24 000482
                3      1     24 000490
               12      1     25 0004A2  24 0004AA
                2      2     0D 0004BC  0D 0004C0  0D 0004C4  0D 0004C8  0D 0004CC  0D 0004D0
                             0C 0004D4
                4      2     8C 0004D8
                5      2     8C 0004DC
                1      2     0D 0004E0  0C 0004E4
                2      2     0C 0004F0
                1      2     0D 0004F8  0D 0004FC  0D 000500  0C 000504
               16      2     9C 000508
               17      2     9C 00050C
               18      2     9C 000510
               19      2     9C 000514
               20      2     9C 0004E8
               21      2     9C 000518
               22      2     9C 00051C
               23      2     0C 000520
```

```
                                   LISTING OF LOAD MODULE PL1LOAD                                    PAGE 0004

RECORD# 7    TYPE 0E - RLD                                         RLD SIZE 188

             R-PTR  P-PTR    FL  ADDR   FL  ADDR   FL  ADDR   FL  ADDR   FL  ADDR   FL  ADDR
               24      2     0C 000524
               25      2     0C 000528
               26      2     0C 00052C
                2      2     09 00053D  09 000559  09 00058D  09 0005CD  0D 0005F8  0C 0005FC
               25      2     0C 000600
                2      2     08 000605
               26      2     0C 000608
                1      2     0C 00060C
                2      2     08 000611
               26      2     0C 000614
                1      2     0C 000618
                2      2     08 00061D
               26      2     0C 000620
                1      2     0C 000624
                2      2     08 000629
               26      2     0C 00062C
                1      2     0C 000630
                2      2     08 000635
                1      8     0C 000718
               10      9     8C 000728
               27     26     24 000748

 ******END OF LOAD MODULE LISTING
```

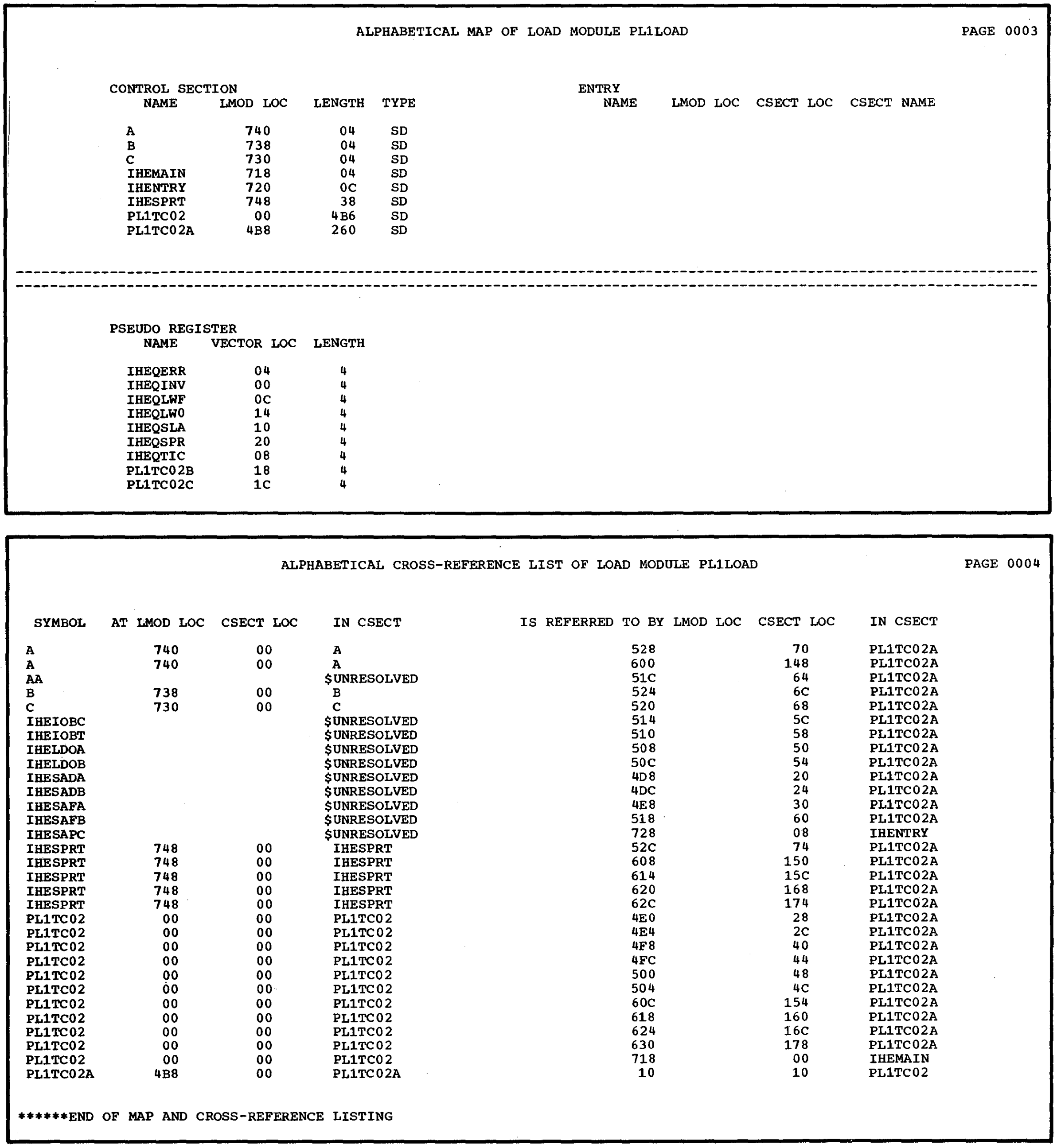
Figure LIST-2. Sample LISTLOAD Output - Load Module Map (Part 2 of 2)



```
                         NUMERICAL MAP AND CROSS-REFERENCE LIST OF LOAD MODULE PL1LOAD                PAGE 0001


                 CONTROL SECTION                                            ENTRY
                  LMOD LOC     NAME       LENGTH  TYPE                       LMOD LOC   CSECT LOC      NAME

                      00      PL1TC02       4B6    SD
                     4B8      PL1TC02A      260    SD
                     718      IHEMAIN        04    SD
                     720      IHENTRY        0C    SD
                     730      C              04    SD
                     738      B              04    SD
                     740      A              04    SD
                     748      IHESPRT        38    SD

-----------------------------------------------------------------------------------------------------------------


       LMOD LOC  CSECT LOC    IN CSECT             REFERS TO SYMBOL  AT LMOD LOC   CSECT LOC     IN CSECT

             10        10     PL1TC02                   PL1TC02A          4B8          00         PL1TC02A
            4D8        20     PL1TC02A                  IHESADA                                  $UNRESOLVED
            4DC        24     PL1TC02A                  IHESADB                                  $UNRESOLVED
            4E0        28     PL1TC02A                  PL1TC02            00          00         PL1TC02
            4E4        2C     PL1TC02A                  PL1TC02            00          00         PL1TC02
            4E8        30     PL1TC02A                  IHESAFA                                  $UNRESOLVED
            4F8        40     PL1TC02A                  PL1TC02            00          00         PL1TC02
            4FC        44     PL1TC02A                  PL1TC02            00          00         PL1TC02
            500        48     PL1TC02A                  PL1TC02            00          00         PL1TC02
            504        4C     PL1TC02A                  PL1TC02            00          00         PL1TC02
            508        50     PL1TC02A                  IHELDOA                                  $UNRESOLVED
            50C        54     PL1TC02A                  IHELDOB                                  $UNRESOLVED
            510        58     PL1TC02A                  IHEIOBT                                  $UNRESOLVED
            514        5C     PL1TC02A                  IHEIOBC                                  $UNRESOLVED
            518        60     PL1TC02A                  IHESAFB                                  $UNRESOLVED
            51C        64     PL1TC02A                  AA                                       $UNRESOLVED
            520        68     PL1TC02A                  C                 730          00         C
            524        6C     PL1TC02A                  B                 738          00         B
            528        70     PL1TC02A                  A                 740          00         A
            52C        74     PL1TC02A                  IHESPRT           748          00         IHESPRT
            600       148     PL1TC02A                  A                 740          00         A
            608       150     PL1TC02A                  IHESPRT           748          00         IHESPRT
            60C       154     PL1TC02A                  PL1TC02            00          00         PL1TC02
            614       15C     PL1TC02A                  IHESPRT           748          00         IHESPRT
            618       160     PL1TC02A                  PL1TC02            00          00         PL1TC02
            620       168     PL1TC02A                  IHESPRT           748          00         IHESPRT
            624       16C     PL1TC02A                  PL1TC02            00          00         PL1TC02
            62C       174     PL1TC02A                  IHESPRT           748          00         IHESPRT
            630       178     PL1TC02A                  PL1TC02            00          00         PL1TC02
            718        00     IHEMAIN                   PL1TC02            00          00         PL1TC02
            728        08     IHENTRY                   IHESAPC                                  $UNRESOLVED

LENGTH OF LOAD MODULE    780
```

```
                         NUMERICAL MAP AND CROSS-REFERENCE LIST OF LOAD MODULE PL1LOAD                PAGE 0002


-----------------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------------


                 PSEUDO REGISTER
                  VECTOR LOC    NAME       LENGTH

                      00      IHEQINV        4
                      04      IHEQERR        4
                      08      IHEQTIC        4
                      0C      IHEQLWF        4
                      10      IHEQSLA        4
                      14      IHEQLWO        4
                      18      PL1TC02B       4
                      1C      PL1TC02C       4
                      20      IHEQSPR        4

LENGTH OF PSEUDO REGISTERS      24
```

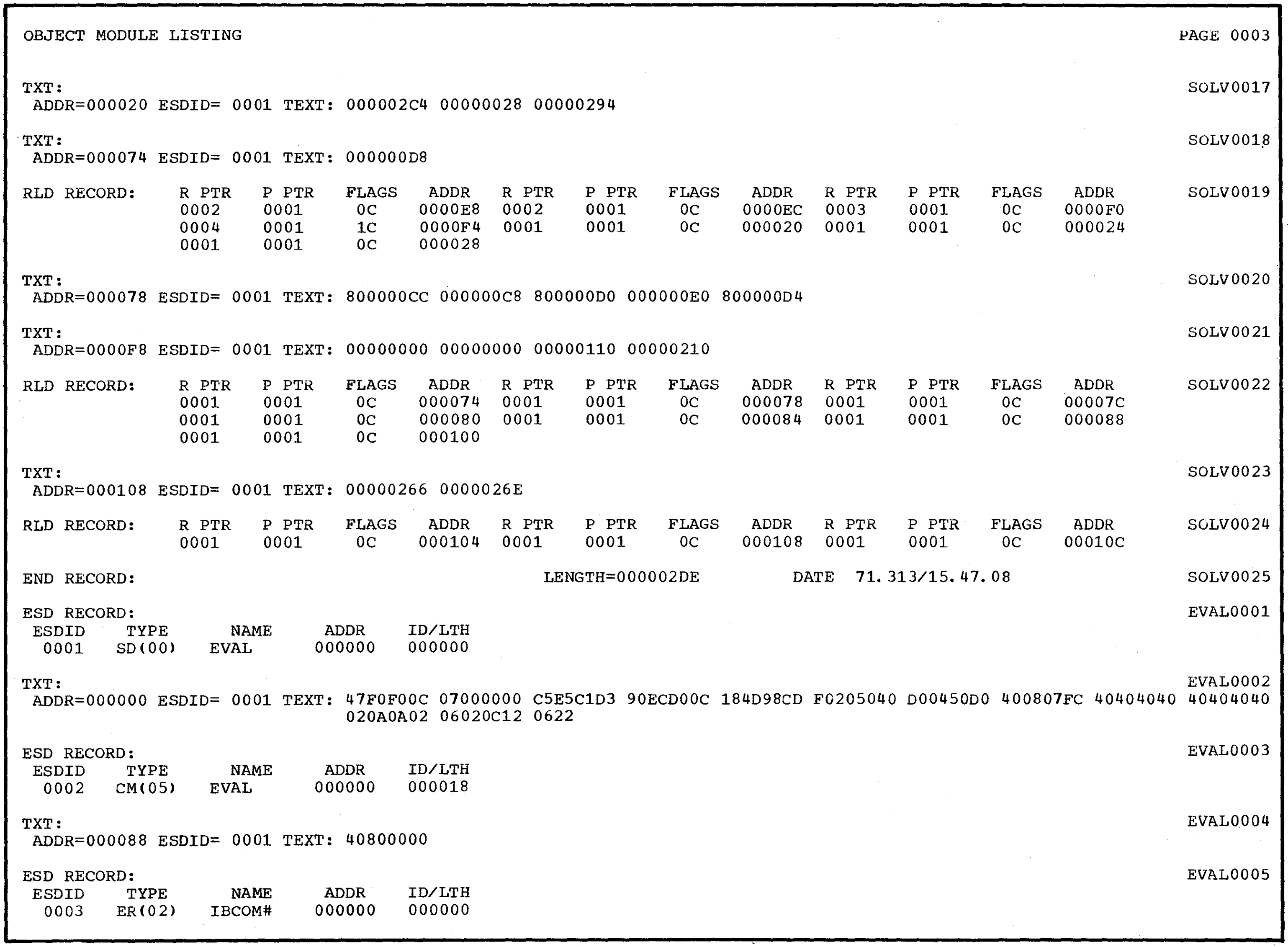
Figure LIST-3. Sample LISTLOAD Output - Cross Reference Listing (Part 1 of 2)



```
                              ALPHABETICAL MAP OF LOAD MODULE PL1LOAD                              PAGE 0003


          CONTROL SECTION                                          ENTRY
             NAME     LMOD LOC    LENGTH  TYPE                        NAME    LMOD LOC   CSECT LOC   CSECT NAME

            A            740        04     SD
            B            738        04     SD
            C            730        04     SD
            IHEMAIN      718        04     SD
            IHENTRY      720        0C     SD
            IHESPRT      748        38     SD
            PL1TC02       00       4B6     SD
            PL1TC02A     4B8       260     SD

-------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------


          PSEUDO REGISTER
             NAME    VECTOR LOC   LENGTH

            IHEQERR      04         4
            IHEQINV      00         4
            IHEQLWF      0C         4
            IHEQLW0      14         4
            IHEQSLA      10         4
            IHEQSPR      20         4
            IHEQTIC      08         4
            PL1TC02B     18         4
            PL1TC02C     1C         4
```

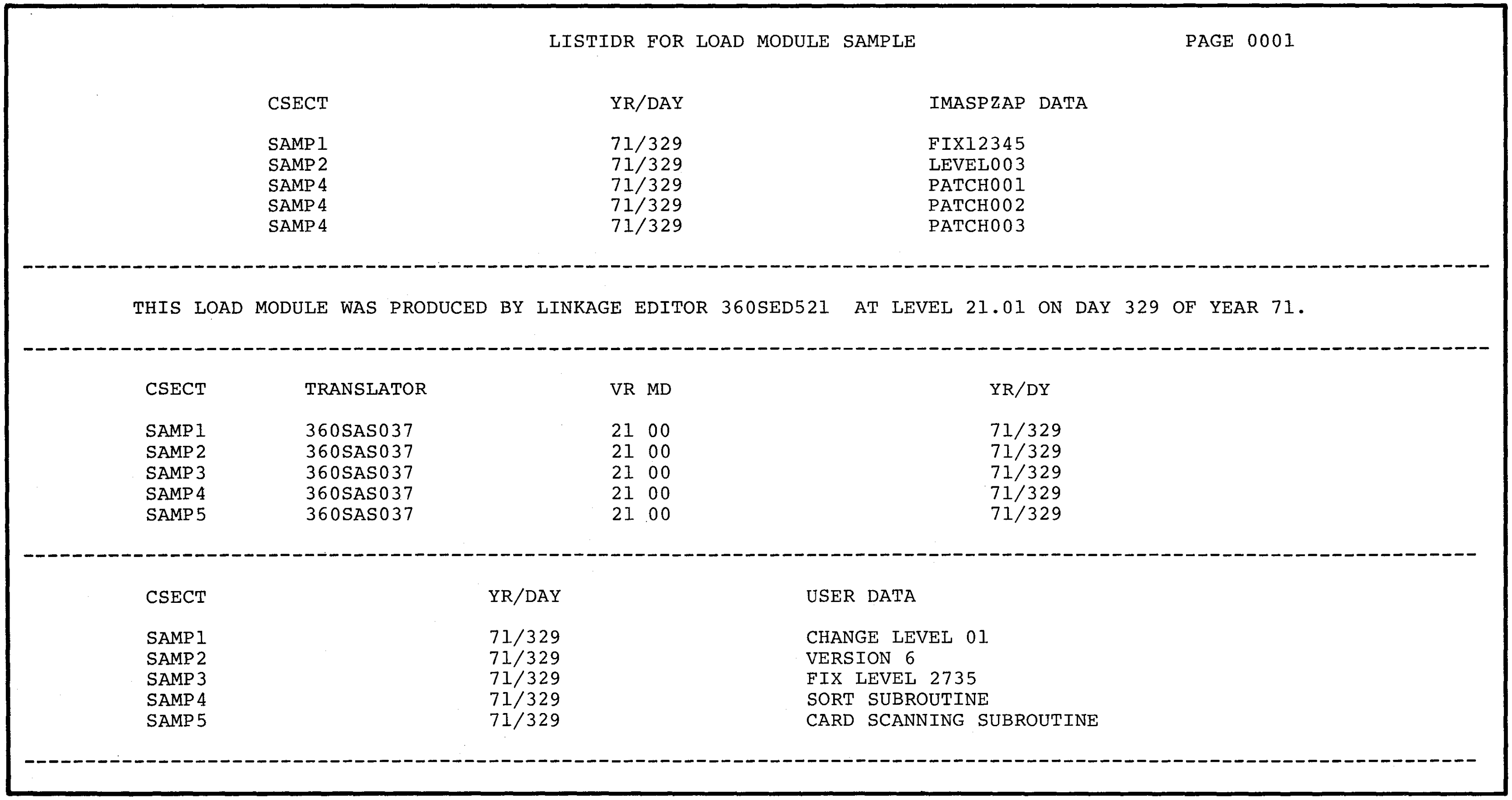
```
                              ALPHABETICAL CROSS-REFERENCE LIST OF LOAD MODULE PL1LOAD                           PAGE 0004


 SYMBOL    AT LMOD LOC   CSECT LOC     IN CSECT                IS REFERRED TO BY LMOD LOC   CSECT LOC      IN CSECT

A              740          00        A                                        528              70         PL1TC02A
A              740          00        A                                        600             148         PL1TC02A
AA                                   $UNRESOLVED                               51C              64         PL1TC02A
B              738          00        B                                        524              6C         PL1TC02A
C              730          00        C                                        520              68         PL1TC02A
IHEIOBC                              $UNRESOLVED                               514              5C         PL1TC02A
IHEIOBT                              $UNRESOLVED                               510              58         PL1TC02A
IHELDOA                              $UNRESOLVED                               508              50         PL1TC02A
IHELDOB                              $UNRESOLVED                               50C              54         PL1TC02A
IHESADA                              $UNRESOLVED                               4D8              20         PL1TC02A
IHESADB                              $UNRESOLVED                               4DC              24         PL1TC02A
IHESAFA                              $UNRESOLVED                               4E8              30         PL1TC02A
IHESAFB                              $UNRESOLVED                               518              60         PL1TC02A
IHESAPC                              $UNRESOLVED                               728              08         IHENTRY
IHESPRT        748          00        IHESPRT                                  52C              74         PL1TC02A
IHESPRT        748          00        IHESPRT                                  608             150         PL1TC02A
IHESPRT        748          00        IHESPRT                                  614             15C         PL1TC02A
IHESPRT        748          00        IHESPRT                                  620             168         PL1TC02A
IHESPRT        748          00        IHESPRT                                  62C             174         PL1TC02A
PL1TC02         00          00        PL1TC02                                  4E0              28         PL1TC02A
PL1TC02         00          00        PL1TC02                                  4E4              2C         PL1TC02A
PL1TC02         00          00        PL1TC02                                  4F8              40         PL1TC02A
PL1TC02         00          00        PL1TC02                                  4FC              44         PL1TC02A
PL1TC02         00          00        PL1TC02                                  500              48         PL1TC02A
PL1TC02         00          00        PL1TC02                                  504              4C         PL1TC02A
PL1TC02         00          00        PL1TC02                                  60C             154         PL1TC02A
PL1TC02         00          00        PL1TC02                                  618             160         PL1TC02A
PL1TC02         00          00        PL1TC02                                  624             16C         PL1TC02A
PL1TC02         00          00        PL1TC02                                  630             178         PL1TC02A
PL1TC02         00          00        PL1TC02                                  718              00         IHEMAIN
PL1TC02A       4B8          00        PL1TC02A                                  10              10         PL1TC02

******END OF MAP AND CROSS-REFERENCE LISTING
```

Figure LIST-3. Sample LISTLOAD Output - Cross Reference Listing (Part 2 of 2)



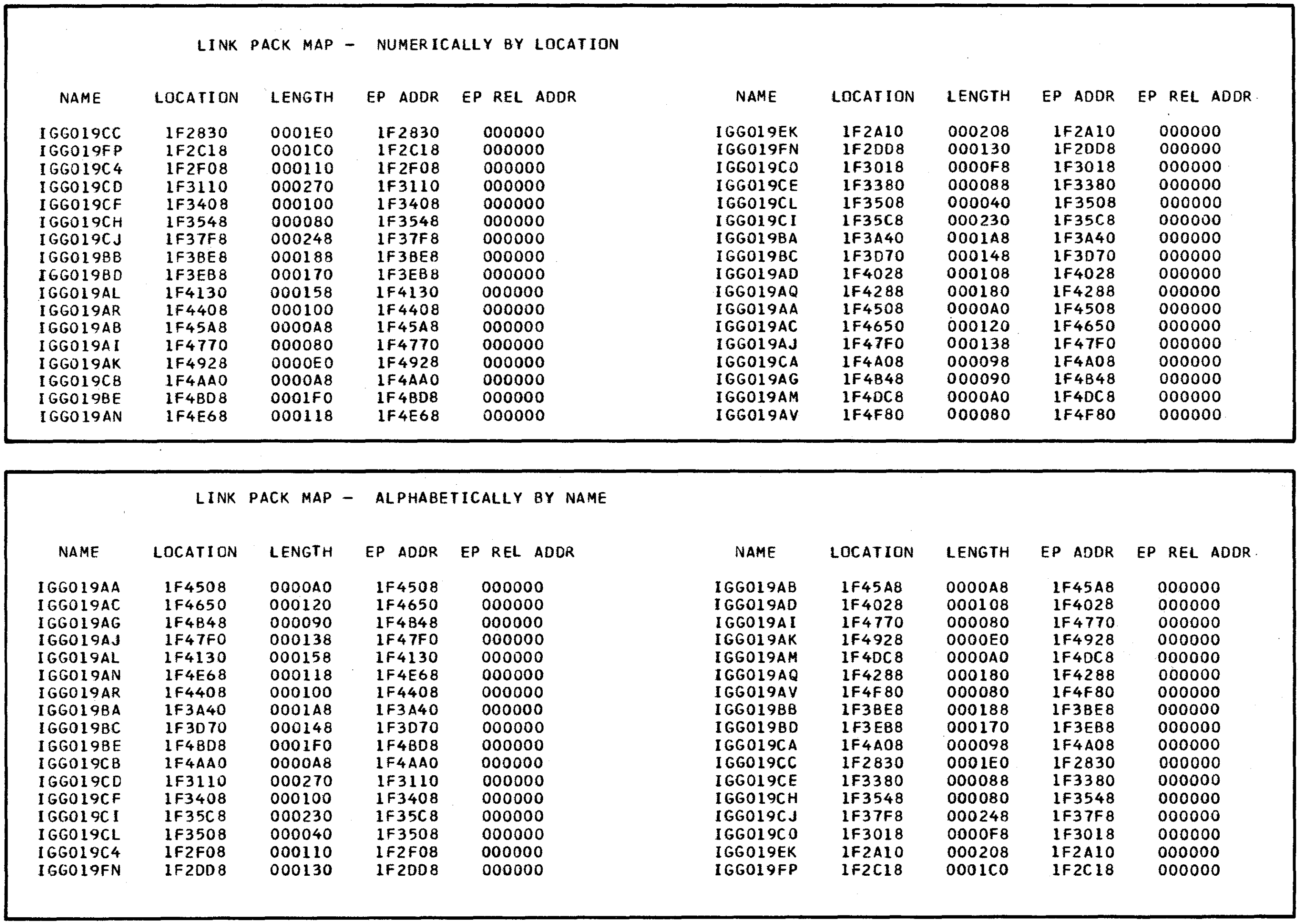
```
OBJECT MODULE LISTING                                                                                                            PAGE 0003

TXT:                                                                                                                              SOLV0017
 ADDR=000020 ESDID= 0001 TEXT: 000002C4 00000028 00000294

TXT:                                                                                                                              SOLV0018
 ADDR=000074 ESDID= 0001 TEXT: 000000D8

RLD RECORD:     R PTR   P PTR   FLAGS   ADDR    R PTR   P PTR   FLAGS   ADDR    R PTR   P PTR   FLAGS   ADDR                      SOLV0019
                0002    0001     0C     0000E8  0002    0001     0C     0000EC  0003    0001     0C     0000F0
                0004    0001     1C     0000F4  0001    0001     0C     000020  0001    0001     0C     000024
                0001    0001     0C     000028

TXT:                                                                                                                              SOLV0020
 ADDR=000078 ESDID= 0001 TEXT: 800000CC 000000C8 800000D0 000000E0 800000D4

TXT:                                                                                                                              SOLV0021
 ADDR=0000F8 ESDID= 0001 TEXT: 00000000 00000000 00000110 00000210

RLD RECORD:     R PTR   P PTR   FLAGS   ADDR    R PTR   P PTR   FLAGS   ADDR    R PTR   P PTR   FLAGS   ADDR                      SOLV0022
                0001    0001     0C     000074  0001    0001     0C     000078  0001    0001     0C     00007C
                0001    0001     0C     000080  0001    0001     0C     000084  0001    0001     0C     000088
                0001    0001     0C     000100

TXT:                                                                                                                              SOLV0023
 ADDR=000108 ESDID= 0001 TEXT: 00000266 0000026E

RLD RECORD:     R PTR   P PTR   FLAGS   ADDR    R PTR   P PTR   FLAGS   ADDR    R PTR   P PTR   FLAGS   ADDR                      SOLV0024
                0001    0001     0C     000104  0001    0001     0C     000108  0001    0001     0C     00010C

END RECORD:                                     LENGTH=000002DE         DATE  71.313/15.47.08                                     SOLV0025

ESD RECORD:                                                                                                                       EVAL0001
 ESDID    TYPE      NAME      ADDR     ID/LTH
  0001    SD(00)   EVAL      000000    000000

TXT:                                                                                                                              EVAL0002
 ADDR=000000 ESDID= 0001 TEXT: 47F0F00C 07000000 C5E5C1D3 90ECD00C 184D98CD F0205040 D00450D0 400807FC 40404040 40404040
                               020A0A02 06020C12 0622

ESD RECORD:                                                                                                                       EVAL0003
 ESDID    TYPE      NAME      ADDR     ID/LTH
  0002    CM(05)   EVAL      000000    000018

TXT:                                                                                                                              EVAL0004
 ADDR=000088 ESDID= 0001 TEXT: 40800000

ESD RECORD:                                                                                                                       EVAL0005
 ESDID    TYPE      NAME      ADDR     ID/LTH
  0003    ER(02)   IBCOM#    000000    000000
```

Figure LIST-4. Sample LISTOBJ Output



```
                                   LISTIDR FOR LOAD MODULE SAMPLE                    PAGE 0001

                  CSECT                      YR/DAY                    IMASPZAP DATA

                  SAMP1                      71/329                    FIX12345
                  SAMP2                      71/329                    LEVEL003
                  SAMP4                      71/329                    PATCH001
                  SAMP4                      71/329                    PATCH002
                  SAMP4                      71/329                    PATCH003

-------------------------------------------------------------------------------------------------------

        THIS LOAD MODULE WAS PRODUCED BY LINKAGE EDITOR 360SED521  AT LEVEL 21.01 ON DAY 329 OF YEAR 71.

-------------------------------------------------------------------------------------------------------

         CSECT        TRANSLATOR                 VR MD                          YR/DY

         SAMP1        360SAS037                  21 00                          71/329
         SAMP2        360SAS037                  21 00                          71/329
         SAMP3        360SAS037                  21 00                          71/329
         SAMP4        360SAS037                  21 00                          71/329
         SAMP5        360SAS037                  21 00                          71/329

-------------------------------------------------------------------------------------------------------

         CSECT                            YR/DAY                    USER DATA

         SAMP1                            71/329                    CHANGE LEVEL 01
         SAMP2                            71/329                    VERSION 6
         SAMP3                            71/329                    FIX LEVEL 2735
         SAMP4                            71/329                    SORT SUBROUTINE
         SAMP5                            71/329                    CARD SCANNING SUBROUTINE

-------------------------------------------------------------------------------------------------------
```

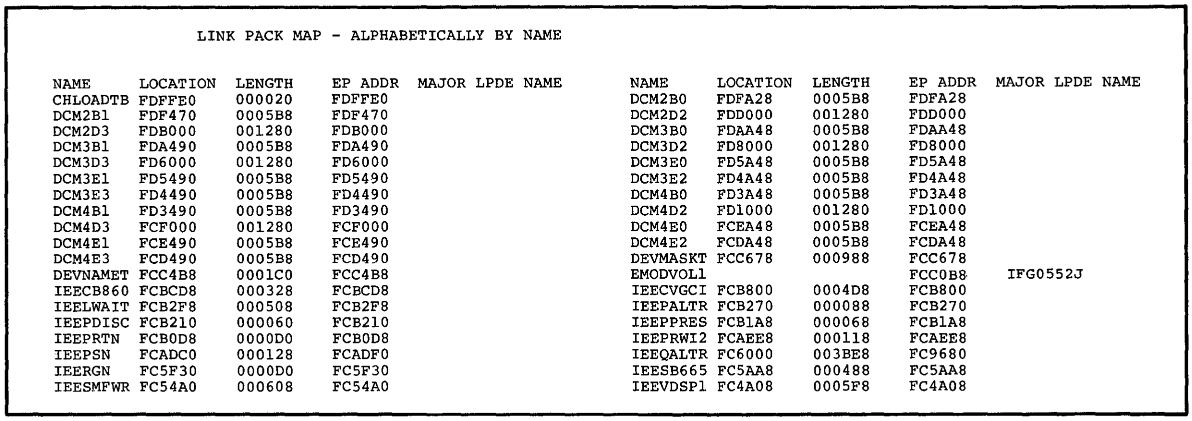
Figure LIST-5. Sample LISTIDR Output



```
              LINK PACK MAP -  NUMERICALLY BY LOCATION

  NAME      LOCATION   LENGTH   EP ADDR  EP REL ADDR           NAME      LOCATION   LENGTH   EP ADDR  EP REL ADDR

IGG019CC     1F2830    0001E0    1F2830    000000            IGG019EK     1F2A10    000208    1F2A10    000000
IGG019FP     1F2C18    0001C0    1F2C18    000000            IGG019FN     1F2DD8    000130    1F2DD8    000000
IGG019C4     1F2F08    000110    1F2F08    000000            IGG019CO     1F3018    0000F8    1F3018    000000
IGG019CD     1F3110    000270    1F3110    000000            IGG019CE     1F3380    000088    1F3380    000000
IGG019CF     1F3408    000100    1F3408    000000            IGG019CL     1F3508    000040    1F3508    000000
IGG019CH     1F3548    000080    1F3548    000000            IGG019CI     1F35C8    000230    1F35C8    000000
IGG019CJ     1F37F8    000248    1F37F8    000000            IGG019BA     1F3A40    0001A8    1F3A40    000000
IGG019BB     1F3BE8    000188    1F3BE8    000000            IGG019BC     1F3D70    000148    1F3D70    000000
IGG019BD     1F3EB8    000170    1F3EB8    000000            IGG019AD     1F4028    000108    1F4028    000000
IGG019AL     1F4130    000158    1F4130    000000            IGG019AQ     1F4288    000180    1F4288    000000
IGG019AR     1F4408    000100    1F4408    000000            IGG019AA     1F4508    0000A0    1F4508    000000
IGG019AB     1F45A8    0000A8    1F45A8    000000            IGG019AC     1F4650    000120    1F4650    000000
IGG019AI     1F4770    000080    1F4770    000000            IGG019AJ     1F47F0    000138    1F47F0    000000
IGG019AK     1F4928    0000E0    1F4928    000000            IGG019CA     1F4A08    000098    1F4A08    000000
IGG019CB     1F4AA0    0000A8    1F4AA0    000000            IGG019AG     1F4B48    000090    1F4B48    000000
IGG019BE     1F4BD8    0001F0    1F4BD8    000000            IGG019AM     1F4DC8    0000A0    1F4DC8    000000
IGG019AN     1F4E68    000118    1F4E68    000000            IGG019AV     1F4F80    000080    1F4F80    000000
```

```
              LINK PACK MAP -  ALPHABETICALLY BY NAME

  NAME      LOCATION   LENGTH   EP ADDR  EP REL ADDR           NAME      LOCATION   LENGTH   EP ADDR  EP REL ADDR

IGG019AA     1F4508    0000A0    1F4508    000000            IGG019AB     1F45A8    0000A8    1F45A8    000000
IGG019AC     1F4650    000120    1F4650    000000            IGG019AD     1F4028    000108    1F4028    000000
IGG019AG     1F4B48    000090    1F4B48    000000            IGG019AI     1F4770    000080    1F4770    000000
IGG019AJ     1F47F0    000138    1F47F0    000000            IGG019AK     1F4928    0000E0    1F4928    000000
IGG019AL     1F4130    000158    1F4130    000000            IGG019AM     1F4DC8    0000A0    1F4DC8    000000
IGG019AN     1F4E68    000118    1F4E68    000000            IGG019AQ     1F4288    000180    1F4288    000000
IGG019AR     1F4408    000100    1F4408    000000            IGG019AV     1F4F80    000080    1F4F80    000000
IGG019BA     1F3A40    0001A8    1F3A40    000000            IGG019BB     1F3BE8    000188    1F3BE8    000000
IGG019BC     1F3D70    000148    1F3D70    000000            IGG019BD     1F3EB8    000170    1F3EB8    000000
IGG019BE     1F4BD8    0001F0    1F4BD8    000000            IGG019CA     1F4A08    000098    1F4A08    000000
IGG019CB     1F4AA0    0000A8    1F4AA0    000000            IGG019CC     1F2830    0001E0    1F2830    000000
IGG019CD     1F3110    000270    1F3110    000000            IGG019CE     1F3380    000088    1F3380    000000
IGG019CF     1F3408    000100    1F3408    000000            IGG019CH     1F3548    000080    1F3548    000000
IGG019CI     1F35C8    000230    1F35C8    000000            IGG019CJ     1F37F8    000248    1F37F8    000000
IGG019CL     1F3508    000040    1F3508    000000            IGG019CO     1F3018    0000F8    1F3018    000000
IGG019C4     1F2F08    000110    1F2F08    000000            IGG019EK     1F2A10    000208    1F2A10    000000
IGG019FN     1F2DD8    000130    1F2DD8    000000            IGG019FP     1F2C18    0001C0    1F2C18    000000
```

Figure LIST-6. Sample LISTLPA Output for VS1.



```
               LINK PACK MAP - ALPHABETICALLY BY NAME

NAME     LOCATION  LENGTH    EP ADDR  MAJOR LPDE NAME       NAME     LOCATION  LENGTH    EP ADDR  MAJOR LPDE NAME
CHLOADTB FDFFE0    000020    FDFFE0                         DCM2B0   FDFA28    0005B8    FDFA28
DCM2B1   FDF470    0005B8    FDF470                         DCM2D2   FDD000    001280    FDD000
DCM2D3   FDB000    001280    FDB000                         DCM3B0   FDAA48    0005B8    FDAA48
DCM3B1   FDA490    0005B8    FDA490                         DCM3D2   FD8000    001280    FD8000
DCM3D3   FD6000    001280    FD6000                         DCM3E0   FD5A48    0005B8    FD5A48
DCM3E1   FD5490    0005B8    FD5490                         DCM3E2   FD4A48    0005B8    FD4A48
DCM3E3   FD4490    0005B8    FD4490                         DCM4B0   FD3A48    0005B8    FD3A48
DCM4B1   FD3490    0005B8    FD3490                         DCM4D2   FD1000    001280    FD1000
DCM4D3   FCF000    001280    FCF000                         DCM4E0   FCEA48    0005B8    FCEA48
DCM4E1   FCE490    0005B8    FCE490                         DCM4E2   FCDA48    0005B8    FCDA48
DCM4E3   FCD490    0005B8    FCD490                         DEVMASKT FCC678    000988    FCC678
DEVNAMET FCC4B8    0001C0    FCC4B8                         EMODVOL1                     FCC0B8    IFG0552J
IEECB860 FCBCD8    000328    FCBCD8                         IEECVGCI FCB800    0004D8    FCB800
IEELWAIT FCB2F8    000508    FCB2F8                         IEEPALTR FCB270    000088    FCB270
IEEPDISC FCB210    000060    FCB210                         IEEPPRES FCB1A8    000068    FCB1A8
IEEPRTN  FCB0D8    0000D0    FCB0D8                         IEEPRWI2 FCAEE8    000118    FCAEE8
IEEPSN   FCADC0    000128    FCADF0                         IEEQALTR FC6000    003BE8    FC9680
IEERGN   FC5F30    0000D0    FC5F30                         IEESB665 FC5AA8    000488    FC5AA8
IEESMFWR FC54A0    000608    FC54A0                         IEEVDSP1 FC4A08    0005F8    FC4A08
```

Figure LIST-7. Sample LISTLPA Output for VS2.

LIST

Examples

Example 1: Listing Several Object Modules

In this example, LIST is used to list all object modules contained in the data set named OBJMODS, and three specific object modules from another data set called OBJMOD.

```
//OBJLIST      JOB       MSGLEVEL=(1,1)
//LISTSTEP     EXEC      PGM=xMBLIST
//SYSPRINT     DD        SYSOUT=A
//OBJLIB       DD        DSN=OBJMODS,DISP=OLD
//OBJSDS       DD        DSN=OBJMOD=DISPOLD
//SYSIN        DD        *
    LISTOBJ         DDN=OBJSDS,
         TITLE=('OBJECT MODULE LISTING OF OBJSDS',20)
    LISTOBJ         DDN=OBJLIB,MEMBER=(OBJ1,OBJ2,OBJ3),
         TITLE=('OBJECT MODULE LISTING OF OBJ1 OBJ2 OBJ3',20)
/*
```

JOB Statement

initiates the job.

EXEC Statement

calls for the execution of HMBLIST (in VS1) or AMBLIST (in VS2).

SYSPRINT DD statement

defines the message data set. This statement must be included; if it is omitted, LIST will produce no output.

OBJLIB and OBJSDS DD Statements

define input data sets that contain object modules.

SYSIN DD Statement

defines the data set in the input stream containing LIST control statements.

LISTOBJ Control Statement #1

instructs LIST to format the data set defined by the OBJSDS DD statement, treating them as a single member. It also specifies a title for each page of output, to be indented 20 characters from the left margin.

LISTOBJ Control Statement #2

instructs LIST to format three members of the partitioned data set defined by the OBJLIB DD statement. It also specifies a title for each page of output, to be indented 20 characters from the left margin.

Example 2: Listing Several Load Modules

In this example, LIST is used to produced formatted listings of several load modules.

```
//LOADLIST    JOB       MSGLEVEL=(1,1)
//LISTSTEP    EXEC      PGM=xMBLIST
//SYSPRINT    DD        SYSOUT=A
//SYSLIB      DD        DSNAME=SYS1.LINKLIB,DISP=OLD
//LOADLIB     DD        DSNAME=LOADMOD,DISP=OLD
//SYSIN       DD        *
    LISTLOAD       OUTPUT=MODLIST,DDN=LOADLIB,
         MEMBER=TESTMOD,
         TITLE=('LOAD MODULE LISTING OF TESTMOD',20)
    LISTLOAD       OUTPUT=XREF,DDN=LOADLIB,
         MEMBER=(MOD1,MOD2,MOD3),
         TITLE=('XREF LISTINGS OF MOD1 MOD2  AND MOD3',20)
    LISTLOAD  TITLE=('XREF & LD MOD LSTNG - ALL MOD IN LINKLIB',20)
/*
```

In this example:

JOB Statement

initiates the job.

EXEC Statement

calls for the execution of HMBLIST (in VS1) or AMBLIST (in VS2).

SYSPRINT DD Statement

defines the message data set.

SYSLIB DD Statement

defines an input data set, SYS1.LINKLIB, that contains load modules to be formatted.

LOADLIB DD Statement

defines a second input data set.

SYSIN DD Statement

defines the data set (in the input stream) containing the LIST control statements.

LISTLOAD Control Statement #1

instructs LIST to format the control and text records including the external symbol dictionary and relocation dictionary records of the load module TESTMOD in the data set defined by the LOADLIB DD statement. It also specifies a title for each page of output, to be indented 20 characters from the left margin.

LISTLOAD Control Statement #2

instructs LIST to produce a module map and cross-reference listing of the load modules MOD1, MOD2, and MOD3 in the data set defined by the LOADLIB DD statement. It also specifies a title for each page of output, to be indented 20 characters from the left margin.

LISTLOAD Control Statement #3

instructs LIST to produce a formatted listing of the load module and its map and cross-reference listing. Because no DDN= parameter is included, the input data set is assumed to be the one defined by the SYSLIB DD statement. Because no MEMBER= parameter is specified, all load modules in the data set will be processed. This control statement also specifies a title for each page of output, to be indented 20 characters from the left margin.

Example 3: Listing IDR Information for Several Load Modules

In this example, LIST is used to list the CSECT identification records in several load modules.

```
//IDRLIST     JOB       MSGLEVEL=(1,1)
//LISTSTEP    EXEC      PGM=xMBLIST
//SYSPRINT    DD        SYSOUT=A
//SYSLIB      DD        DSN=SYS1.LINKLIB,DISP=OLD
//LOADLIB     DD        DSN=LOADMODS,DISP=OLD
//SYSIN       DD        *
    LISTIDR   TITLE=('IDR LISTINGS OF ALL MODS IN LINKLIB',20)
    LISTIDR   OUTPUT=IDENT,DDN=LOADLIB,MEMBER=TESTMOD
              TITLE=('LISTING OF MODIFICATIONS TO TESTMOD',20)
    LISTIDR   OUTPUT=ALL,DDN=LOADLIB,MEMBER=(MOD1,MOD2,MOD3),
              TITLE=('IDR LISTINGS OF MOD1 MOD2 MOD3',20)
/*
```

In this example:

JOB Statement

initiates the job.

EXEC Statement

calls for the execution of HMBLIST (in VS1) or AMBLIST (in VS2).

SYSPRINT DD Statement

defines the message data set.

SYSLIB DD Statement

defines the input data set SYS1.LINKLIB, which contains load modules to be processed.

LOADLIB DD Statement

defines a second input data set.

SYSIN DD Statement

defines the data set (in the input stream) containing the LIST control statements.

LISTIDR Control Statement #1

instructs LIST to list all CSECT identification records for all modules in SYS1.LINKLIB (this is the default data set since no DDN= parameter was included). It also specifies a title for each page of output, to be indented 20 characters from the left margin.

LISTIDR Control Statement #2

instructs LIST to list CSECT identification records that contain SPZAP or user-supplied data for load module TESTMOD. TESTMOD is a member of the data set defined by the LOADLIB DD statement. This control statement also specifies a title for each page of output, to be indented 20 characters from the left margin.

LISTIDR Control Statement #3

instructs LIST to list all CSECT identification records for load modules MOD1,MOD2, and MOD3. These are members in the data set defined by the LOADLIB DD statement. This control statement also specifies a title for each page of output, to be indented 20 characters from the left margin.

Example 4. Verifying an Object Deck

In this example, LIST is used to format and list an object module included in the input stream.

```
//LSTOBJDK    JOB       MSGLEVEL=(1,1)
//            EXEC      PGM=xMBLIST
//SYSPRINT    DD        SYSOUT=A
//OBJDECK     DD        *
    object deck
//SYSIN       DD        *
    LISTOBJ        DDN=OBJDECK,
         TITLE=('OBJECT DECK LISTING FOR MYJOB',25)
/*
```

JOB Statement

initiates the job.

EXEC Statement

calls for the execution of HMBLIST (in VS1) or AMBLIST (in VS2).

SYSPRINT DD Statement

defines the message data set.

OBJDECK DD statement

defines the input data set, which follows immediately. In this case the input data set is an object deck.

SYSIN DD statement

defines the data set containing LIST control statements, which follows immediately.

LISTOBJ Control Statement

instructs LIST to format the data set defined by the OBJDECK DD statement. It also specifies a title for each page of output, to be indented 20 characters from the left margin.

Example 5: Verifying Several Load Modules

Assume that an unsuccessful attempt has been made to link edit an object module with two load modules to produce one large load module. This example shows how to use LIST to verify all three modules.

```
//LSTLDOBJ     JOB        MSGLEVEL=(1,1)
//             EXEC       PGM=xMBLIST
//SYSPRINT     DD         SYSOUT=A
//OBJMOD       DD         DSN=MYMOD,DISP=OLD
//LOADMOD1     DD         DSN=YOURMOD,DISP=OLD
//LOADMOD2     DD         DSN=HISMOD,DISP=OLD
//SYSIN        DD         *
    LISTOBJ         DDN=OBJMOD,
         TITLE=('OBJECT LISTING FOR MYMOD',20)
    LISTLOAD        DDN=LOADMOD1,OUTPUT=BOTH,
         TITLE=('LISTING FOR YOURMOD',25)
    LISTIDR         DDN=LOADMOD1,OUTPUT=ALL,
         TITLE=('IDRS FOR YOURMOD',25)
    LISTLOAD        DDN=LOADMOD2,OUTPUT=BOTH,
         TITLE=('LISTING FOR HISMOD',25)
    LISTIDR         DDN=LOADMOD2,OUTPUT=ALL,
         TITLE=('IDRS FOR HISMOD',25)
/*
```

JOB Statement

initiates the job.

EXEC Statement

calls for the execution of HMBLIST (in VS1) or AMBLIST (in VS2).

SYSPRINT DD Statement

defines the message data set.

OBJMOD DD Statement

defines an input load module data set.

LOADMOD1 and LOADMOD2 DD Statements

define input load module data sets.

SYSIN DD Statement

defines the data set containing LIST control statements, which follows immediately.

LISTOBJ Control Statement

instructs LIST to format the data set defined by the OBJMOD DD statement. It also specifies a title for each page of output, to be indented 20 characters from the left margin.

LISTLOAD Control Statement #1

instructs LIST to fromat all records associated with the data set defined by the LOADMOD1 DD statement. It also specifies a title for each page of output, to be indented 25 characters from the left margin.

LISTIDR Control Statement #1

instructs LIST to list all CSECT identification records associated with the data set defined by the LOADMOD1 DD statement. It also specifies a title for each page of output, to be indented 25 characters from the left margin.

LISTLOAD Control Statement #2

instructs LIST to format all records associated with the data set defined by the LOADMOD2 DD statement. It also specifies a title for each page of output, to be indented 25 characters from the left margin.

LISTIDR Control Statement #2

instructs LIST to list all CSECT identification records associated with the data set defined by the LOADMOD2 DD statement. It also specifies a title for each page of output to be indented 25 characters from the left margin.

Example 6: Listing a Working Nucleus and Mapping the Link Pack Area

This example shows how to use the LISTLOAD and LISTLPA control statements to list a working nucleus and map the link pack area (VS2) or reenterable load module area (VS1). Note that in this example the data set containing the nucleus is named SYS1.NUCLEUS, and the nucleus occupies the member named IEANUC01.

```
//LISTNUC      JOB  MSGLEVEL=(1,1)
//STEP         EXEC PGM=xMBLIST
//SYSPRINT     DD   SYSOUT=A
//SYSLIB       DD   DSN=SYS1.NUCLEUS,DISP=OLD,UNIT=3330
//SYSIN        DD   *
    LISTLOAD        DDN=SYSLIB,MEMBER=IEANUC01,
         TITLE=('LISTING FOR NUCLEUS IEANUC01',25)
    LISTLPA
/*
```

JOB Statement

initiates the job.

EXEC Statement

calls for the execution of HMBLIST (in VS1) or AMBLIST (in VS2).

SYSPRINT DD Statement

defines the message data set.

SYSLIB DD Statement

defines the input data set, which in this case contains the nucleus.

SYSIN DD Statement

defines the data set containing LIST control statements, which follows immediately.

LISTLOAD Control Statement

instructs LIST to format the control and text records including the external symbol dictionary and reloaction dictionary records of the load module IEANUC01 in the data set defined by the SYSLIB DD statement. It also specifies a title for each page of output, to be indented 25 characters from the left margin.

LISTLPA Control Statement

instructs LIST to map the reenterable load module area or the link pack area.

Chapter 4: OSJQD →

Operates as a problem program to format and print the system job queue. (VS2 Only)

Contents

OSJQD

Figures

Introduction

OSJQD is an OS/VS2 service aid that formats and prints the contents of the system job queue data set (SYS1.SYSJOBQE). OSJQD is similar in function to the stand-alone service aid IMCJOBQD, provided in OS/VS1; however, OSJQD operates as a problem program under OS/VS2, using standard access methods. OSJQD can therefore be used without disrupting normal operating system processing; this is a great advantage in a large installation where stopping and restarting the operating system can take a long time.

To save even more time, you can specify that OSJQD output should be stored temporarily on tape rather than printed immediately. The tape can be printed later, at your convenience.

You can use OSJQD to dump the entire job queue, or you can select specific queues within the job queue and their associated logical tracks.

Starting OSJQD

OSJQD resides in the linkage library (SYS1.LINKLIB data set). You can invoke it either through job control statements in the input stream or through the system console.

In almost every case you will run OSJQD to produce a listing that will help you diagnose a problem connected with the job queue. If the problem is relatively minor, and the system can continue processing, you can schedule OSJQD immediately. For more severe problems, when the operating system cannot continue processing, you must restart the system before running OSJQD.

Restarting the System

If the system goes down, first try a system restart (warm start); that is, IPL without reformatting the job queue. If the restart fails, take action as suggested below:

If your installation has a volume containing an alternate SYS1.SYSJOBQE data set, restart the system, requesting that that volume be formatted as the new job queue data set. Then run OSJQD, specifying the original job queue data set as input.

If your installation has more than one operating system, and time is not critical, mount the volume containing the job queue on another system. Then run OSJQD on that system, specifying the transferred data set as input.

If you cannot use an alternate volume, or if the volume containing the job queue data set cannot be moved, dump the job queue data set to another direct access volume with a different volume serial number, as follows:

1. Execute the IBCDMPRS utility to dump the SYS1.SYSJOBQE data set to a direct access device. Use IBCDMPRS control statements like those shown in the following example:

```
DUMP JOB DUMP 2314 ONTO 2314
     DUMP FROMDEV=2314,FROMADDR=230,
          TODEV=2314,TOADDR=232,
          VOLID=ALTQUE
     END
```

   For more information about the IBCDMPRS utility program, refer to the publication OS/VS Utilities, GC35-0005.

2. Restart the operating system, specifying that the job queue should be reformatted. This will establish a fresh job queue.

4. Run OSJQD, specifying the new direct access data set as input.

Invoking OSJQD by JCL

Figure OSJQD-1 shows an example of job control statements used to invoke OSJQD. The statements are described below.

```
//DUMP               JOB        MSGLEVEL=(1,1)
//                   EXEC       PGM=IMCOSJQD
//OSJQDIN            DD         DSNAME=SYS1.SYSJOBQE,
//               UNIT=2314,VOL=SER=111111,DISP=SHR
//OSJQDOUT           DD         UNIT=2400,DISP=(NEW,KEEP),
//               DSNAME=QUEUEOUT,LABEL=(,NL)
//SYSPRINT           DD         SYSOUT=A
[//SYSIN             DD         *]
          .
          .
          .
/*
```

Figure OSJQD-1. An Example of Job Control Statements Used to Invoke OSJQD

EXEC Statement

calls for the execution of OSJQD.

OSJQDIN DD Statement

defines the job queue to be processed. Note that the DD statement that defines the input data set imustl be named OSJQDIN.

OSJQDOUT DD Statement

defines the output data set. In this case the output data set, named QUEUEOUT, resides on a tape device. Note that the DD statement that defines the output data set must be named OSJQDOUT.

SYSPRINT DD Statement

defines the OSJQD message data set.

SYSIN DD Statement (optional)

defines the data set that contains OSJQD options. In this case, the options follow the job control statements in the input stream. If this statement is omitted, the operator will be prompted to supply options.

Invoking OSJQD from the System Console

If you wish, you can include the job control statements shown in Figure OSJQD-1 as a cataloged procedure in the procedure library (SYS1.PROCLIB data set); this allows the operator to initiate OSJQD processing from the console.

Use the IEBUPDTE Utility to include your OSJQD cataloged procedure in SYS1.PROCLIB. The name you specify in the ADD control statement for IEBUPDTE is the name of the procedure that you must specify in the START command. For information on using IEBUPDTE, refer to the publication OS/VS Utilities, GC35-0005.

Figure OSJQD-2 shows an example of a cataloged procedure that calls OSJQD.

```
//OSJBQDMP          PROC      REG=20,D='SYS1.SYSJOBQE',U=2314,VS=111111,
//        DSP=SHR,UN=2400,DISP=(NEW,KEEP),DSN=QUEUEOUT
//                  EXEC      PGM=IMCOSJQD,REGION=&REG.K
//OSJQDIN           DD        DSNAME=&D,UNIT=&U,VOL=SER=&VS,DISP=&DSP
//OSJQDOUT          DD        UNIT=&UN,DISP=&DISP,DSNAME=&DSN
//SYSPRINT          DD        SYSOUT=A
/*
```

Figure OSJQD-2. An Example of a User-Written Cataloged Procedure to Call OSJQD from the System Console

PROC Statement

defines the name of the cataloged procedure and default values for any symbolic parameters included in the remaining statements in the procedure. In this case, the defaults are as follows: the input data set is SYS1.SYSJOBQE, the output data set is QUEUEOUT, and the region size is 20K. Note that you can specify any name for the procedure on the PROC statement.

EXEC Statement

calls for the execution of OSJQD, and specifies the region size by a symbolic parameter. (The default region size specified in the PROC statement is 20K; this is the minimum region size required for OSJQD processing.)

OSJQDIN DD Statement

defines the input data set. In this case, symbolic parameters permit the operator to specify an input data set or accept the defaults specified in the PROC statement.

OSJQDOUT DD Statement

defines the output data set. In this case, symbolic parameters permit the operator to specify an output data set or accept the defaults specified in the PROC statement.

SYSPRINT DD Statement

defines the message data set.

Note that the SYSIN DD statement has been omitted from this cataloged procedure; as a result the operator will be prompted to supply options when he starts OSJQD.

Figure OSJQD-3 shows an example of an exchange between the operator and OSJQD while starting OSJQD. Note that in this example the operator made an error the first time he selected dump parameters, and OSJQD prompted him to correct his error.

```
start osjbqdmp,,,reg=24
     .
     .
     .
   00 IMC001A SPECIFY SELECT PARAMETERS OR END
r 00,'qcr=cls=c'
   01 IMC002A COMMAND ERROR - ENTER QDUMP PARAMETERS
r 01,'qcr=class=c'
   00 IMC001A SPECIFY SELECT PARAMETERS OR END
r 00,'qcr=class=g'
     .
     .
     .
IMC005I SPECIFIED QUEUE IS EMPTY
   02 IMC001A SPECIFY SELECT PARAMETERS OR END
r 02,'qcr=class=a,jobname=(myjob,youjob,hisjob)'
     .
     .
     .
IMC006I THESE JOBS NOT FOUND
HISJOB
   03 IMC001A SPECIFY SELECT PARAMETERS OR END
r 03,'qcr=class=a,jobname=(myjob,herjob)'
     .
     .
     .
   04 IMC001A SPECIFY SELECT PARAMETERS OR END
r 04,'end'
     IMC004I QDUMP COMPLETE
```

Figure OSJQD-3. A sample exchange between operator and OSJQD.

Controlling OSJQD

You control OSJQD processing by defining the input data set and by supplying control statements.

Defining the Input Data Set

In most cases, the input to OSJQD will be the system job queue, SYS1.SYSJOBQE. However, OSJQD will accept as input any data set on a direct access device that has the format of the system job queue. This feature is useful when you have transferred the contents of the SYS1.SYSJOBQE data set to another volume, as described earlier in "Preparing to Use OSJQD".

Using the Control Statements

Several control statements allow you to specify how much of the job queue you want to format and print. You can enter these control statements in two ways:

- If you invoke OSJQD with JCL and include a SYSIN DD *, you can include control statements as cards in the input stream. If you want more than one dump operation, you must supply a separate card for each dump. OSJQD will process the cards sequentially and produce a separate output listing for each one. (Blank cards will be ignored.) OSJQD will terminate when it reaches end-of-file.

- If you start OSJQD from the console, or if you omit the SYSIN DD * statement from the JCL, OSJQD will prompt you to supply dump options. In reply you should define one dump operation fully. OSJQD will prompt you again when it has finished processing the first dump, and you can then define a new dump operation. If you want to terminate OSJQD processing, you must wait for a prompting message and reply END. (See Figure OSJQD-3.)

There are four OSJQD control statements: QCR= , JOBNAME= , ALL, and END.

```
QCR={CLASS=y  }
    {FREE     }
    {HOLD     }
    {SYSOUT=x }
    {SUBMIT   }
```

specifies that the job queue data set's master queue control record and the queue records associated with the named work queue should be formatted and printed. The parameters are mutually exclusive; if you want more than one specific work queue, you must request separate dump operations for each.

For each QCR= option, OSJQD dumps the master queue control record, the requested minor queue control record, and the logical tracks associated with that minor queue. The QCR= options and the minor queue control records they request are as follows:

CLASS=y	- An input job queue (A through O)
FREE	- Free Track Queue
HOLD	- Hold Queue
SYSOUT=x	- An output job queue (A through Z and 0 through 9)
SUBMIT	- TSO Background Reader Queue

JOBNAME=(jobname1[...,jobname4])

requests OSJQD to search all fifteen input work queues for logical track areas assigned to the specified jobname(s). These will be dumped along with associated system message blocks and data set blocks.

Note that searching all the input work queues for a job is a time-consuming operation. To reduce this time, use the QCR=CLASS=x control statment in combination with the JOBNAME= control statement to specify the input class of the requested job(s). For this purpose both control statements may be coded on a single card or entered as a single reply to a prompting message. An example of such an entry is:

```
QCR=CLASS=B,JOBNAME=(NEWJOB)
```

ALL

requests a dump of the entire job queue. This is the default option; it will take effect if the operator replies to the message prompting him for dump options by entering r xx,'U'.

OSJQD Output

OSJQD output can be directed either to a printer device or to a scratch tape, from which it can be printed later. Immediate printing can take a long time, so in most cases you should direct OSJQD's output to a tape.

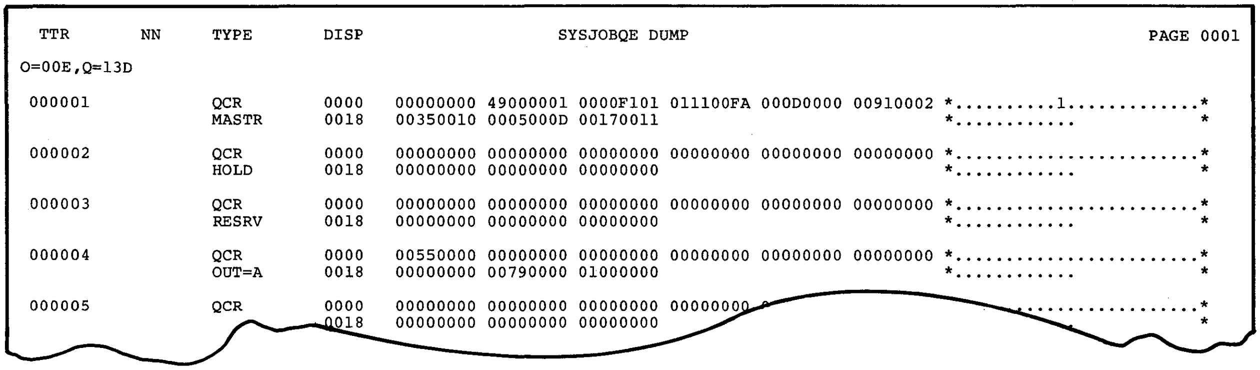
Once OSJQD's output is on a scratch tape, you can print it at any time using IEBPTPCH. Figure OSJQD-4 shows an example of the job control statements needed for this operation. For more information, refer to the publication OS/VS Utilities, GC35-0005.

```
//PRINT       JOB       MSGLEVEL=(1,1)
//            EXEC      PGM=IEBPTPCH
//SYSPRINT    DD        SYSOUT=A
//SYSUT1      DD        UNIT=2400,LABEL=(,NL),VOL=SER=QDUMPT,
//       DISP=(OLD,KEEP),DCB=(RECFM=F,BLKSIZE=121,LRECL=121)
//SYSUT2      DD        SYSOUT=A
//SYSIN       DD        *
         PRINT     PREFORM=M
/*
```

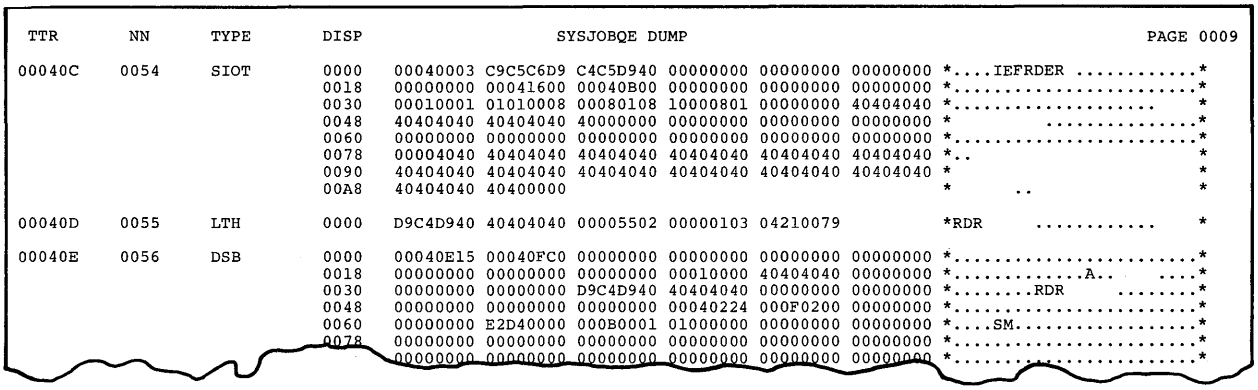
Figure OSJQD-4. Sample JCL and Control Statements Used to Print a 9-Track Tape Containing OSJQD Output

Figure OSJQD-5 shows a sample listing of a job queue as produced by OSJQD.

For a description of the fields in OSJQD output, refer to OS/VS2 Debugging Guide, GC28-0632.



```
  TTR        NN      TYPE       DISP                    SYSJOBQE DUMP                                           PAGE 0001
O=00E,Q=13D

 000001              QCR        0000     00000000 49000001 0000F101 011100FA 000D0000 00910002 *..........1.............*
                     MASTR      0018     00350010 0005000D 00170011                            *............            *

 000002              QCR        0000     00000000 00000000 00000000 00000000 00000000 00000000 *........................*
                     HOLD       0018     00000000 00000000 00000000                            *............            *

 000003              QCR        0000     00000000 00000000 00000000 00000000 00000000 00000000 *........................*
                     RESRV      0018     00000000 00000000 00000000                            *............            *

 000004              QCR        0000     00550000 00000000 00000000 00000000 00000000 00000000 *........................*
                     OUT=A      0018     00000000 00790000 01000000                            *............            *

 000005              QCR        0000     00000000 00000000 00000000 0000000                         ..................*
                                0018     00000000 00000000 00000000                                                   *
```



```
 TTR        NN       TYPE       DISP                    SYSJOBQE DUMP                                           PAGE 0009

00040C     0054      SIOT       0000     00040003 C9C5C6D9 C4C5D940 00000000 00000000 00000000 *....IEFRDER ............*
                                0018     00000000 00041600 00040B00 00000000 00000000 00000000 *........................*
                                0030     00010001 01010008 00080108 10000801 00000000 40404040 *....................    *
                                0048     40404040 40404040 40000000 00000000 00000000 00000000 *         ...............*
                                0060     00000000 00000000 00000000 00000000 00000000 00000000 *........................*
                                0078     00004040 40404040 40404040 40404040 40404040 40404040 *..                      *
                                0090     40404040 40404040 40404040 40404040 40404040 40404040 *                        *
                                00A8     40404040 40400000                                     *      ..                *

00040D     0055      LTH        0000     D9C4D940 40404040 00005502 00000103 04210079          *RDR     ............    *

00040E     0056      DSB        0000     00040E15 00040FC0 00000000 00000000 00000000 00000000 *........................*
                                0018     00000000 00000000 00000000 00010000 40404040 00000000 *.............A..    ....*
                                0030     00000000 00000000 D9C4D940 40404040 00000000 00000000 *........RDR     ........*
                                0048     00000000 00000000 00000000 00040224 000F0200 00000000 *........................*
                                0060     00000000 E2D40000 000B0001 01000000 00000000 00000000 *....SM..................*
                                0078     00000000 00000000 00000000 00000000 00000000 00000000 *........................*
                                         00000000 00000000 00000000 00000000 00000000 00000000 *........................*
```

Sample OSJQD-5. Sample OSJQD Output,

Output Comments

OSJQD does not dump records that consist entirely of binary zeroes. Instead, when it comes to an all-zero record, it prints

    ENTIRE RECORD CONTAINS BINARY ZEROES

and supplies TTR and NN information as described in the previous section. If OSJQD comes to subsequent all-zero records, it will stop printing records until it comes to the next non-zero record or the next logical track header record. To indicate that all-zero records are not being printed, OSJQD prints

    ZERO RECORDS SUPPRESSED

Error Recovery Procedures

OSJQD error recovery depends on what kind of dump is being produced, what record was being read when the error occurred, and how many times the error has already occurred.

If you have requested a full dump (by specifying ALL when starting OSJQD), OSJQD will attempt to recover from all errors except those that occur while reading the master queue control record. To recover, OSJQD prints an output error indicator, attempts to print the record associated with the error, and proceeds by reading the next record. If OSJQD could not read the record associated with the error, it prints an appropriate output error indicator on the output listing, and then continues processing with the next queue record.

OSJQD will permit up to 20 consecutive errors to occur before abandoning its attempts to recover. After the twentieth consecutive error, it will issue message IMC016I (PERMANENT I/O ERROR ON OSJQDIN), print the contents of the SYNAD buffer, and obtain the next dump option.

If you have requested a selective dump, or if an error occurs while reading the master queue control record, OSJQD does not attempt to recover from any errors. It prints the record associated with the error or an output error indicator, issues message IMC016I, prints the contents of the SYNAD buffer, and obtains the next dump option. It does this by searching the SYSIN data set, if control statements were entered from the input stream, or by prompting the operator to supply dump options, if control statements were entered from the console. It will not terminate processing unless it encounters an END control statement or an end-of-file condition.

The error messages and their meanings are as follows:

badttr - INVALID TTR

OSJQD will print this line in place of the record it could not find, followed by the contents of the SYNAD buffer.

UNABLE TO READ RECORD

An input/output error occurred while OSJQD was trying to read a queue record. OSJQD prints the TTR and NN values associated with the record, and substitutes this message for the contents of the record itself. The message is followed by the contents of the SYNAD buffer.

I/O ERROR READING FOLLOWING RECORD

An input/output error occurred while OSJQD was trying to read a queue record; the error did not prevent OSJQD from reading the record. OSJQD prints this message to indicate that the record contains an error, and follows it with the record itself and the contents of the SYNAD buffer.

INVALID LENGTH RECORD

OSJQD has encountered a record which is not a standard length (for a normal queue record, standard length is 176 bytes; for logical track header records, 20 bytes; for queue control records, 36 bytes). OSJQD prints this message, followed by the record and its associated TTR and NN values. No SYNAD information is included.

PRDM

JCL and Control Statement Examples

The following examples illustrate some of the functions that OSJQD can perform.

Example 1: Dumping the Input Job Queues

This example shows how to format and print three input job queues and two output job queues. Note that the only JCL statement shown is the SYSIN DD statement; for an example of the other JCL statements required to invoke OSJQD, see Figure OSJQD-1.

```
//SYSIN        DD        *
QCR=CLASS=A
QCR=CLASS=B
    QCR=CLASS=C
  QCR=SYSOUT=A
QCR=SYSOUT=B
/*
```

Note that each control statement requests a separate queue, and that the control statements are entered in free form.

Example 2: Searching the Input Job Queues for a Specific Job

This example shows how to combine the QCR= and JOBNAME= control statements to search a limited number of queues for specific jobs. Note that the only JCL statement shown is the SYSIN DD statement; for an example of the other JCL statements required to invoke OSJQD, see Figure OSJQD-1.

```
//SYSIN  DD   *
    QCR=CLASS=A,JOBNAME=(MYJOB,YOURJOB,HISJOB,HERJOB)
/*
```

Note that the maximum of four jobnames are specified in the JOBNAME= control statement.

Example 3: Dumping the Entire Job Queue

This example shows how to dump the entire job queue. Note that the only JCL statement shown is the SYSIN DD statement; for an example of the other JCL statements required to invoke OSJQD, see Figure OSJQD-1.

```
//SYSIN  DD   *
    ALL
/*
```

Coding the ALL control statement has the same effect as replying r xx,'U' to message IMC001A.

Chapter 5: PRDMP ————————————————————➤

PRDM

Formats and prints SADMP high-speed output (including page dump), SYS1.DUMP data set, and GTF trace data.

Contents

PRDM

Figures

Introduction

PRDMP is a service aid that prints system dump and trace information. Its principal function is to save you time; it does this by producing formatted output that you can scan quickly and easily. Within certain limits, it even allows you to suppress formatting and printing of information that does not interest you.

PRDMP can process the following kinds of input:

Dump data sets of OS/VS systems. These include:

- SADMP high-speed dump data set, which may include page data sets. Note: Address translation will be performed on the real storage dump portion if you request any format control statement except PRINT PAGE or PRINT REAL.

- SYS1.DUMP data set. This type of dump input will be processed by virtual addresses only.

- SADMP low-speed dump data set which has been written to tape.

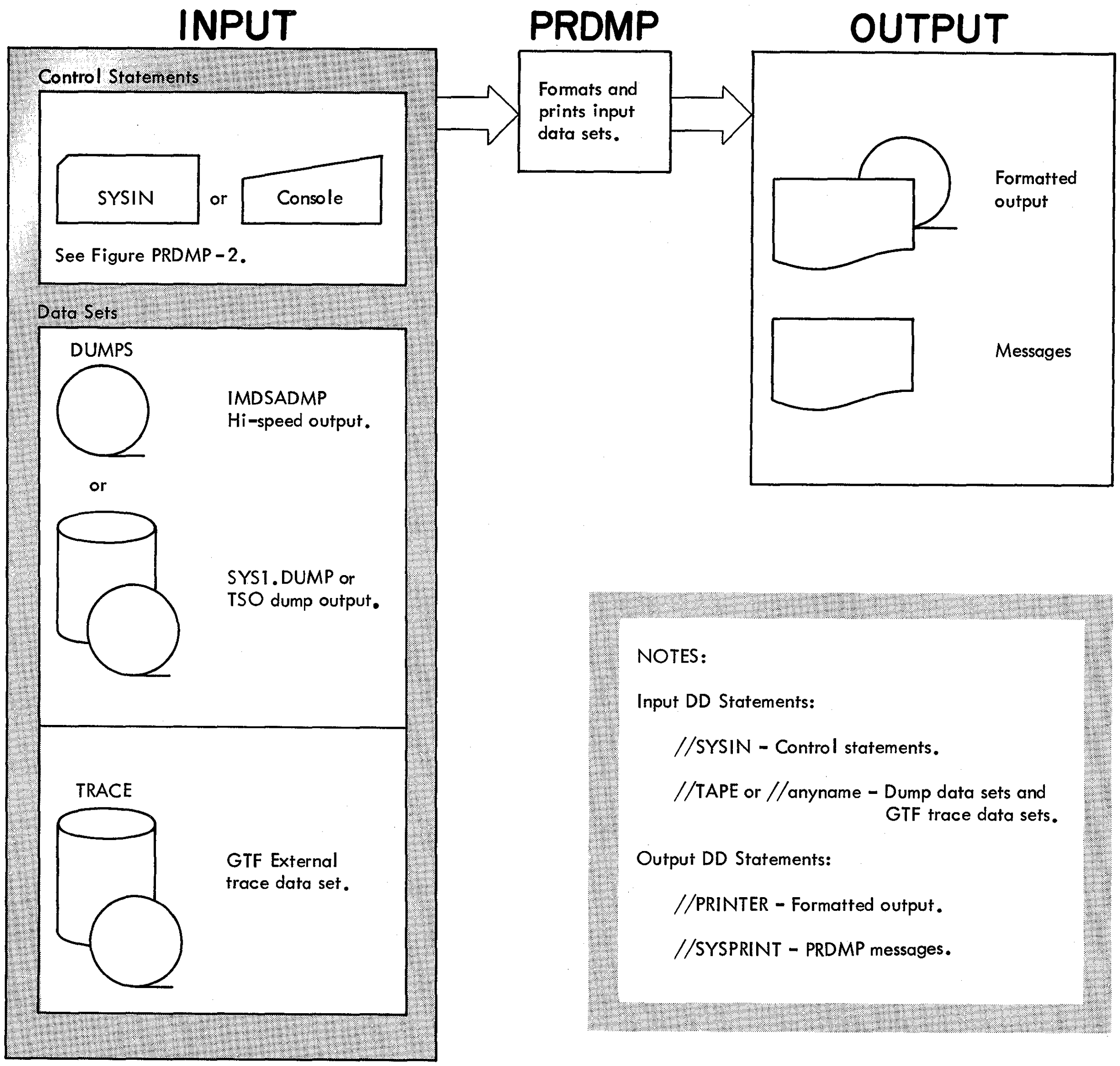
- TSO dumps (VS2 only), which may contain all or only selected portions of virtual storage, such as the nucleus, link pack area, or a region and its associated LSQA. These dumps will be processed by virtual addresses only.

GTF trace data. This may exist as:

- GTF external trace data set (usually called SYS1.TRACE).

- GTF trace data in buffers within a dump of real storage.

Figure PRDMP-1 shows the general characteristics of these types of input and how they relate to PRDMP processing.



NOTES:

Input DD Statements:

//SYSIN - Control statements.

//TAPE or //anyname - Dump data sets and GTF trace data sets.

Output DD Statements:

//PRINTER - Formatted output.

//SYSPRINT - PRDMP messages.

Figure PRDMP-1. PRDMP Input and Output.

You vary the formatting and printing of a dump by supplying PRDMP control statements. You can enter these either as replies to prompting messages issued to the console, or as cards in the input stream.

The control statements provide the following functions:

Data Area Formatting

You can specify one control statement (FORMAT) that will cause PRDMP to format some system data areas for each task in the system. Note: SADMP low-speed dump tapes can be printed using PRDMP, but they will not be formatted.

In VS2, the TSO control statement is provided to allow you to format data areas associated with tasks in the TSO subsystem.

Editing GTF Trace Data

PRDMP can format GTF trace data either as records in the trace data set or as buffers contained in a dump data set. You can edit trace data by specifying special keywords in the EDIT control statements. You can also write exit routines to inspect the data before PRDMP formats it, or user format appendages to process records generated by the GTRACE macro. Suggestions on how to write user exit routines and format appendages will be provided in Appendix A: Writing EDIT User Programs.

Clearing SYS1.DUMP

There are two ways you can use PRDMP to transfer the contents of the SYS1.DUMP data set to another data set and clear the SYS1.DUMP data set:

- You can transfer the contents of SYS1.DUMP to the data set defined by the SYSUT2 DD statement; PRDMP will not permit you to format or print the dump unless you define the receiving data set as input to a later step. For more information about this method, see Example 2.

- You can transfer the contents of SYS1.DUMP to the data set defined by the SYSUT1 DD statement and process the dump all in the same step. For more information about this method, see Example 3.

Printing Selectively

In a single control statement called PRINT, you can specify precisely what areas of real or virtual storage or what records from the page data sets you want PRDMP to print. Certain parameters of the PRINT control statement will cause PRDMP to format and print data areas that are associated with specified areas of virtual storage.

PRINT allows you to specify printing of virtual storage areas that are associated with:

- A certain jobname.
- The current task.
- (VS1 only) The task terminated by the damage assessment routine (DAR), where applicable.

You can also choose printing of the nucleus, system queue area, all or part of real or virtual storage, all or part of the page data sets, and all of allocated virtual storage.

Other control statements provide the following functions:

Mapping Reenterable System Modules (VS1 Only)

PRDMP can generate a reenterable load module area map. This map describes reenterable system modules that were loaded into virtual storage by the nucleus initialization program (NIP). If you request a map, it will be printed on a separate page or pages of the PRDMP formatted dump listing. These maps are useful in diagnosing system failures that occurred in program modules residing outside the user's partition.

Mapping the Active Link Pack Area (VS2 Only)

PRDMP can generate a map of the Link Pack Area active queue, which describes the reenterable system modules from the Link pack area that were in use when the dump was taken. If you request a map, it will be printed on a separate page or pages of the PRDMP formatted dump listing. These maps are useful in diagnosing system failures that occurred in program modules resideing outside the user's region. The entire link pack area may be mapped using AMBLIST, which is described in Chapter 3.

Tracing Queue Control Records

PRDMP can provide a separate listing of the formatted queue control blocks for all task control blocks in the system. This listing, known as a QCB trace, may be used to resolve problems arising from task contention or system interlock.

Job Control Language Statements

Job control statements are important in determining what functions PRDMP is to perform. This section describes the JCL statements that have special significance in executing PRDMP. For more complete information about using JCL statements, refer to the publication OS/VS JCL Reference, GC28-0618.

JOB Statement

initiates the job. In VS2, AMDPRDMP requires a minimum region size of 128K.

EXEC Statement

calls for the execution of HMDPRDMP (VS1) or AMDPRDMP (VS2) and specifies certain actions that PRDMP should take. The operands are:

$\text{PGM=}\left\{\begin{matrix}\text{HMDPRDMP}\\\text{AMDPRDMP}\end{matrix}\right\}$

identifies HMDPRDMP or AMDPRDMP to the system. This is the only required operand.

PARM='[n][,T][,LINECNT=nn][,S][,ER=x]'

n should be used only when the input is a dump data set. It specifies what PRDMP should do if it detects a permanent I/O error or format error while extracting data from the dump during its initialization processing.

0 -- print the nucleus (and the system queue area).

1 (or n not specified) -- print the entire virtual or real storage portion of the input data set.

2 -- read the next control card from the SYSIN data set, or request control statements from the operator.

If an error occurs when n is 0 or 1 and the input data set is a SADMP dump, PRDMP will print storage with real addresses only.

T specifies that the operator should be prompted to supply a title for the listing. The title may contain a maximum of 64 characters. If T is not specified, no prompting will occur.

LINECNT=nn specifies the number of lines per page to be printed on the output listing. The value specified for nn may be any decimal integer greater than 10. If this parameter is omitted, LINECNT=58 is assumed.

S instructs PRDMP to issue a message which the operator may reply to at any time during processing. By replying, the operator may stop PRDMP from processing the current input data set and start a new phase of PRDMP execution.

ER=x specifies what action the EDIT portion of PRDMP should take if it detects an error in an exit or format routine while editing trace data from a dump or trace data set. The valid values of x and their meanings are:

0 -- EDIT will display in hexadecimal the record associated with the error and ignore the faulty routine in subsequent processing. If the error was in a format routine, all subsequent records that require processing by the same format routine will be ignored. If the error was in an exit routine, record formatting will continue.

1 -- EDIT will display in hexadecimal the record associated with the error and ignore the faulty routine in subsequent processing. If the error was in a format routine, all subsequent records that require processing by the same format routine will be dumped in hexadecimal. If the error was in an exit routine, record formatting will continue.

2 -- EDIT will display in hexadecimal the record associated with the error; EDIT will then terminate, and the next PRDMP verb will be executed.

3 -- EDIT will allow ABEND to get control if a program check occurs in an exit or format routine. (If ER=3 is not specified, EDIT will issue the SPIE macro before entering the exit routine or format appendage and thus bypass ABEND processing.) If the recognized error is not a program check, the associated record will be dumped in hexadecimal; then EDIT will terminate and the next PRDMP verb will be executed.

If this value is not included in the PARM= parameter list, a value of ER=2 will be assumed. Note that ER=0 and ER=1 are the same for exit programs.

Input DD Statements

$\left\{\begin{array}{l}\text{TAPE}\\\text{anyname}\end{array}\right\}$ DD Statement

defines an input dump or trace data set, which may reside on single or (for VS2) multiple direct access storage volumes or on single or multiple tape volumes. If the input data set is a dump, you can specify any ddname. Remember, however, that for ddnames other than TAPE, you must use a NEWDUMP control statement to identify the input data set. You can define any number of input data sets, as long as each is identified by a different ddname, and each ddname is specified in a separate NEWDUMP control statement.

If the input is a GTF trace data set, the ddname must be the same as the one specified in the DDNAME parameter of the EDIT control statement. You can define any number of trace data sets, provided that you identify each data set with a unique ddname and a separate EDIT control statement. (Note that you can use the same ddname for several trace data sets, as long as you provide a new tape volume for each.)

If the input data set resides on a direct access device, or in VS1 if it spans multiple tape volumes, you must supply a SYSUT1 DD statement.

Use the following parameters to describe each input data set:

* DSNAME=name (for direct access only)

VOL=SER=(volser,volser...,volser) (for multiple volume dumps, specify the volume serial numbers in order)

```
    UNIT=ddd[,P]   (ddd may be either a device address, a device type,
                   or a group name; use P to request parallel mounting
                   of multi-volume input data sets.)

  * LABEL={([n],NL)}   (for tape only)
          {([n],SL)}

    DISP=OLD

    DCB=(BUFNO=number,BLKSIZE=size)  (for trace data sets only)
```

* If the input is a trace data set on a standard label tape, you must include the DSNAME= parameter and code the LABEL= parameter as LABEL=([n],SL), where n is the file number.

Use the DCB parameter to specify a greater blocksize or more input buffers, or both, if you think the default values will be inadequate. The default blocksize is 3500 bytes; the default number of buffers is 2.

Note: Do not omit the TAPE DD statement unless you supply a NEWDUMP control statement. If you do not define the input data set, PRDMP assumes that the input is in the SYSUT1 data set.

SYSIN DD Statement

defines the data set that contains the PRDMP control statements.

Output DD Statements

PRINTER DD Statement

defines the PRDMP output data set. For best performance, you should specify a blocking factor for this data set. To determine what blocking factor to use, see the section "Specifying the Maximum Output Block Size."

SYSPRINT DD Statement (Optional)

defines the PRDMP message data set.

SYSUT1 DD Statement

defines a direct access work data set in which PRDMP can collect input data. Performance may be improved when a SYSUT1 DD statement is included, because PRDMP can reference dump information directly on a direct access device faster than on a tape device.

This statement is required when (for VS1 only) input spans more than one volume or when the input resides on a direct access device. It is optional if input is a dump data set on a single tape. Do <u>not</u> use it if input is an external trace data set or if the job step already contains a SYSUT2 DD statement.

Required parameters are:

UNIT=ddd (ddd may be a device address, a device type, or a group name)

SPACE=(2056,(N,10))

In the SPACE= parameter, N is calculated as follows:

For a dump of real storage:

N = (K/2048) + 1 where

K = (maximum real storage address)

For a dump of virtual address space:

N = (K/2048) + 1 where

K = (maximum virtual address) - (minimum virtual address)

For a page data set dump (VS1 only):

N = 1 + (K) where

K = (number of pages represented in the page data set)

If the input dump data set contains more than one type of dump, the total value of N is the sum of N for the individual dump types.

Note that the SYSUT1 data set must reside on a single volume.

SYSUT2 DD Statement

identifies a data set onto which PRDMP may transfer the contents of the SYS1.DUMP data set, or any dump data set, when time will not permit immediate formatting and printing of the data set. Whenever the SYSUT2 DD statement is present in the input stream and the SYSUT1 DD statement is absent, any PRDMP format control statement will cause the input data set to be transferred to the SYSUT2 data set. For more information about this function, refer to Example 2.

Note: Do not use the SYSUT1 DD statement and the SYSUT2 DD statement in the same step.

SYSTSO DD Statement (VS2 Only)

defines the TSO work data set for AMDPRDMP. This statement is required only if the TSO control statement is used to request formatting of TSO user storage and/or data areas. It can define a tape or direct access data set, which must reside on a single volume; however, to save processing time, you should define this work data set as a direct access data set.

Use the following parameters to describe the data set.

DSNAME=name   (for direct access only)

VOL=SER=volser

UNIT=ddd   (ddd may be a device name, device type, or a group name)

```
LABEL= {([n],NL)}     (for tape only)
       {([n],SL)}
```

DISP=NEW

SPACE=(2056,(M,M/2)) (for direct access only)

M is calculated as follows:

For a dump produced by AMDSADMP:

$M = 8(T) + (S/2048)$ where

T = number of TSO users

S = Size of TSC region

This calculation is based on the assumption that 16K of LSQA per TSO user is being used.

For a dump produced by SVCDUMP:

$M = (R+S)/2048 + 1$ where

R = largest TSO region size

S = Size of TSC region

This is based on the assumption that the TSO region size does not include the LSQA.

Function Control Statements

Standard Form	Abbreviated Form

CVT= {hhhhhh}
{P}
	C= {hhhhhh}
	{P}

SEGTAB=hhhhhh	S=hhhhhh
NEWDUMP [DDNAME= {TAPE }] [,FILESEQ=nn] {anyname}	ND [DD= {TAPE }] [,F=nn] {anyname}
NEWTAPE	N
GO	G

ONGO [QCBTRACE] [,LPAMAP] [,FORMAT] [,CVT=parm]
{[,PRINT parm]}
{[,TSO parm]}
{[,EDIT parm]}
	O [Q] [,L] [,F] [,C=parm]
	{[,P parm]}
	{[,TSO parm]}
	{[,E parm]}

TITLE text	T text
END	EN

Format Control Statements

Standard Form	Abbreviated Form
QCBTRACE	Q
LPAMAP	L
FORMAT	F
PRINT [ALL] [,CURRENT] [,NUCLEUS] [,STORAGE=(addresses)] [,JOBNAME=(parm)] [,F03] [,REAL=(addresses)] [,PAGE=parm]	P [A] [,C] [,N] [,S=(addresses)] [,J=(parm)] [,F] [,R=(addresses)] [,P=parm]

TSO [SYSTEM= {YES }] [,USER= {PRINT }]
{USER} {STORAGE}
{NO } {FORMAT}
{NO}
	TSO [S= {YES }] [,U= {PRINT }]
	{USER} {STORAGE}
	{NO } {FORMAT}
	{NO}

EDIT parm	E parm

Figure PRDMP-2. PRDMP Function and Format Control Statements, Standard and Abbreviated Forms.

User Control Statements

User control statements allow you to select specific dump formatting options and control basic operation of the PRDMP program.

PRDMP will prompt you to supply control statements if no SYSIN data set exists, or if the supply of control statements in the SYSIN data set is exhausted before PRDMP finds an END control statement. Note: If you enter control statements on cards in the input stream, do not mark the cards with sequence numbers. PRDMP scans all 80 columns of any card in the input stream, and may mistake sequence numbers for invalid keywords.

There are two kinds of user control statements: function control statements and format control statements. All the control statements are fully described below. Figure PRDMP-2 shows the complete format of the function control statements.

Function Control Statements

The function control statements allow you to control certain operations of the PRDMP program, such as input tape handling, dump listing titles, and job termination.

CVT=$\left\{\begin{array}{l}\text{hhhhhh}\\ \text{P}\end{array}\right\}$

allows you to specify the address of the communications vector table (CVT) in the virtual storage dump information. Use this if you think that the CVT pointer in virtual storage location X'4C' of the system that was dumped has been destroyed. If you omit this control statement, and PRDMP cannot locate the CVT at location X'4C', it will scan the dump data set for unique identifiers associated with the CVT. If PRDMP cannot locate the CVT by this scanning process, it will not format the input but will instead take action as specified by "n" in the parameter list supplied in the PARM= operand of the EXEC statement. Once the CVT has been located, it remains in effect until a NEWDUMP, NEWTAPE, or another CVT= control statement is encountered.

hhhhhh

is a hexadecimal address specifying the location of the CVT in the input dump information.

P

specifies that the location found at X'4C' in the system on which PRDMP is being executed can be used as a valid pointer to the CVT in the dumped system.

SEGTAB=hhhhhh

allows you to specify the hexadecimal real storage address of the segment table. Use this control statement if you have forgotten to perform the store status operation before executing the stand-alone dump program (SADMP). In VS1, if you do not provide the segment table address either by performing the store status operation or by using the SEGTAB= control statement, PRDMP will be unable to provide any address translation. In VS2, if you do not perform the store status operation, PRDMP will search the CVT for the address of the system segment table; the SEGTAB= control statement should be used as insurance against the possibility that the CVT may be unreliable.

The SEGTAB= control statement must precede all format control statements to be useful when processing a given dump.

```
NEWDUMP  DDNAME={TAPE    }[,FILESEQ=nn]
                {anyname }
```

defines an input data set. If you want to process more than one input data set in a single execution of PRDMP, you must supply a separate NEWDUMP or NEWTAPE control statement for each. If there is only one input data set, defined by the ddname TAPE, NEWDUMP is not needed.

NEWDUMP has two keyword parameters:

DDNAME=

gives the ddname of the input dump data set. The ddname used in this parameter must differ from the ddnames associated with the permanent data sets used by PRDMP, such as SYSUT1, PRINTER, SYSPRINT, etc. Otherwise unpredictable results may occur. This parameter is not required if the TAPE DD statement describes the input data set.

FILESEQ=

identifies the sequence number of an input data set that is one of several data sets on a single magnetic tape volume. If this parameter is omitted, PRDMP assumes a default value of FILESEQ=1.

NEWTAPE

has the same function as the NEWDUMP statement with parameters specified as DDNAME=TAPE and FILESEQ=1.

```
ONGO [QCBTRACE][,LPAMAP][,CVT=parm][,SEGTAB=parm][,FORMAT][,PRINT  parm]

       [,EDIT parm]
```

overrides the predefined set of format control statements requested by the GO control statement, which must follow it in the input stream. The new set of format control statements will remain in effect for all subsequent uses of the GO control statement, until PRDMP ends or a new ONGO control statement is entered. An ONGO control statement with no parameters restores the original GO functions: QCBTRACE, LPAMAP, FORMAT, EDIT, and PRINT ALL. Note that in using the ONGO control statement you must conform to the rules for combining control statements as defined later in this chapter.

GO

specifies a predefined set of format control statements. They are: QCBTRACE, LPAMAP, FORMAT, EDIT, and PRINT ALL. The effects of the GO control statement may be overridden by the ONGO control statement.

NOTE: The ONGO-GO combination is not required for PRDMP execution. You need not specify GO unless you want to use a predefined set of PRDMP options; you need not use ONGO unless you want to change that predefined set. Each PRDMP control statement may be specified directly at any time.

TITLE text

specifies a title that is to be printed at the top of each page in the output listing. Use this statement if you do not expect PRMDP to prompt you to supply title information; that is, if you did not specify T in the PARM= field of the EXEC statment. You can specify any title up to 64 characters in length.

END

signals PRMDMP to stop processing, close all data sets, and return control to the system. (If END is the only control statement specified, PRDMP will load the data set defined by the SYSUT2 DD statement, if present. See Example 2.)

Format Control Statements

Format control statements allow you to choose particular parts of the input to be formatted and printed.

Note that if input is an SVC dump and the CVT can not be found, PRDMP only processes PRINT STORAGE and PRINT NUCLEUS control statements. This situation occurs when the console dump does not include the supervisor areas in the specified storage ranges or in the SDATA operand.

QCBTRACE

requests a trace of the queue control blocks (QCBs) in the input data set.

LPAMAP

causes PRDMP to map the reenterable load module area (for VS1) or the link pack area active queue (for VS2) in the input data set. If the input data set does not contain this area, LPAMAP wlll cause an error message to be printed.

Note that this control statement maps only the active modules in the reenterable load module area or the link pack area; it does not include any storage associated with the area. If you want a map of storage, you must use the PRINT STORAGE= control statement.

FORMAT

causes PRDMP to format and print the contents of the following system data areas in the input data set:

All Task Control Blocks (TCBs)
All Request Blocks (RBs)
All Problem Program Boundaries
Load List
Job Pack Queue
All Data Extent Blocks (DEBs)
All Task Input/Output Tables (TIOTs)

```
PRINT     [ALL][,CURRENT][,NUCLEUS][,STORAGE=(addresses)]
          [,JOBNAME=(jobnames)][,F03][,REAL=(addresses)]
          [,PAGE=ddd=[(ttrs)][,PAGE= {[cuu]} =[(sggs)]
                                     {[dn] }
```

indicates which parts of the input data set PRDMP should print, according to several parameters.

ALL (allocated storage)

instructs PRDMP to print the resident nucleus, the system queue area, the pageable nucleus (VS1 only), and all virtual storage allocated to partitions or regions in the input data set. This parameter also requests printing of the dumped system's registers and current PSW, if available. However, it does not request printing of any module in the reenterable load module area or link pack area.

CURRENT

instructs PRDMP to format and print only the area of virtual storage that was associated with the current task when the input data set was created. This parameter also requests printing of the dumped system's registers and current PSW, if available, and formatting of the following data areas for the current task:

Task Control Block (TCB)
Request Blocks (RBs)
Problem Program Boundaries
Load List
Job Pack Queue
Data Extent Block (DEB)
Task Input/Output Table (TIOT)

Note that if the dump was produced by SVC Dump and reflects only selected portions of storage, the current task formatted may not be the current task as pointed to by the NEW/OLD task pointer in the nucleus. Note also that the dumped system's registers will not be printed if the current task is the dummy wait task.

NUCLEUS

instructs PRDMP to print the resident nucleus, the system queue area and (for VS1 only) the pageable nucleus portion of the input data set.

STORAGE=(addresses)

allows you to supply beginning and ending virtual addresses of areas in the input data set that you want printed, in the form:

```
(start1,end1[,start2,end2]...[,startn[,endn]])
```

You may specify a single address or any number of pairs of 1- to 6-character hexadecimal addresses, so long as the beginning address in each pair is lower than the ending address. If the beginning address of any pair is greater than the ending address, the pair is ignored.

If you omit the ending address from the last address pair in a sequence, or if you only specify a single address, PRDMP prints the entire contents of virtual storage starting at the last address you specify.

If you do not specify any addresses, PRDMP will print the entire contents of virtual storage, whether allocated or not, and will also print the dumped system's registers.

JOBNAME=(jobnames)

allows you to limit the scope of the output listing to areas in virtual storage that are associated with specific jobs. You can specify up to ten jobnames, using the form:

```
(jobname1[,jobname2]...[,jobname10])
```

PRDMP will print the areas associated with each job name. It will also format the following data areas associated with these areas:

Task Control Block (TCB)
Request Blocks (RBs)
Problem Program Boundaries
Load List
Job Pack Queue
Data Extent Block (DEB)
Task Input/Output Table (TIOT)

F03 (VS1 Only)

instructs PRDMP to print areas of virtual storage that were associated with all tasks terminated by the damage assessment routine (DAR). It also requests formatting of the following data areas associated with these areas:

Task Control Block (TCB)
Request Blocks (RBs)
Problem Program Boundaries
Load List
Job Pack Queue
Data Extent Block (DEB)
Task Input/Output Table (TIOT)

REAL=(addresses)

allows you to supply ranges of real storage addresses to be printed, using the form:

```
(start1,end1[,start2,end2]...[,startn[,endn]])
```

You may specify a single address or any number of pairs of 1- to 6-character hexadecimal addresses, so long as the beginning address in each pair is lower than the ending address. If the starting address of any pair is greater than the ending address, the pair is ignored.

If you omit the ending address from the last address pair in a sequence, or if you only specify a single address, PRDMP prints the entire contents of real storage starting at the last address you specify.

If you do not specify any addresses, PRDMP will print the entire contents of real storage, whether allocated or not, and will also print the dumped system's registers.

This parameter will casue an error message to be printed if the input data set contains a virtual storage dump -- that is, a dump not produced by HMDSADMP.

PAGE=cuu=(ttrs)   (VS1 Only)

allows you to specify TTR ranges within the page data set that you want PRDMP to print, using the form:

```
cuu=(start1,end1[,start2,end2]...[,startn[,endn]])
```

cuu specifies the unit address of the device where the page data set resided when the dump was taken; this allows you to distinguish between two or more page data sets possibly residing on the same input dump data set.

Note that if the device is a 2305-2, ddd may be any one of eight different addresses (for example, 1D0-1D7). The address you specify in this statement must be the physical address of the device.

To determine the specific TTR address(es) that you need to specify to format and print paged-out data, refer to the publication OS/VS1 Debugging Guide, GC24-5093.

You may specify a single TTR or any number of pairs of TTR addresses, so long as the beginning TTR in each pair is lower than the ending TTR. If the starting TTR of any pair is greater than the ending TTR, the pair is ignored.

If you omit the ending TTR from the last pair in a sequence, or if you only specify a single TTR, PRDMP prints the entire contents of the page data set starting at the last TTR you specify. For each TTR, PRDMP will print one 2K page.

If you do not specify any TTR addresses, PRDMP will print the entire page data setfor the device specified. If you omit all subparameters of the PAGE= parameter, PRDMP will print all page data sets in the input data set.

```
PAGE{=cuu=(sggs)            }        (VS2 Only)
    { =dn=(sgg1,sgg2...,sggn)}
```

allows you to specify SGGs (Slot Group Group numbers) within the page data set that you want PRDMP to print. To determine the specific SGGs that you need to specify to format and print paged-out data, refer to the publication OS/VS2 Debugging Guide, GC28-0632. Note that there are two forms of this control statement, of which the first is:

```
PAGE=cuu=(start1,end1[,start2,end]...[,startn[,endn]])
```

This allows you to specify a single SGG or any number of ranges of SGGs that you want PRDMP to print. cuu specifies the 3-digit unit address of the device where the page data set resided when the dump was taken; this allows you to choose the correct page data set when there are more than one residing in the same input data set.

You may specify a single SGG or any number of pairs of SGGs, as long as the beginning SGG in each pair is lower than the ending SGG. If the starting SGG of any pair is greater than the ending SGG, the pair is ignored.

If you omit the ending SGG from the last pair in a sequence, or if you specify only a single SGG, PRDMP prints the entire contents of the page data set starting at the last SGG specified. (Note that PRDMP treats SGGs as if they were specified as GGS when checking the validity of pairs.)

The second form of the PAGE= control statement is:

```
PAGE=dn=(SGG1,SGG2...,SGGn)
```

This allows you to specify single pages from the page data set that you want PRDMP to print. dn specifies the page device number for the paging device as found in the Page Device Table; it can be specified as a one- or two-character value.

You may specify any number of single SGGs, in any order. For each SGG, PRDMP will print one 4K page of storage. If you specify no SGGs, PRDMP will print the entire dump for that relative device number.

If you specify a slot or group of zero, PRDMP will treat it as though you had specified a slot or group of one. If you specify an SGG as 000, PRDMP will treat it as though you had specified 101.

To save processing time, you should specify PRINT PAGE= as the last control statement in a series for a given dump. Also, when you specify devices or slot groups (SGGs), you should specify them in the same order in which they were dumped.

```
TSO [SYSTEM={YES }] [,USER={PRINT  }]          (VS2 Only)
            {USER}         {STORAGE}
            {NO  }         {FORMAT }
                           {NO     }
```

instructs PRDMP to format and/or print storage for tasks in the TSO subsystem. Two parameters allow you to limit the amount of formatting that PRDMP will do. If you omit a parameter, PRDMP will give you maximum formatting.

SYSTEM=

defines the extent of formatting for TSO system data areas. The default value is SYSTEM=YES; it causes PRDMP to format the following data areas:

TCB family for TSC
TSCVT
RCBs for each TS region
Active TJBs
Active TSBs
SPCTs for each TS region and each user
SPCAs for each TS region

If you specify SYSTEM=USER, PRDMP will format only active TJBs, active TSBs, and user SPCTs. If you specify SYSTEM=NO, PRDMP will not format any TSO system data areas.

USER=

defines the extent of formatting for the TSO user region and the TSO user data areas. The default is USER=PRINT, which causes PRDMP to format both the region and the data areas. User=STORAGE requests only the region, USER-FORMAT requests only the data areas. USER=NO requests no formatting of the user region or data areas.

Depending on the nature of the input dump, the TSO user region and user data areas may or may not be available for formatting. The following summary shows what portions of TSO user areas are available in various kinds of dumps:

AMDSADMP Dumps:

Real storage only -- Paged-in user's region and LSQA

Real storage and all page data sets -- Paged-in user's region and LSQA; paged-out user's LSQA

SVC Dumps:

Virtual storage only -- Paged-in user's regions and LSQA

Virtual storage and dumps of TSO users -- Storage for users as contained in dumps for TSO users

If you want to format paged-out storage associated with the TSO user's regions (AMDSADMP input only), you must use the PRINT PAGE=dn=(sggs) control statement. To find the slot group numbers for paged-out TSO user's regions, consult the TSO user's SPCA data area.

To save processing time, you should specify PRINT TSO= as the last control statement in a series for a given dump.

EDIT Control Statement

The EDIT control statement causes PRDMP to obtain and process trace data created by the Generalized Trace Facility (GTF). Like other control statements, it may be specified either from the operator's console or through cards in the input stream.

```
EDIT       [EXIT=pgmname]

           ⎡,⎧DDNAME⎫=ddname⎤
           ⎣ ⎩ DD   ⎭       ⎦

           [,START=(ddd,hh.mm.ss)]

           [,STOP=(ddd,hh.mm.ss)]

           ⎡,⎧JOBNAME⎫=(jobname1[,jobname2]...[,jobname5])⎤
           ⎣ ⎩J      ⎭                                     ⎦

           [,TCB=(address1[,address2]...[,address5])]

           [,SYS]

           ⎡,⎧IO    ⎫[=(cuu1[,cuu2]...[,cuu50])]⎤
           ⎢ ⎨IO=SIO⎬                           ⎥
           ⎢ ⎨SIO=IO⎬                           ⎥
           ⎣ ⎩SIO   ⎭                           ⎦

           ⎡,⎧SVC                                    ⎫⎤
           ⎣ ⎩SVC=(svcnum1[,svcnum2]...[,svcnum256])⎭⎦

           ⎡,⎧PI                            ⎫⎤
           ⎣ ⎩PI=(code1[,code2]...[,code19])⎭⎦

           [,EXT]

           [,DSP]

           ⎡,USR=⎧ ALL                                                              ⎫⎤
           ⎢     ⎨⎛⎧symbol1  ⎫ ⎡,⎧symbol2  ⎫⎤...⎡,⎧symbol20  ⎫⎤⎞⎬⎥
           ⎢     ⎨⎜⎨idvalue1 ⎬ ⎢ ⎨idvalue2 ⎬⎥   ⎢ ⎨idvalue20 ⎬⎥⎟⎬⎥
           ⎣     ⎩⎝⎩idrange1 ⎭ ⎣ ⎩idrange2 ⎭⎦   ⎣ ⎩idrange20 ⎭⎦⎠⎭⎦
```

Figure PRDMP-3. Format of the EDIT Control Statement, Showing All Valid Keywords.

Edit Parameters

The keywords associated with the EDIT control statement are shown in Figure PRDMP-3; they are described below. All EDIT keyword parameters are optional.

EXIT=pgmname

> defines the program name of a user-written exit routine that will inspect all trace records when PRDMP gives it control. If the routine does not exist or cannot be loaded successfully, EDIT execution will terminate and the next PRDMP control statement will be read.

DDNAME=ddname

> specifies the name of the DD statement that defines the input trace data set.
>
> If you omit this keyword, PRDMP assumes that trace data exists in buffers in a dump of virtual storage, and therefore will not accept any other EDIT keywords except EXIT.

START=(ddd,hh.mm.ss)

STOP=(ddd,hh.mm.ss)

These optional keywords specify that PRDMP is to edit all trace records produced during the time of day indicated. If no START= time is specified, EDIT processing will begin at the beginning of the trace data set. If no STOP= time is specified, EDIT processing will continue to the end of the data set.

JOBNAME=(jobname1[,jobname2]...[,jobname5])

allows you to specify up to five 8-character jobnames for which EDIT will process trace data. If all the jobnames to be specified cannot fit on one line, close the first line with a right parenthesis followed by a comma; on the next line respecify the JOBNAME keyword with the additional jobnames.

This keyword is not valid if SYSM data is to be edited.

TCB=(address1[,address2]...[,address5])

allows you to specify addresses of up to five task control blocks for which EDIT should process trace data. The addresses must be specified as 1- to 6-digit hexadecimal addresses. If all addresses cannot fit on one line, close the first line with a right parenthesis followed by a comma; on the next line respecify the TCB keyword with the additional addresses.

This keyword is not valid if SYSM data is to be edited.

SYS

This optional keyword requests EDIT to process all system event trace records -- that is, SVC, SIO, IO, PI, EXT, and DSP. If no EDIT keyword except DDNAME, EXIT, START, STOP, JOBNAME, and/or TCB is specified, EDIT will assume SYS as the default. (See Figure PRDMP-4).

```
⎧ IO     ⎫ [=(cuu1[,cuu2]...[,cuu50])]
⎪ SIO    ⎪
⎨ IO=SIO ⎬
⎩ SIO=IO ⎭
```

defines up to fifty different devices for which IO trace records (which includes PCI records), SIO trace records, or both should be formatted. If no specific devices are requested, all IO and/or SIO trace records will be formatted. If any specific devices are specified, only trace records associated with those devices will be formatted and all others will be ignored. (See Figure PRDMP-4.)

Devices should be specified as 3-digit device addresses. If all devices to be specified cannot fit on one line, close the first line with a right parenthesis followed by a comma; on the next line respecify the keyword with the remaining addresses.

```
(SVC                                          )
{                                             }
(SVC=(svcnum1[,svcnum2]...,[,svcnum256]))
```

defines up to 256 SVC trace records that EDIT is to format. svcnum is a 1- to 3-digit decimal SVC number.

If no svcnum parameters are specified or if both SVC and SVC= are specified, all SVC trace records will be formatted. If any SVC numbers are specified, only trace records associated with those SVC numbers will be formatted; all others will be ignored. (See Figure PRDMP-4.)

If all SVC numbers cannot fit on one line, close the first line with a right parenthesis followed by a comma; on the next line respecify the keyword with the remaining SVC numbers.

```
(PI                                                        )
{                                                          }
(PI=(code1[,code2]...[,code15][,code17]...[,code19])
```

requests EDIT to format trace records associated with up to eighteen specified program interrupt codes (1-15, 17, 18, 19). If no program interrupt codes are specified or if both PI and PI= are specified, all program interrupt trace records will be formatted. If any program interrupt codes are specified, only those program interrupt trace records will be formatted; all others will be ignored. (See Figure PRDMP-4.)

If all codes to be specified cannot fit on one line, close the first line with a right parenthesis followed by a comma; on the next line respecify the keyword with the remaining codes.

EXT

requests that EDIT format all external interrupt trace records. (See Figure PRDMP-4.)

DSP

requests that EDIT format all dispatcher task-switch trace records. (See Figure PRDMP-4.)

```
USR= ( ALL                                              )
     (                                                  )
     ( DMA1                                             )
     (                                                  )
     {(symbol1 ) [,symbol2 ] [... ,symbol20 ]           }
     ({idvalue1} [,idvalue2] [...,idvalue20 ]           )
     ((idrange1) [,idrange2] [...,idrange20 ]           )
```

specifies which user/subsystem trace records should be formatted; (user or subsystem trace records are created by the GTF GTRACE macro.) If you specified DCB= DIAGNS= TRACE for a data set, you may indicate USR= DMA1 to format the GTF trace record. Or you can specify up to 20 ID values, ranges or symbols representing single components or subsystems. idvalue is a 3-digit hexadecimal ID specified in the GTrace macro when the records to be formatted were created. idrange is a pair of idvalues defining a range of records to be formatted, for example, USR=(010-040,BFD-BFF).

If you want to edit data management trace records, specify USR=DMA. To edit VSAM records, specify USR=AM01, USR=AM02, etc., through USR=AM10.

If ALL is specified alone or in combination with other parameters, all user or subsystem trace entries will be formatted. (See Figure PRDMP-4.)

If all parameters cannot fit on one line, close the first line with a right parenthesis followed by a comma, making sure that any idrange specified is complete; on the next line respecify the USR= keyword and continue with the remaining parameters.

EDIT Parameter Priorities				Trace Events Selected
1	2	3	4	
SYS				All SIO, IO, SVC, PI, DSP, and EXT
	SIO=IO			All SIO and IO
		SIO		All SIO
			SIO=ddd	SIO for device(s) ddd
			SIO=IO=ddd	SIO and IO for device(s) ddd
		IO		All IO
			IO=ddd	IO for device(s) ddd
			IO=SIO=ddd	IO and SIO for device(s) ddd
	SVC			All SVCs
		SVC=num		Specified SVCs
	PI			All PIs
		PI=code		Specified PI code(s)
	DSP			All DSP
	EXT			All EXT
USR=ALL				All USR
	USR=notall			Specified USR

Figure PRDMP-4. Priorities and Effects of EDIT Keywords Used to Select Records by Trace Event Type.

EDIT Parameter Defaults and Priorities

All EDIT defaults depend on the presence or absence of the DDNAME= parameter.

- If the DDNAME= parameter is present, the input is an external trace data set. All parameters are valid. If none except DDNAME= are specified, EDIT assumes a default of SYS.

- If the DDNAME= parameter is absent, the input is a main storage dump containing trace buffers. No parameters except EXIT= are valid, since EDIT cannot select records from a dump. All records, both system and user, will be processed. If you attempt to select specific records, EDIT will prompt you to supply the missing DDNAME= parameter or terminate EDIT processing.

Figure PRDMP-4 summarizes the priority and effect of those EDIT parameters that select records by trace event type. Any keyword shown in the table can be considered to include as subsets all the parameters shown indented below it; for example, SVC=svcnum is a subset of SVC, and SVC is a subset of SYS. Any parameter can override another parameter in the same set that has a lower priority. You should not combine any parameter with another parameter that can override it; for example, do not combine SIO with SIO=ddd. You can, however, combine parameters that are part of separate sets; for example, you can combine SIO=ddd with IO and SVC, or SYS with USR=ALL. You can also combine any parameters that have the same priority; for example, you can combine SIO=aaa with SIO=IO=bbb. In this case the effect will be IO=bbb and SIO=(aaa,bbb).

Note: START=, STOP=, JOBNAME=, and TCB= have no effect on trace event selection; they exercise further selectivity over records already chosen by default or by parameters that select system trace events.

Combining Control Statements

If you are controlling PRDMP operation from the system console, you may want to save time by combining control statements in a single reply to a prompting message. This section describes the rules for combining control statements. Note: These rules also apply if you are invoking PRDMP by JCL and you want to combine control statements on a single card in the input stream.

PRDMP control statements fall into two categories: restricted and free. The names of the categories refer to the way the control statements in them can be combined with each other. The following table shows the categories and the control statements they contain.

FREE	RESTRICTED
CVT=parm	NEWDUMP
NEWTAPE	GO
QCBTRACE	ONGO
LPAMAP	TITLE
FORMAT	END
* EDIT	* EDIT parm
SEGTAB=hhhhhh	PRINT

* Note that the EDIT control statement coded with no parameters falls into the FREE category, while the EDIT control statement with parameters is RESTRICTED.

Here are the rules for combining control statements in the two categories:

- Control statements in the FREE category can be combined freely with each other.
- Control statements in the RESTRICTED category may never be combined with each other.
- Any number of control statements from the FREE category may be combined with one control statement from the RESTRICTED category, provided that the control statement from the RESTRICTED category comes last in the reply.

Here are some examples of control statements combined correctly:

```
LPAMAP,EDIT,PRINT NUCLEUS

FORMAT,QCBTRACE,EDIT DDNAME=TRACE,SVC,SIO=IO=ALL,PI

NEWTAPE,CVT=parm,LPAMAP,GO

NEWTAPE,TITLE

LPAMAP,END

SEGTAB=parm,FORMAT,PRINT ALL
```

PRDMP Storage Requirements

PRDMP requires a virtual storage partition or region size of at least 128K.

PRDMP also requires large amounts of space on peripheral storage devices for the output data set. The following sections describe how to calculate the space needed for the output data set for several different conditions of running PRDMP.

Allocating Space for the Output Data Set

PRDMP output is usually directed to a SYSOUT device; therefore in most cases its output is stored temporarily on a direct access storage device from which it is later written to the printer. This temporary storage allows the user to specify space allocation and blocking factors that will enhance PRDMP's performance.

(Note that if time is not critical and the output data set is very large, the output data set may be allocated directly to a printer. Do this by specifying the UNIT parameter in the PRINTER DD statement, for example UNIT=00E.)

Specifying the Maximum Output Block Size

Since PRDMP uses QSAM as the access method for the SYSOUT data set, you can improve performance by specifying the largest possible block size for the data set. The maximum block size within the limits of the track capacity of the output device can be calculated by the following method: Divide the maximum track capacity in bytes by the output record length, 121 bytes, and ignore any remainder. The quotient is the number of records per block. Multiply this number by 121 to find the maximum block size.

To illustrate: A 2314 disk storage unit has a track capacity of 7294 bytes. The PRDMP output record length is 121 bytes. Thus the number of records per block is 60. This value multiplied by the output record length (121) gives the maximum block size, 7260 bytes. Code this value in the DCB= parameter of the PRINTER DD statement as follows:

```
    DCB=(BLKSIZE=7260)
```

Increasing the Space Allocated to SYSOUT

The amount of space normally allocated to a SYSOUT data set may not be enough to contain the entire formatted dump or trace listing. To prevent this problem, allocate extra direct access storage space for the SYSOUT data set via the SPACE= operand in the PRINTER DD statement that represents the data set. This extra space may be expressed in terms of bytes, tracks, or cylinders.

Use the table below to determine the approximate number of lines that will be printed in a dump listing. (The table does not include figures for the EDIT function of PRDMP.)

STORAGE SIZE	PRINTED LINES
128K	4000
256K	8000
512K	16000
1024K	32000
2048K	64000

Calculating Space Requirements by Block Size

Each printed line is represented by a 121-byte record; the space requirement can therefore be expressed in bytes as the record length multipled by the number of records. As an example, the SPACE= operand for a 512K dump SYSOUT data set might be expressed as: SPACE=(121,(16000,100)).

If a blocking factor was specified for this SYSOUT data set (as discussed above), the space allocation can be expressed in terms of block size. For example, if the block size has been calculated as 7260 bytes (or a blocking factor of 60 records per block), the same 512K dump listing would require 267 blocks to contain all of the listing information. This block figure was calculated as follows:

16000 Output records / 60 Records per block = 267 Blocks

The PRINTER DD statement might then be expressed as:

```
//PRINTER DD SYSOUT=x,
//            SPACE=(7260,(267,10)),
//            UNIT=2314,DCB=(BLKSIZE=7260)
```

Maximum Trace Buffer Size	SYSM Trace	SYSM With User Time Stamp	Comprehensive Trace	Comprehensive Trace With User Time Stamp
1024	25	50	30	60
2048	50	100	60	120
3500	65	130	110	220
4096	100	200	120	240

Figure PRDMP-5. Number of Lines of Edit Output Per Buffer as a Function of Maximum Buffer Size and Trace Type.

Calculating Space Requirements for EDIT Output

When GTF trace data is edited using the EDIT function of PRDMP, you can estimate the number of lines of output provided you know the maximum GTF trace buffer size and the number of blocks to be edited. Figure PRDMP-5 shows the number of lines of EDIT output as a function of maximum buffer size (block size) and the type of trace.

Editing Internal Trace Data

To estimate the number of lines to be printed when GTF buffers are edited from a dump data set, multiply the number of buffers by the number if lines per buffer, as shown in Figure PRDMP-5. The number of buffers is determined by the DCB=(BUFNO=) parameter or the BUF= parameter in the GTF START command; if neither of these was specified the number of buffers defaults to 4 if GTF was started with MODE=INT, and 3 if MODE=EXT.

Editing an External Trace Data Set

To estimate the number of lines to be printed when GTF data is edited from the trace data set on a direct access device, determine the number of blocks per track and multiply that value by the allocated number of tracks; the resulting value is the number of blocks per data set. Multiply that value by the number of lines per block as indicated in Figure PRDMP-5.

For example: A comprehensive trace with user time stamps is to be edited from a data set that occupies 50 tracks of a device whose track capacity is 7200 bytes. The maximum blocksize for the trace (established by the IEFRDER DD statement in the GTF start procedure) is 3500 bytes. Thus the number of blocks per trace (in round figures) is 2, and the number of blocks in the data set is 2(50) or 100. Figure PRDMP-5 indicates that for a comprehensive trace with user time stamps the number of lines per block is 220; thus the expected number of printed lines is 100(220) or 22000.

In this case the PRINTER DD statement might be expressed as:

```
//PRINTER DD SYSOUT=A,SPACE=(121,(22000,100))
```

If the trace data set is on a tape volume, you can estimate the maximum number of lines to be printed by calculating the number of blocks per foot of tape and multiplying by the length of the tape.

Cataloged Procedure

Figure PRDMP-6 shows the cataloged procedure, PRDMP, that IBM supplies for executing PRDMP in VS1. Figure PRDMP-7 shows the cataloged procedure that IBM supplies in VS2. Note that you should not use either of these cataloged procedures to request quick transfer of the SYS1.DUMP data set contents into a data set defined by the SYSUT2 DD statement, since the SYSUT1 and SYSUT2 DD statements may never be used in the same step.

```
//PRDMP        PROC
//DMP          EXEC PGM=HMDPRDMP
//SYSPRINT     DD   SYSOUT=A
//TAPE         DD   DSNAME=SYS1.DUMP,DISP=OLD
//PRINTER      DD   SYSOUT=A
//SYSUT1       DD   UNIT=SYSDA,SPACE=(2056,(770,128))
```

Figure PRDMP-6. The cataloged procedure PRDMP (VS1 ONLY).

```
//PRDMP        PROC
//DMP          EXEC PGM=AMDPRDMP
//SYSPRINT     DD   SYSOUT=A
//TAPE         DD   DSNAME=SYS1.DUMP,DISP=OLD
//PRINTER      DD   SYSOUT=A
//SYSUT1 DD    UNIT=SYSDA,SPACE=(2056,(800,200))
//SYSTSO DD    UNIT=SYSDA,SPACE=(2056,(200,100))
```

Figure PRDMP-7. The cataloged procedure PRDMP (VS2 Only).

The statements for both cataloged procedures are explained below.

EXEC Statement

calls for the execution of HMDPRDMP or AMDPRDMP.

SYSPRINT DD Statement

defines the PRDMP message data set.

TAPE DD Statement

defines the input data set. Unless overridden with other data set names, this statement defines SYS1.DUMP as the input data set.

PRINTER DD Statement

defines the output data set.

SYSUT1 DD Statement

defines the work data set.

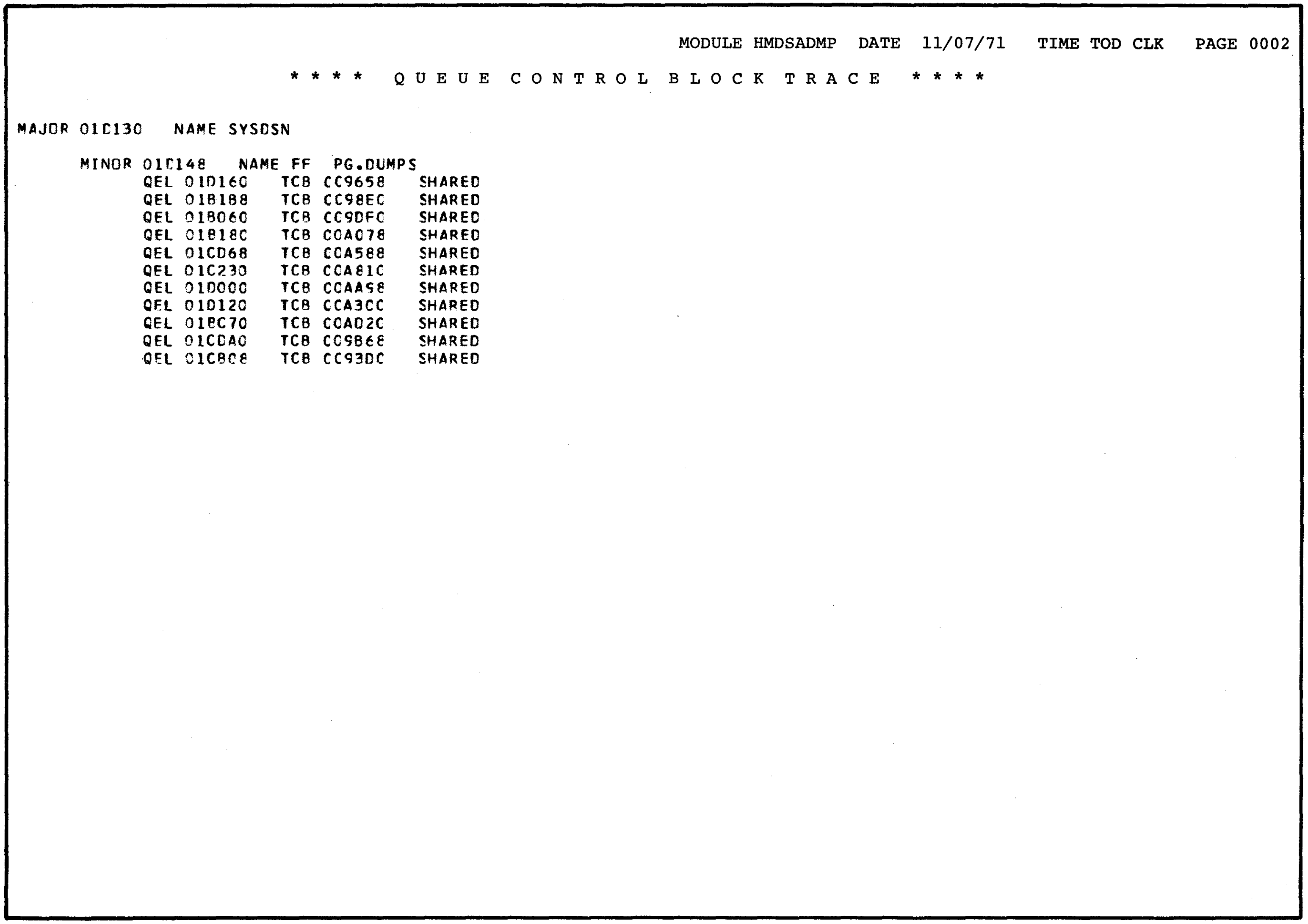
SYSTSO DD Statement (VS2 Only)

defines the work data set for TSO processing if the TSO control statement is included in the input stream.

Note that the SYSIN DD statement has been omitted. Unless this statement is supplied, PRDMP will prompt the operator to enter control statements through the console.

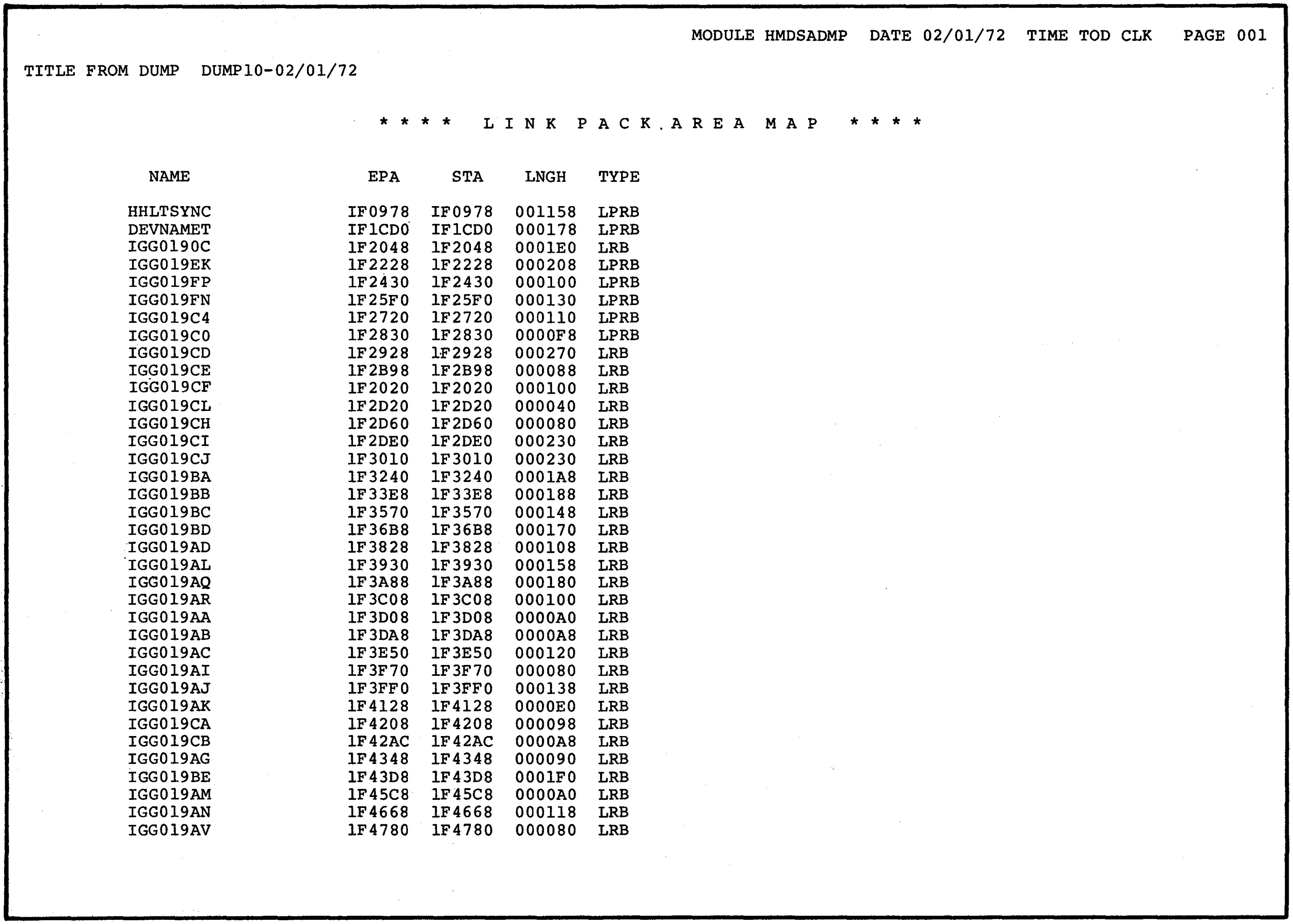
PRDMP Output

Figure PRDMP-8 through PRDMP-12 are samples of PRDMP output for VS1. The formats are explained in detail in the publication OS/VS1 Debugging Guide, GC24-5093. Figures PRDMP-14 through PRDMP-21 are samples of PRDMP output for VS2. The formats for these are explained in detail in the publication OS/VS2 Debugging Guide, GC28-0632.



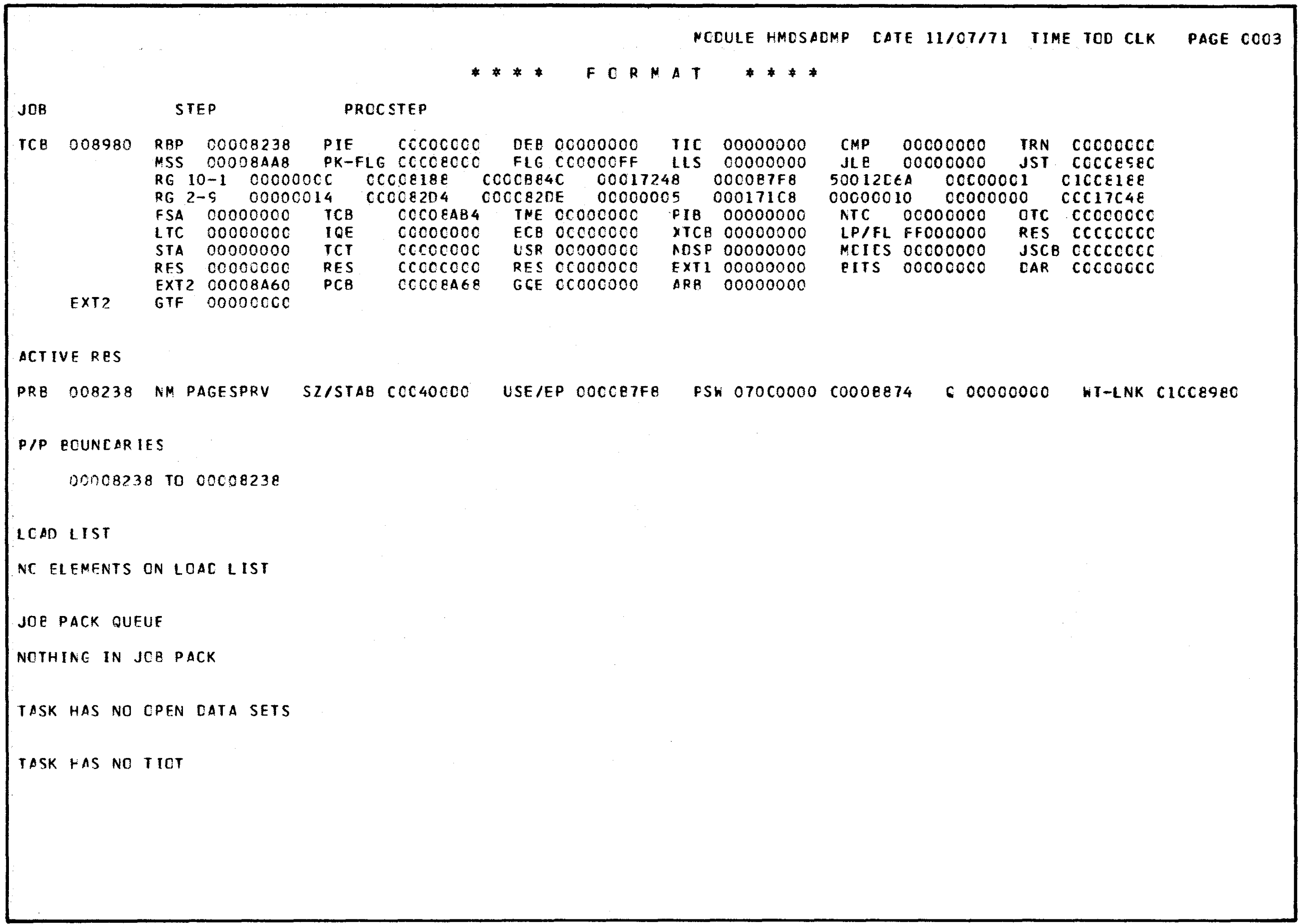
```
                                                              MODULE HMDSADMP  DATE  11/07/71   TIME TOD CLK   PAGE 0002
                          * * * *   Q U E U E   C O N T R O L   B L O C K   T R A C E   * * * *

MAJOR 01C130   NAME SYSDSN

      MINOR 01C148   NAME FF  PG.DUMPS
            QEL 01D160   TCB CC9658   SHARED
            QEL 01B188   TCB CC98EC   SHARED
            QEL 01B060   TCB CC9DFC   SHARED
            QEL 01B18C   TCB CCA078   SHARED
            QEL 01CD68   TCB CCA588   SHARED
            QEL 01C230   TCB CCA81C   SHARED
            QEL 01D000   TCB CCAA98   SHARED
            QEL 01D120   TCB CCA3CC   SHARED
            QEL 01BC70   TCB CCAD2C   SHARED
            QEL 01CDA0   TCB CC9B68   SHARED
            QEL 01C808   TCB CC93DC   SHARED
```

Figure PRDMP-8. Sample Queue Control Block Trace (VS1 Only).



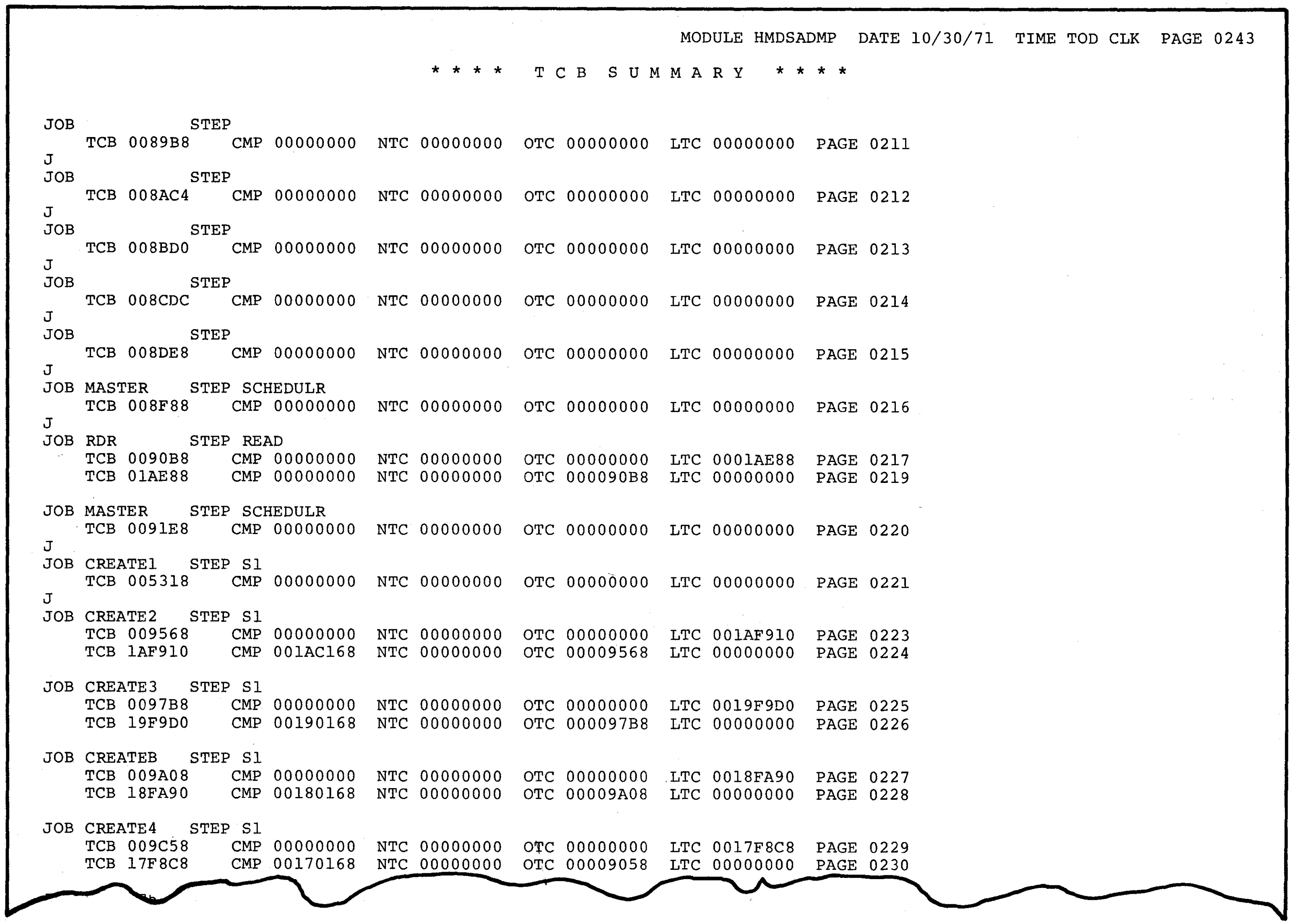
```
                                                       MODULE HMDSADMP  DATE 02/01/72  TIME TOD CLK   PAGE 001
TITLE FROM DUMP  DUMP10-02/01/72

                                  * * * *   L I N K  P A C K . A R E A  M A P   * * * *

          NAME                EPA     STA     LNGH    TYPE

        HHLTSYNC             IF0978  IF0978  001158  LPRB
        DEVNAMET             IF1CD0  IF1CD0  000178  LPRB
        IGG0190C             1F2048  1F2048  0001E0  LRB
        IGG019EK             1F2228  1F2228  000208  LPRB
        IGG019FP             1F2430  1F2430  000100  LPRB
        IGG019FN             1F25F0  1F25F0  000130  LPRB
        IGG019C4             1F2720  1F2720  000110  LPRB
        IGG019C0             1F2830  1F2830  0000F8  LPRB
        IGG019CD             1F2928  1F2928  000270  LRB
        IGG019CE             1F2B98  1F2B98  000088  LRB
        IGG019CF             1F2020  1F2020  000100  LRB
        IGG019CL             1F2D20  1F2D20  000040  LRB
        IGG019CH             1F2D60  1F2D60  000080  LRB
        IGG019CI             1F2DE0  1F2DE0  000230  LRB
        IGG019CJ             1F3010  1F3010  000230  LRB
        IGG019BA             1F3240  1F3240  0001A8  LRB
        IGG019BB             1F33E8  1F33E8  000188  LRB
        IGG019BC             1F3570  1F3570  000148  LRB
        IGG019BD             1F36B8  1F36B8  000170  LRB
        IGG019AD             1F3828  1F3828  000108  LRB
        IGG019AL             1F3930  1F3930  000158  LRB
        IGG019AQ             1F3A88  1F3A88  000180  LRB
        IGG019AR             1F3C08  1F3C08  000100  LRB
        IGG019AA             1F3D08  1F3D08  0000A0  LRB
        IGG019AB             1F3DA8  1F3DA8  0000A8  LRB
        IGG019AC             1F3E50  1F3E50  000120  LRB
        IGG019AI             1F3F70  1F3F70  000080  LRB
        IGG019AJ             1F3FF0  1F3FF0  000138  LRB
        IGG019AK             1F4128  1F4128  0000E0  LRB
        IGG019CA             1F4208  1F4208  000098  LRB
        IGG019CB             1F42AC  1F42AC  0000A8  LRB
        IGG019AG             1F4348  1F4348  000090  LRB
        IGG019BE             1F43D8  1F43D8  0001F0  LRB
        IGG019AM             1F45C8  1F45C8  0000A0  LRB
        IGG019AN             1F4668  1F4668  000118  LRB
        IGG019AV             1F4780  1F4780  000080  LRB
```

Figure PRDMP-9. Sample Link Pack Area Map (VS1 Only).



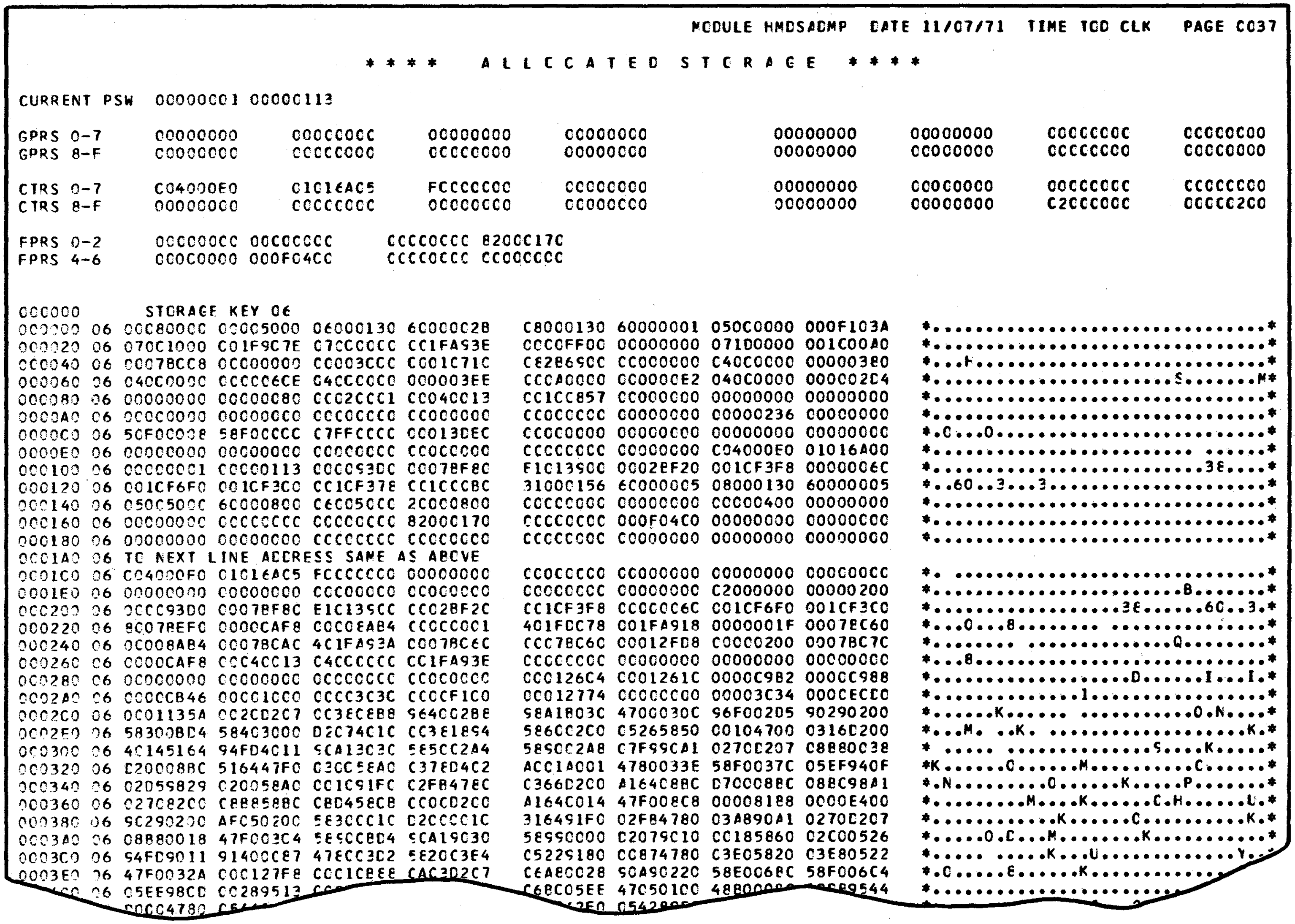
```
                                                                                MCDULE HMDSADMP  CATE 11/07/71  TIME TOD CLK   PAGE C003

                                                     * * * *    F C R M A T    * * * *

JOB                  STEP                 PROCSTEP

TCB   008980    RBP   00008238     PIE      CCCOCCCC    DEB 00000000    TIC   00000000    CMP    00000000    TRN   CCCOCCCC
                MSS   00008AA8     PK-FLG CCCC8CCC      FLG CC0000FF    LLS   00000000    JLB    00000000    JST   CCCC898C
                RG  10-1   0000000CC     CCCC8188     CCCCB84C     00017248     0000B7F8     50012D6A     00C00001     C1CCE188
                RG  2-9    00000014      CCC082D4     COCC82DE     00000005     000171C8     00000010     CC000000     CCC17C48
                FSA   00000000     TCB      CCCO8AB4    TME 0C000C0C    PIB   00000000    NTC    00000000    OTC   CC0C0CCC
                LTC   0000000C     IQE      CCCCC0CC    ECB 0CC0C0C0    XTCB  00000000    LP/FL  FF000000    RES   CCCCCCCC
                STA   00000000     TCT      CCCCCOCC    USR 0000000C    NDSP  00000000    MCICS  00000000    JSCB  CCCCCCCC
                RES   0000C0CC     RES      CCCCCOCC    RES CC0000C0    EXT1  00000000    BITS   00C0C0C0    DAR   CCCC0GCC
                EXT2  00008A60     PCB      CCCC8A68    GCE CC000000    ARB   00000000
      EXT2      GTF   000000CC


ACTIVE RBS

PRB   008238    NM PAGESPRV    SZ/STAB CCC40CDO    USE/EP 000CB7F8    PSW 070C0000 C0008874    C 00000000    WT-LNK C1CC898C


P/P BOUNCARIES

      00008238 TO 00C08238


LCAD LIST

NC ELEMENTS ON LOAC LIST


JOB PACK QUEUE

NOTHING IN JCB PACK


TASK HAS NO CPEN CATA SETS


TASK HAS NO TICT
```

Figure PRDMP-10. Sample Formatted Data Areas (VS1 Only).



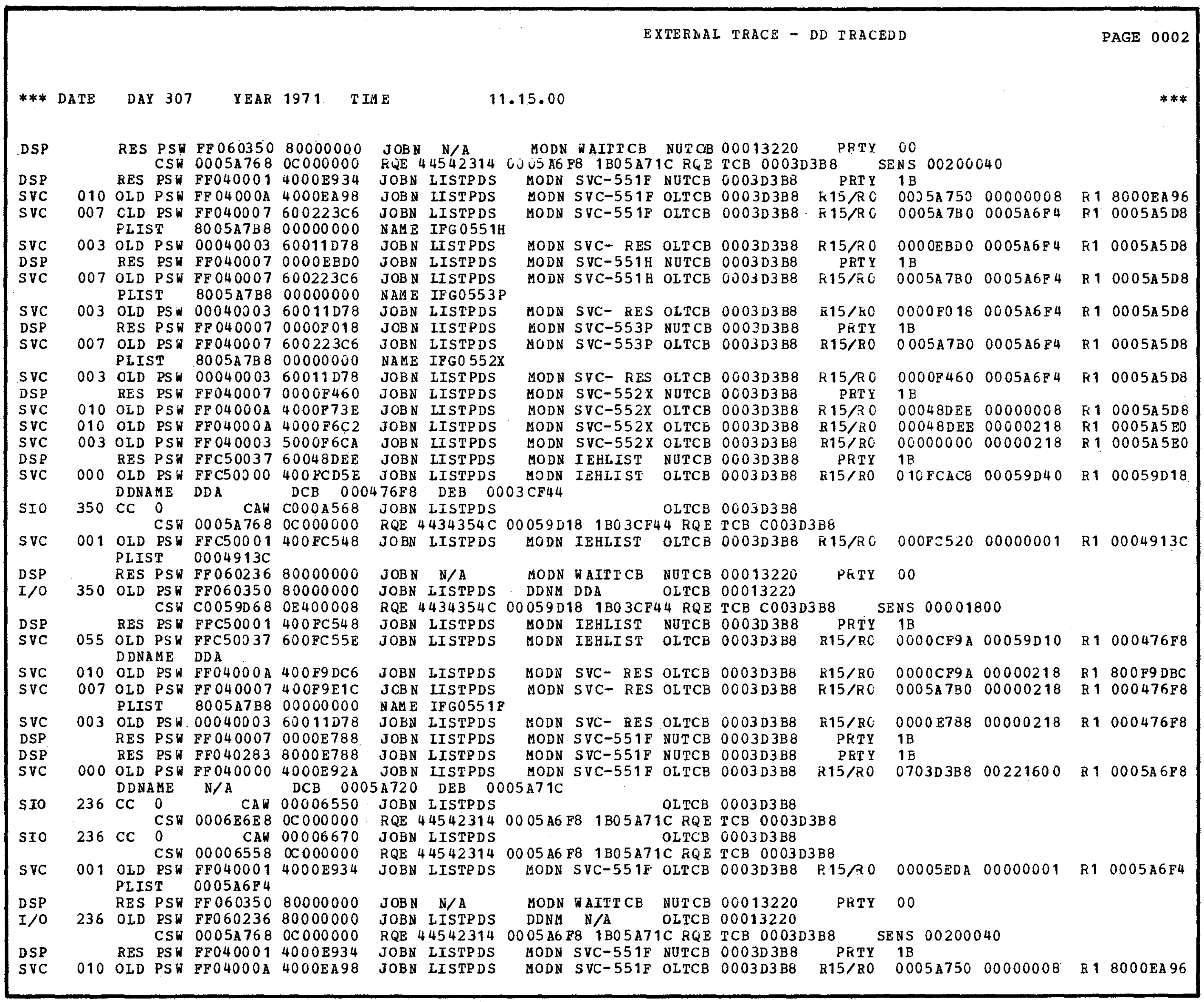
```
                                                  MODULE HMDSADMP  DATE 10/30/71  TIME TOD CLK  PAGE 0243
                              * * * *   T C B  S U M M A R Y   * * * *


JOB           STEP
    TCB 0089B8    CMP 00000000  NTC 00000000  OTC 00000000  LTC 00000000  PAGE 0211
J
JOB           STEP
    TCB 008AC4    CMP 00000000  NTC 00000000  OTC 00000000  LTC 00000000  PAGE 0212
J
JOB           STEP
    TCB 008BD0    CMP 00000000  NTC 00000000  OTC 00000000  LTC 00000000  PAGE 0213
J
JOB           STEP
    TCB 008CDC    CMP 00000000  NTC 00000000  OTC 00000000  LTC 00000000  PAGE 0214
J
JOB           STEP
    TCB 008DE8    CMP 00000000  NTC 00000000  OTC 00000000  LTC 00000000  PAGE 0215
J
JOB MASTER    STEP SCHEDULR
    TCB 008F88    CMP 00000000  NTC 00000000  OTC 00000000  LTC 00000000  PAGE 0216
J
JOB RDR       STEP READ
    TCB 0090B8    CMP 00000000  NTC 00000000  OTC 00000000  LTC 0001AE88  PAGE 0217
    TCB 01AE88    CMP 00000000  NTC 00000000  OTC 000090B8  LTC 00000000  PAGE 0219

JOB MASTER    STEP SCHEDULR
    TCB 0091E8    CMP 00000000  NTC 00000000  OTC 00000000  LTC 00000000  PAGE 0220
J
JOB CREATE1   STEP S1
    TCB 005318    CMP 00000000  NTC 00000000  OTC 00000000  LTC 00000000  PAGE 0221
J
JOB CREATE2   STEP S1
    TCB 009568    CMP 00000000  NTC 00000000  OTC 00000000  LTC 001AF910  PAGE 0223
    TCB 1AF910    CMP 001AC168  NTC 00000000  OTC 00009568  LTC 00000000  PAGE 0224

JOB CREATE3   STEP S1
    TCB 0097B8    CMP 00000000  NTC 00000000  OTC 00000000  LTC 0019F9D0  PAGE 0225
    TCB 19F9D0    CMP 00190168  NTC 00000000  OTC 000097B8  LTC 00000000  PAGE 0226

JOB CREATEB   STEP S1
    TCB 009A08    CMP 00000000  NTC 00000000  OTC 00000000  LTC 0018FA90  PAGE 0227
    TCB 18FA90    CMP 00180168  NTC 00000000  OTC 00009A08  LTC 00000000  PAGE 0228

JOB CREATE4   STEP S1
    TCB 009C58    CMP 00000000  NTC 00000000  OTC 00000000  LTC 0017F8C8  PAGE 0229
    TCB 17F8C8    CMP 00170168  NTC 00000000  OTC 00009058  LTC 00000000  PAGE 0230
```

Figure PRDMP-11. Sample TCB Summary (VS1 Only).



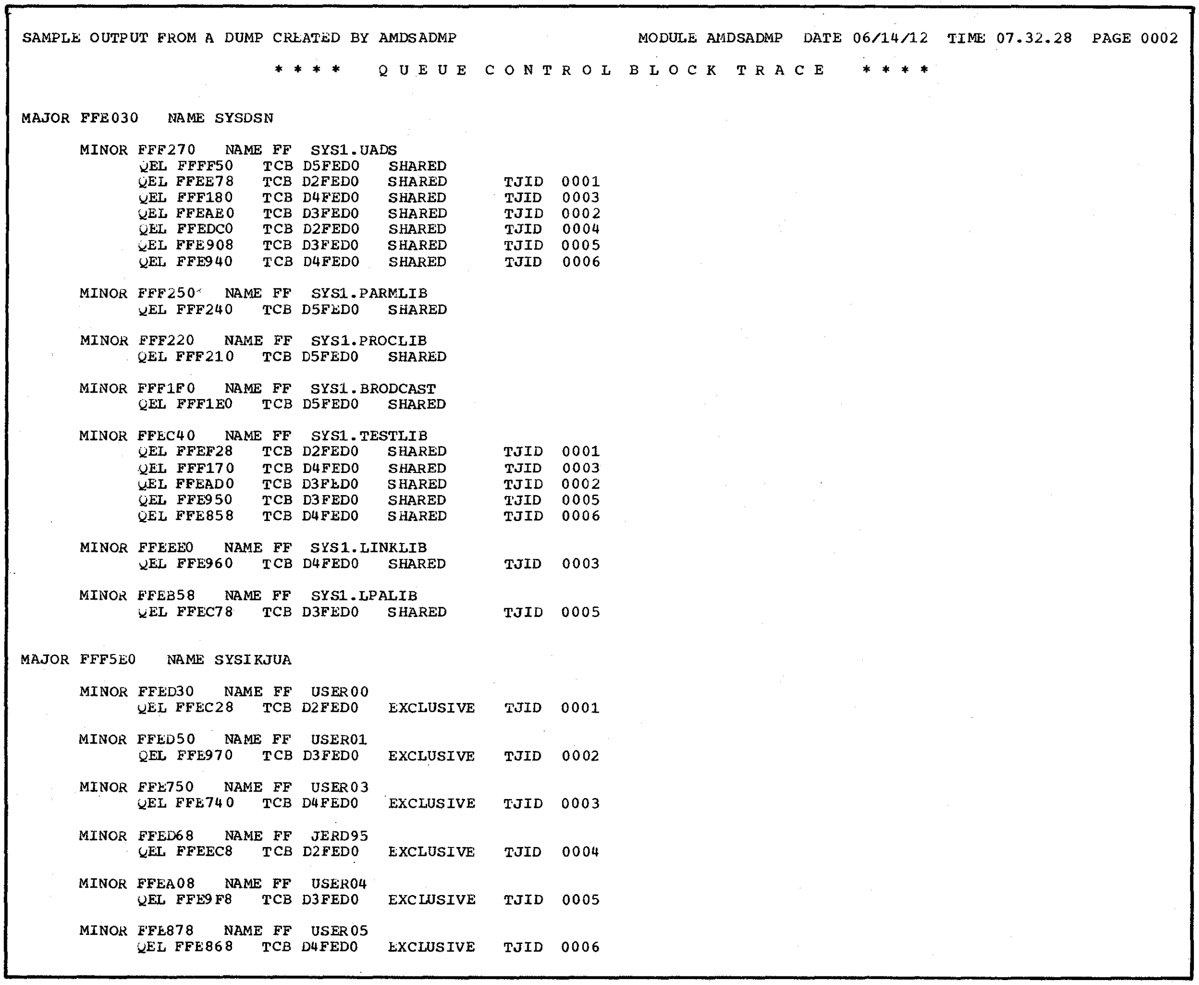
```
                                                                   MCDULE HMDSADMP  CATE 11/07/71  TIME TCD CLK   PAGE C037

                                        * * * *   A L L C C A T E D   S T C R A G E   * * * *

CURRENT PSW   00000CC1 00000113

GPRS 0-7      00000000      000CC0CC      00000000      CC0000C0            00000000      00000000      C0CCCC0C      CCCC0C00
GPRS 8-F      C000000C      CCCCC00C      0CCCC000      00000CC0            00000000      0C000000      CCCCCCC0      C0CC0000

CTRS 0-7      C04000E0      C1C16AC5      FCCCCCCC      CCCCC000            00000000      C0C00000      00CCCCCC      CCCCCCC0
CTRS 8-F      00000CC0      CCCCCCCC      0CCCCCC0      CC0000C0            00000000      00000000      C2CCC0CC      0CCCC2C0

FPRS 0-2      0CC000CC 00CC0CCC      CCCC0CCC 8200C17C
FPRS 4-6      C0C00000 000FC4CC      CCCC0CCC CC00CCCC


CCC000       STCRAGE KEY 06
000000 06 00C8000C 0C0C5000 06000130 6C000C28   C8000130 60000001 050C0000 000F103A   *................................*
000020 06 070C1000 C01F9C7E C7CCCCCC CC1FA93E   CCC0FF00 00000000 071D0000 001C00A0   *................................*
CC0040 06 0007BCC8 0C000000 CC003CCC C001C71C   CE2B69CC CC000000 C40C0C00 00000380   *...F............................*
000060 06 040C0000 CCCCC6CE G4CCCCC0 000003EE   CCCA0000 C00000E2 040C0000 000002D4   *........................S......M*
000080 06 00000000 00C00080 CCC2CCC1 CC040C13   CC1CC857 CC000C00 00000000 00000000   *................................*
0000A0 06 0C0C0000 00000CC0 CCCCCCC0 CC000000   CC0CCCCC 00000000 00000236 00000000   *................................*
0000C0 06 50F0C008 58F0CCCC C7FFCCCC CC013DEC   CC0C0000 00000C00 00000000 0000000C   *.0...0..........................*
0000E0 06 00000000 0000000C CCCCCCCC CC0CC000   CCCCCCCC C0000C00 C04000E0 01016A00   *........................ ......*
000100 06 CCCCCCC1 CCCC0113 CCCC93DC CC07BF8C   F1C1390C 0002BF20 001CF3F8 0000006C   *..........................3E....*
000120 06 C01CF6F0 C01CF3C0 CC1CF378 CC1CCCBC   31000156 6C000005 08000130 60000005   *..60..3...3.....................*
000140 06 050C500C 6C000800 C6C050CC 2C000800   CCCCC0CC C0000C00 CCC00400 00000000   *................................*
000160 06 0000000C CCCCCCCC CCCCCCCC 82000170   CCCCCCCC 000F04C0 00000000 00000C0C   *................................*
000180 06 00000000 00000000 CCCCCCCC CCCCCCC0   CCCCCC0C C0000000 00000000 00000000   *................................*
0C01A0 06 TC NEXT LINE ACCRESS SAME AS ABCVE
0C01C0 06 C04000F0 C1C16AC5 FCCCCCCC 0000000C   CC0CCCC0 CC000000 00000000 000C00CC   *. ..............................*
0001E0 06 00000000 C0000000 CCC000C0 CC0C0CC0   C0CCCCCC C000000C C2000000 00000200   *.........................B......*
0CC200 06 0CCC93D0 C007BF8C E1C139CC CCC2BF2C   CC1CF3F8 CCCCCC6C C01CF6F0 001CF3C0   *..................3E......6C..3.*
000220 06 8C07BEFC 0000CAF8 CCC0EAB4 CCCCC0C1   401FDC78 001FA918 0000001F 0007EC60   *...0...8........ ...............*
000240 06 0C008AB4 C007BCAC 4C1FA93A C007BC6C   CCC7BC6C C0012FD8 C000C200 0007BC7C   *........ ...............Q.......*
000260 06 C000CAF8 CCC4CC13 C4CCCCCC CC1FA93E   CCCCCCCC C0000000 00000000 00C000CC   *...8............................*
0C0280 06 0C000000 C000000C 0CCCCCCC CC0C0CCC   CCC126C4 C001261C 0000C982 000CC988   *...................D......I...I.*
0C02A0 06 C0CCCB46 00CC1CC0 CCCC3C3C CCCCF1C0   0C012774 CCCCCC00 00003C34 000CECD0   *...............1................*
0002C0 06 0C01135A CC2CD2C7 CC3EC8B8 964CC2B8   98A1803C 4700030C 96F00205 90290200   *......K...... ...........0.N....*
0C02E0 06 58300BD4 584C3000 D2C74C1C CC3E1894   586CC2C0 C5265850 00104700 0316D200   *...M. ..K. ...................K.*
0C0300 06 4C145164 94FD4C11 9CA13C3C 5E5CC2A4   589CC2A8 C7F99CA1 027CD207 C8B80C38   * ..... ..............S....K.....*
0C0320 06 C20008BC 516447F0 C3CC5EA0 C37ED4C2   ACC1A001 4780033E 58F0037C 05EF940F   *K......0......M..........C......*
0C0340 06 02D59829 C20058A0 CC1C91FC C2FB47EC   C366D2C0 A164C8BC D7CC08BC 08BC98A1   *.N.........0......K.....P.......*
000360 06 C27C82CC C8B858BC CBD458CB CC0CD2C0   A164C014 47F008C8 00008188 0000E400   *.........M....K......C.H......U.*
000380 06 9C29020C AFC5020C 5E3CCC1C D2CCCC1C   316491F0 02FB4780 03A890A1 0270D207   *.............K......C.........K.*
0C03A0 06 08B80018 47F003C4 5E9CCBD4 9CA19C30   58990000 D2079C10 CC185860 02C00526   *.....0.C...M.........K..........*
0003C0 06 94FD9011 91400C87 47ECC3D2 5E20C3E4   C5229180 CC874780 C3E05820 C3E80522   *..... .....K...U..............Y..
0003E0 06 47F0032A C0C127F8 CCC1CBEE CAC3D2C7   C6A8C028 9CA9022C 58E006BC 58F006C4   *.0.....E......K..........
   4CC 06 05EE98CD CC289513 CC                  C68C05EE 47C501C0 488000         544   *.........................
       0CC4780 C5                                  2E0 05428
```

Figure PRDMP-12. Sample Dump -- General Format (VS1 Only).



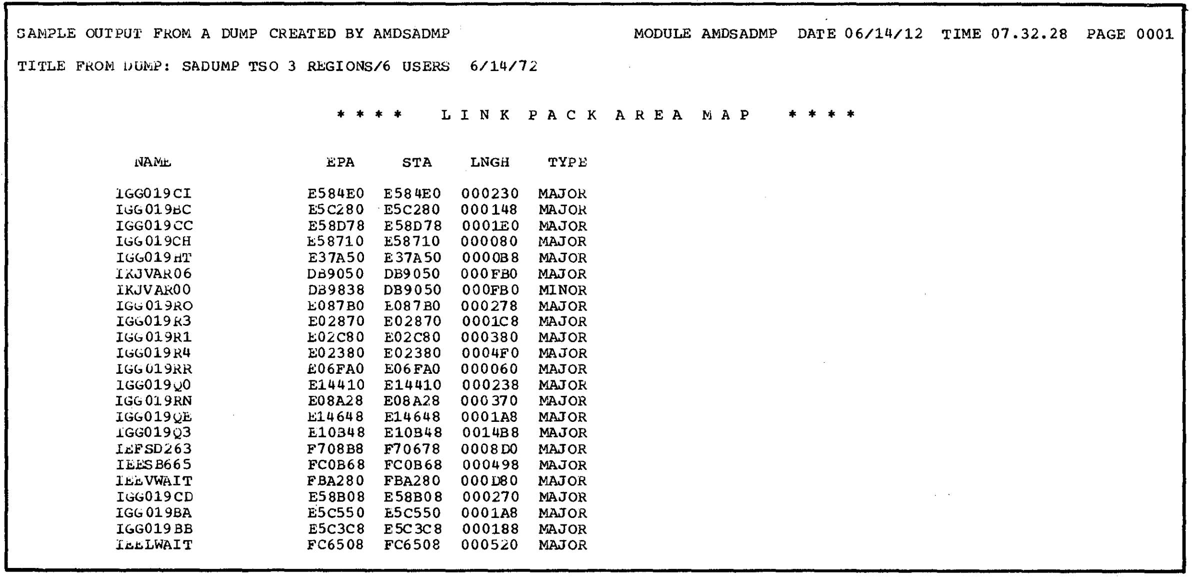
```
                                                     EXTERNAL TRACE - DD TRACEDD                         PAGE 0002


*** DATE    DAY 307    YEAR 1971   TIME         11.15.00                                                      ***

DSP          RES PSW FF060350 80000000  JOBN  N/A      MODN WAITTCB  NUTCB 00013220     PRTY  00
                 CSW 0005A768 0C000000  RQE 44542314 0005A6F8 1B05A71C RQE TCB 0003D3B8    SENS 00200040
DSP          RES PSW FF040001 4000E934  JOBN LISTPDS   MODN SVC-551F NUTCB 0003D3B8     PRTY  1B
SVC     010 OLD PSW FF04000A 4000EA98  JOBN LISTPDS   MODN SVC-551F OLTCB 0003D3B8  R15/R0   0005A750 00000008  R1 8000EA96
SVC     007 CLD PSW FF040007 600223C6  JOBN LISTPDS   MODN SVC-551F OLTCB 0003D3B8  R15/R0   0005A7B0 0005A6F4  R1 0005A5D8
             PLIST    8005A7B8 00000000  NAME IFG0551H
SVC     003 OLD PSW 00040003 60011D78  JOBN LISTPDS   MODN SVC- RES OLTCB 0003D3B8  R15/R0   0000EBD0 0005A6F4  R1 0005A5D8
DSP          RES PSW FF040007 0000EBD0  JOBN LISTPDS   MODN SVC-551H NUTCB 0003D3B8     PRTY  1B
SVC     007 OLD PSW FF040007 600223C6  JOBN LISTPDS   MODN SVC-551H OLTCB 0003D3B8  R15/R0   0005A7B0 0005A6F4  R1 0005A5D8
             PLIST    8005A7B8 00000000  NAME IFG0553P
SVC     003 OLD PSW 00040003 60011D78  JOBN LISTPDS   MODN SVC- RES OLTCB 0003D3B8  R15/R0   0000F018 0005A6F4  R1 0005A5D8
DSP          RES PSW FF040007 0000F018  JOBN LISTPDS   MODN SVC-553P NUTCB 0003D3B8     PRTY  1B
SVC     007 OLD PSW FF040007 600223C6  JOBN LISTPDS   MODN SVC-553P OLTCB 0003D3B8  R15/R0   0005A7B0 0005A6F4  R1 0005A5D8
             PLIST    8005A7B8 00000000  NAME IFG0552X
SVC     003 OLD PSW 00040003 60011D78  JOBN LISTPDS   MODN SVC- RES OLTCB 0003D3B8  R15/R0   0000F460 0005A6F4  R1 0005A5D8
DSP          RES PSW FF040007 0000F460  JOBN LISTPDS   MODN SVC-552X NUTCB 0003D3B8     PRTY  1B
SVC     010 OLD PSW FF04000A 4000F73E  JOBN LISTPDS   MODN SVC-552X OLTCB 0003D3B8  R15/R0   00048DEE 00000008  R1 0005A5D8
SVC     010 OLD PSW FF04000A 4000F6C2  JOBN LISTPDS   MODN SVC-552X OLTCB 0003D3B8  R15/R0   00048DEE 00000218  R1 0005A5E0
SVC     003 OLD PSW FF040003 5000F6CA  JOBN LISTPDS   MODN SVC-552X OLTCB 0003D3B8  R15/R0   00000000 00000218  R1 0005A5E0
DSP          RES PSW FFC50037 60048DEE  JOBN LISTPDS   MODN IEHLIST  NUTCB 0003D3B8     PRTY  1B
SVC     000 OLD PSW FFC50000 400FCD5E  JOBN LISTPDS   MODN IEHLIST  OLTCB 0003D3B8  R15/R0   010FCAC8 00059D40  R1 00059D18
             DDNAME   DDA       DCB  000476F8  DEB  0003CF44
SIO     350 CC   0        CAW C000A568  JOBN LISTPDS                  OLTCB 0003D3B8
                 CSW 0005A768 0C000000  RQE 4434354C 00059D18 1B03CF44 RQE TCB C003D3B8
SVC     001 OLD PSW FFC50001 400FC548  JOBN LISTPDS   MODN IEHLIST  OLTCB 0003D3B8  R15/R0   000FC520 00000001  R1 0004913C
             PLIST    0004913C
DSP          RES PSW FF060236 80000000  JOBN  N/A      MODN WAITTCB  NUTCB 00013220     PRTY  00
I/O     350 OLD PSW FF060350 80000000  JOBN LISTPDS   DDNM DDA      OLTCB 00013220
                 CSW C0059D68 0E400008  RQE 4434354C 00059D18 1B03CF44 RQE TCB C003D3B8    SENS 00001800
DSP          RES PSW FFC50001 400FC548  JOBN LISTPDS   MODN IEHLIST  NUTCB 0003D3B8     PRTY  1B
SVC     055 OLD PSW FFC50037 600FC55E  JOBN LISTPDS   MODN IEHLIST  OLTCB 0003D3B8  R15/R0   0000CF9A 00059D10  R1 000476F8
             DDNAME   DDA
SVC     010 OLD PSW FF04000A 400F9DC6  JOBN LISTPDS   MODN SVC- RES OLTCB 0003D3B8  R15/R0   0000CF9A 00000218  R1 800F9DBC
SVC     007 OLD PSW FF040007 400F9E1C  JCBN LISTPDS   MODN SVC- RES OLTCB 0003D3B8  R15/R0   0005A7B0 00000218  R1 000476F8
             PLIST    8005A7B8 00000000  NAME IFG0551F
SVC     003 OLD PSW 00040003 60011D78  JOBN LISTPDS   MODN SVC- RES OLTCB 0003D3B8  R15/R0   0000E788 00000218  R1 000476F8
DSP          RES PSW FF040007 0000E788  JOBN LISTPDS   MODN SVC-551F NUTCB 0003D3B8     PRTY  1B
DSP          RES PSW FF040283 8000E788  JOBN LISTPDS   MODN SVC-551F NUTCB 0003D3B8     PRTY  1B
SVC     000 OLD PSW FF040000 4000E92A  JOBN LISTPDS   MODN SVC-551F OLTCB 0003D3B8  R15/R0   0703D3B8 00221600  R1 0005A6F8
             DDNAME    N/A      DCB  0005A720  DEB  0005A71C
SIO     236 CC   0        CAW 00006550  JOBN LISTPDS                  OLTCB 0003D3B8
                 CSW 0006E6E8 0C000000  RQE 44542314 0005A6F8 1B05A71C RQE TCB 0003D3B8
SIO     236 CC   0        CAW 00006670  JOBN LISTPDS                  OLTCB 0003D3B8
                 CSW 00006558 0C000000  RQE 44542314 0005A6F8 1B05A71C RQE TCB 0003D3B8
SVC     001 OLD PSW FF040001 4000E934  JOBN LISTPDS   MODN SVC-551F OLTCB 0003D3B8  R15/R0   00005EDA 00000001  R1 0005A6F4
             PLIST    0005A6F4
DSP          RES PSW FF060350 80000000  JOBN  N/A      MODN WAITTCB  NUTCB 00013220     PRTY  00
I/O     236 OLD PSW FF060236 80000000  JOBN LISTPDS   DDNM  N/A     OLTCB 00013220
                 CSW 0005A768 0C000000  RQE 44542314 0005A6F8 1B05A71C RQE TCB 0003D3B8    SENS 00200040
DSP          RES PSW FF040001 4000E934  JOBN LISTPDS   MODN SVC-551F NUTCB 0003D3B8     PRTY  1B
SVC     010 OLD PSW FF04000A 4000EA98  JOBN LISTPDS   MODN SVC-551F OLTCB 0003D3B8  R15/R0   0005A750 00000008  R1 8000EA96
```

Figure PRDMP-13. Sample EDIT for Trace Data Set.



```
SAMPLE OUTPUT FROM A DUMP CREATED BY AMDSADMP                MODULE AMDSADMP  DATE 06/14/12  TIME 07.32.28  PAGE 0002
                                   ****   QUEUE CONTROL BLOCK TRACE   ****

MAJOR FFE030   NAME SYSDSN

      MINOR FFF270    NAME FF  SYS1.UADS
            QEL FFFF50    TCB D5FED0   SHARED
            QEL FFEE78    TCB D2FED0   SHARED      TJID  0001
            QEL FFF180    TCB D4FED0   SHARED      TJID  0003
            QEL FFEAE0    TCB D3FED0   SHARED      TJID  0002
            QEL FFEDC0    TCB D2FED0   SHARED      TJID  0004
            QEL FFE908    TCB D3FED0   SHARED      TJID  0005
            QEL FFE940    TCB D4FED0   SHARED      TJID  0006

      MINOR FFF250    NAME FF  SYS1.PARMLIB
            QEL FFF240    TCB D5FED0   SHARED

      MINOR FFF220    NAME FF  SYS1.PROCLIB
            QEL FFF210    TCB D5FED0   SHARED

      MINOR FFF1F0    NAME FF  SYS1.BRODCAST
            QEL FFF1E0    TCB D5FED0   SHARED

      MINOR FFEC40    NAME FF  SYS1.TESTLIB
            QEL FFEF28    TCB D2FED0   SHARED      TJID  0001
            QEL FFF170    TCB D4FED0   SHARED      TJID  0003
            QEL FFEAD0    TCB D3FED0   SHARED      TJID  0002
            QEL FFE950    TCB D3FED0   SHARED      TJID  0005
            QEL FFE858    TCB D4FED0   SHARED      TJID  0006

      MINOR FFEEE0    NAME FF  SYS1.LINKLIB
            QEL FFE960    TCB D4FED0   SHARED      TJID  0003

      MINOR FFEB58    NAME FF  SYS1.LPALIB
            QEL FFEC78    TCB D3FED0   SHARED      TJID  0005

MAJOR FFF5E0   NAME SYSIKJUA

      MINOR FFED30    NAME FF  USER00
            QEL FFEC28    TCB D2FED0   EXCLUSIVE   TJID  0001

      MINOR FFED50    NAME FF  USER01
            QEL FFE970    TCB D3FED0   EXCLUSIVE   TJID  0002

      MINOR FFE750    NAME FF  USER03
            QEL FFE740    TCB D4FED0   EXCLUSIVE   TJID  0003

      MINOR FFED68    NAME FF  JERD95
            QEL FFEEC8    TCB D2FED0   EXCLUSIVE   TJID  0004

      MINOR FFEA08    NAME FF  USER04
            QEL FFE9F8    TCB D3FED0   EXCLUSIVE   TJID  0005

      MINOR FFE878    NAME FF  USER05
            QEL FFE868    TCB D4FED0   EXCLUSIVE   TJID  0006
```

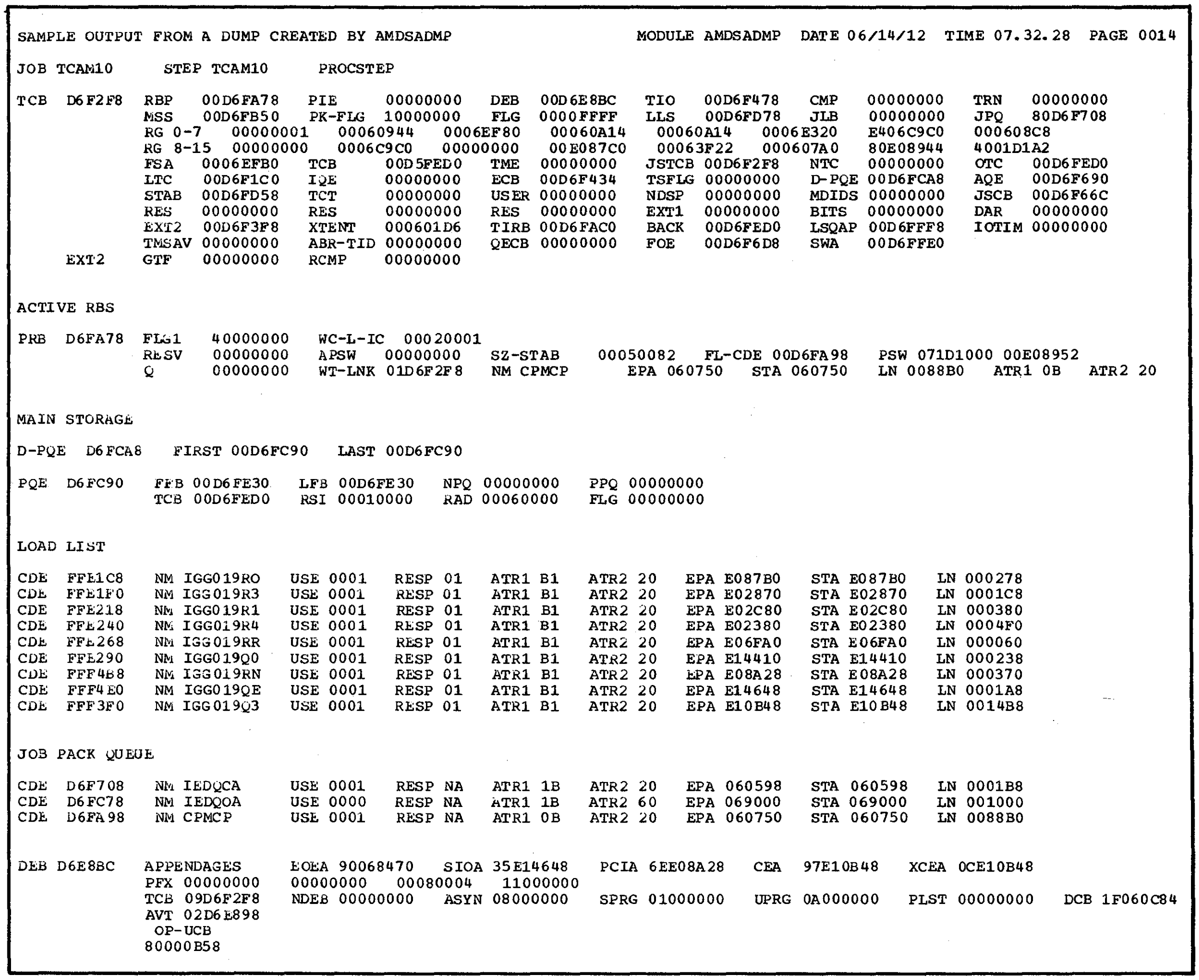
Figure PRDMP-14. Sample Queue Control Block Trace (VS2 Only).



```
SAMPLE OUTPUT FROM A DUMP CREATED BY AMDSADMP              MODULE AMDSADMP  DATE 06/14/12  TIME 07.32.28  PAGE 0001

TITLE FROM DUMP: SADUMP TSO 3 REGIONS/6 USERS  6/14/72

                                     * * * *   L I N K  P A C K  A R E A  M A P   * * * *

           NAME                 EPA     STA     LNGH    TYPE

         IGG019CI             E584E0  E584E0  000230  MAJOR
         IGG019BC             E5C280  E5C280  000148  MAJOR
         IGG019CC             E58D78  E58D78  0001E0  MAJOR
         IGG019CH             E58710  E58710  000080  MAJOR
         IGG019HT             E37A50  E37A50  0000B8  MAJOR
         IKJVAR06             DB9050  DB9050  000FB0  MAJOR
         IKJVAR00             DB9838  DB9050  000FB0  MINOR
         IGG019RO             E087B0  E087B0  000278  MAJOR
         IGG019R3             E02870  E02870  0001C8  MAJOR
         IGG019R1             E02C80  E02C80  000380  MAJOR
         IGG019R4             E02380  E02380  0004F0  MAJOR
         IGG019RR             E06FA0  E06FA0  000060  MAJOR
         IGG019Q0             E14410  E14410  000238  MAJOR
         IGG019RN             E08A28  E08A28  000370  MAJOR
         IGG019QE             E14648  E14648  0001A8  MAJOR
         IGG019Q3             E10B48  E10B48  0014B8  MAJOR
         IEFSD263             F708B8  F70678  0008D0  MAJOR
         IEESB665             FC0B68  FC0B68  000498  MAJOR
         IEEVWAIT             FBA280  FBA280  000D80  MAJOR
         IGG019CD             E58B08  E58B08  000270  MAJOR
         IGG019BA             E5C550  E5C550  0001A8  MAJOR
         IGG019BB             E5C3C8  E5C3C8  000188  MAJOR
         IEELWAIT             FC6508  FC6508  000520  MAJOR
```

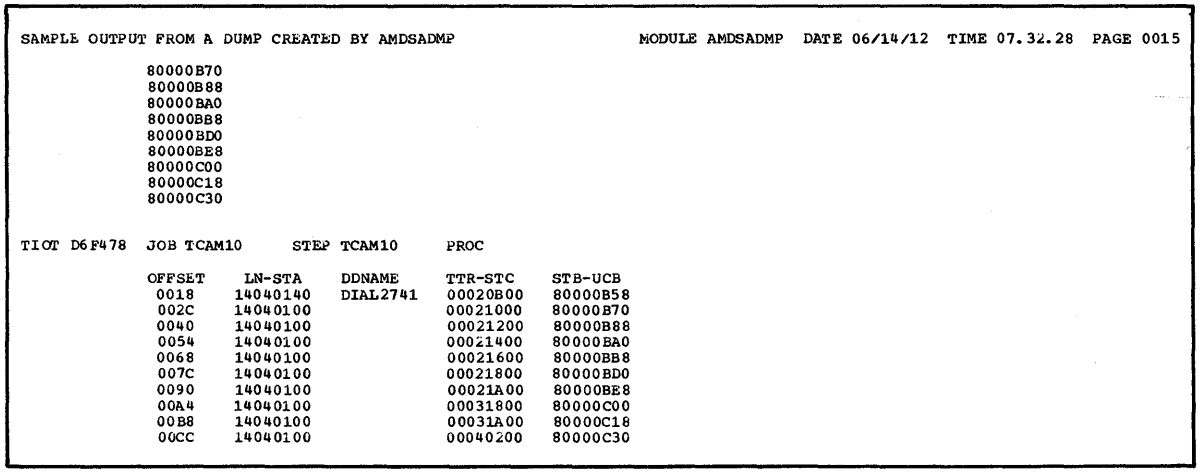
Figure PRDMP-15. Sample Link Pack Area Map (VS2 Only).

PRDM



```
SAMPLE OUTPUT FROM A DUMP CREATED BY AMDSADMP                MODULE AMDSADMP  DATE 06/14/12  TIME 07.32.28  PAGE 0014

JOB TCAM10      STEP TCAM10      PROCSTEP

TCB  D6F2F8  RBP   00D6FA78   PIE      00000000   DEB  00D6E8BC   TIO   00D6F478   CMP   00000000   TRN   00000000
             MSS   00D6FB50   PK-FLG   10000000   FLG  0000FFFF   LLS   00D6FD78   JLB   00000000   JPQ   80D6F708
             RG 0-7   00000001   00060944   0006EF80   00060A14   00060A14   0006E320   E406C9C0   000608C8
             RG 8-15  00000000   0006C9C0   00000000   00E087C0   00063F22   000607A0   80E08944   4001D1A2
             FSA   0006EFB0   TCB      00D5FED0   TME  00000000   JSTCB 00D6F2F8   NTC   00000000   OTC   00D6FED0
             LTC   00D6F1C0   IQE      00000000   ECB  00D6F434   TSFLG 00000000   D-PQE 00D6FCA8   AQE   00D6F690
             STAB  00D6FD58   TCT      00000000   USER 00000000   NDSP  00000000   MDIDS 00000000   JSCB  00D6F66C
             RES   00000000   RES      00000000   RES  00000000   EXT1  00000000   BITS  00000000   DAR   00000000
             EXT2  00D6F3F8   XTENT    000601D6   TIRB 00D6FAC0   BACK  00D6FED0   LSQAP 00D6FFF8   IOTIM 00000000
             TMSAV 00000000   ABR-TID  00000000   QECB 00000000   FOE   00D6F6D8   SWA   00D6FFE0
     EXT2    GTF   00000000   RCMP     00000000


ACTIVE RBS

PRB  D6FA78  FLG1    40000000   WC-L-IC  00020001
             RESV    00000000   APSW    00000000   SZ-STAB     00050082   FL-CDE 00D6FA98   PSW 071D1000 00E08952
             Q       00000000   WT-LNK  01D6F2F8   NM CPMCP       EPA 060750   STA 060750   LN 0088B0   ATR1 0B   ATR2 20


MAIN STORAGE

D-PQE   D6FCA8    FIRST 00D6FC90    LAST 00D6FC90

PQE   D6FC90    FFB 00D6FE30   LFB 00D6FE30   NPQ 00000000   PPQ 00000000
                TCB 00D6FED0   RSI 00010000   RAD 00060000   FLG 00000000


LOAD LIST

CDE   FFE1C8    NM IGG019RO   USE 0001   RESP 01   ATR1 B1   ATR2 20   EPA E087B0   STA E087B0   LN 000278
CDE   FFE1F0    NM IGG019R3   USE 0001   RESP 01   ATR1 B1   ATR2 20   EPA E02870   STA E02870   LN 0001C8
CDE   FFE218    NM IGG019R1   USE 0001   RESP 01   ATR1 B1   ATR2 20   EPA E02C80   STA E02C80   LN 000380
CDE   FFE240    NM IGG019R4   USE 0001   RESP 01   ATR1 B1   ATR2 20   EPA E02380   STA E02380   LN 0004F0
CDE   FFE268    NM IGG019RR   USE 0001   RESP 01   ATR1 B1   ATR2 20   EPA E06FA0   STA E06FA0   LN 000060
CDE   FFE290    NM IGG019Q0   USE 0001   RESP 01   ATR1 B1   ATR2 20   EPA E14410   STA E14410   LN 000238
CDE   FFF4B8    NM IGG019RN   USE 0001   RESP 01   ATR1 B1   ATR2 20   EPA E08A28   STA E08A28   LN 000370
CDE   FFF4E0    NM IGG019QE   USE 0001   RESP 01   ATR1 B1   ATR2 20   EPA E14648   STA E14648   LN 0001A8
CDE   FFF3F0    NM IGG019Q3   USE 0001   RESP 01   ATR1 B1   ATR2 20   EPA E10B48   STA E10B48   LN 0014B8


JOB PACK QUEUE

CDE   D6F708    NM IEDQCA     USE 0001   RESP NA   ATR1 1B   ATR2 20   EPA 060598   STA 060598   LN 0001B8
CDE   D6FC78    NM IEDQOA     USE 0000   RESP NA   ATR1 1B   ATR2 60   EPA 069000   STA 069000   LN 001000
CDE   D6FA98    NM CPMCP      USE 0001   RESP NA   ATR1 0B   ATR2 20   EPA 060750   STA 060750   LN 0088B0


DEB D6E8BC    APPENDAGES      EOEA 90068470   SIOA 35E14648   PCIA 6EE08A28   CEA   97E10B48   XCEA 0CE10B48
              PFX 00000000    00000000    00080004    11000000
              TCB 09D6F2F8    NDEB 00000000   ASYN 08000000   SPRG 01000000   UPRG 0A000000   PLST 00000000   DCB 1F060C84
              AVT 02D6E898
               OP-UCB
              80000B58
```

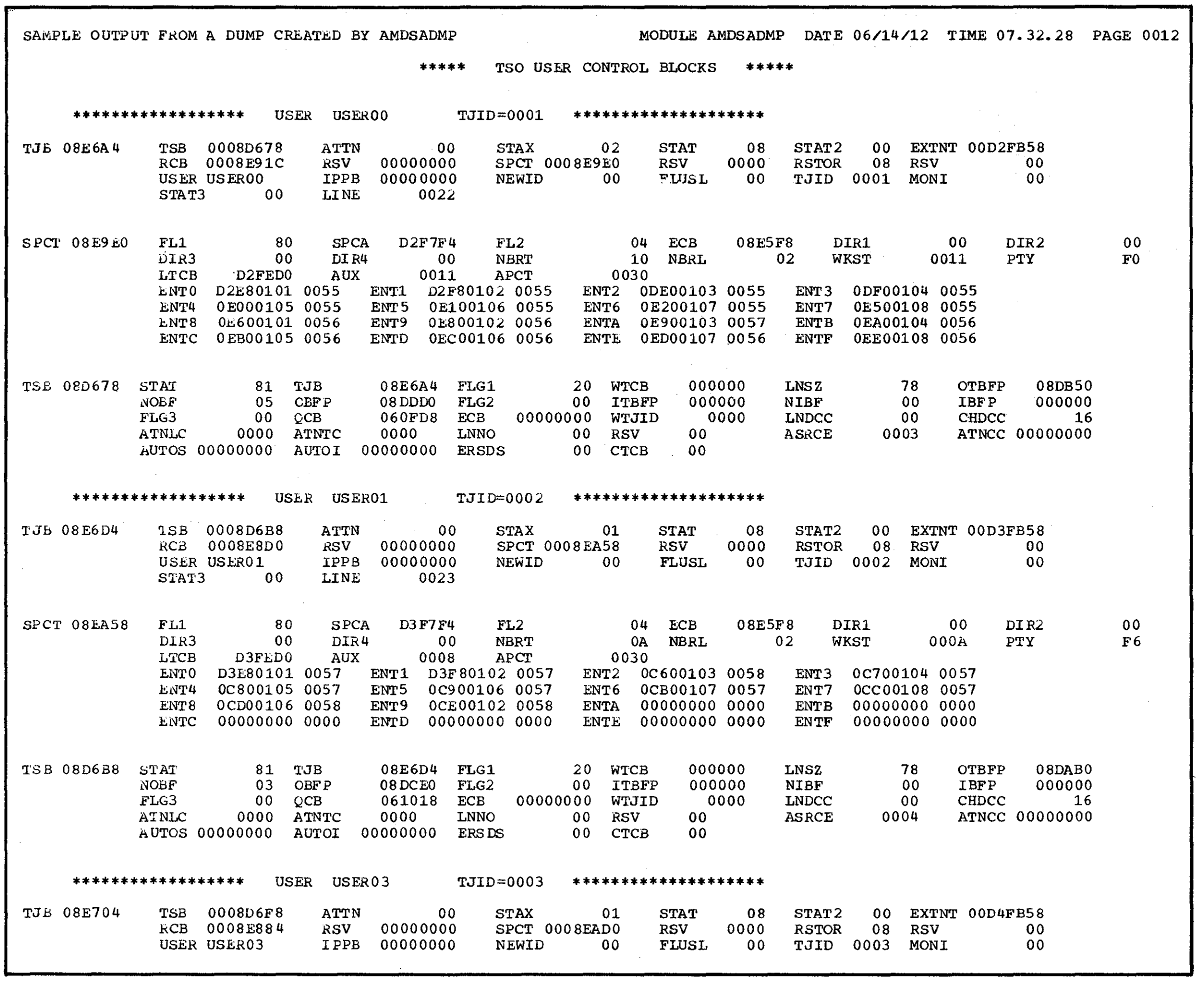
Figure PRDMP-16. Sample Formatted Data Areas (VS2 Only) (Part 1 of 2).



```
SAMPLE OUTPUT FROM A DUMP CREATED BY AMDSADMP                  MODULE AMDSADMP  DATE 06/14/12  TIME 07.32.28  PAGE 0015
             80000B70
             80000B88
             80000BA0
             80000BB8
             80000BD0
             80000BE8
             80000C00
             80000C18
             80000C30

TIOT  D6F478 JOB TCAM10     STEP TCAM10     PROC

             OFFSET    LN-STA    DDNAME     TTR-STC    STB-UCB
              0018    14040140  DIAL2741   00020B00   80000B58
              002C    14040100             00021000   80000B70
              0040    14040100             00021200   80000B88
              0054    14040100             00021400   80000BA0
              0068    14040100             00021600   80000BB8
              007C    14040100             00021800   80000BD0
              0090    14040100             00021A00   80000BE8
              00A4    14040100             00031800   80000C00
              00B8    14040100             00031A00   80000C18
              00CC    14040100             00040200   80000C30
```

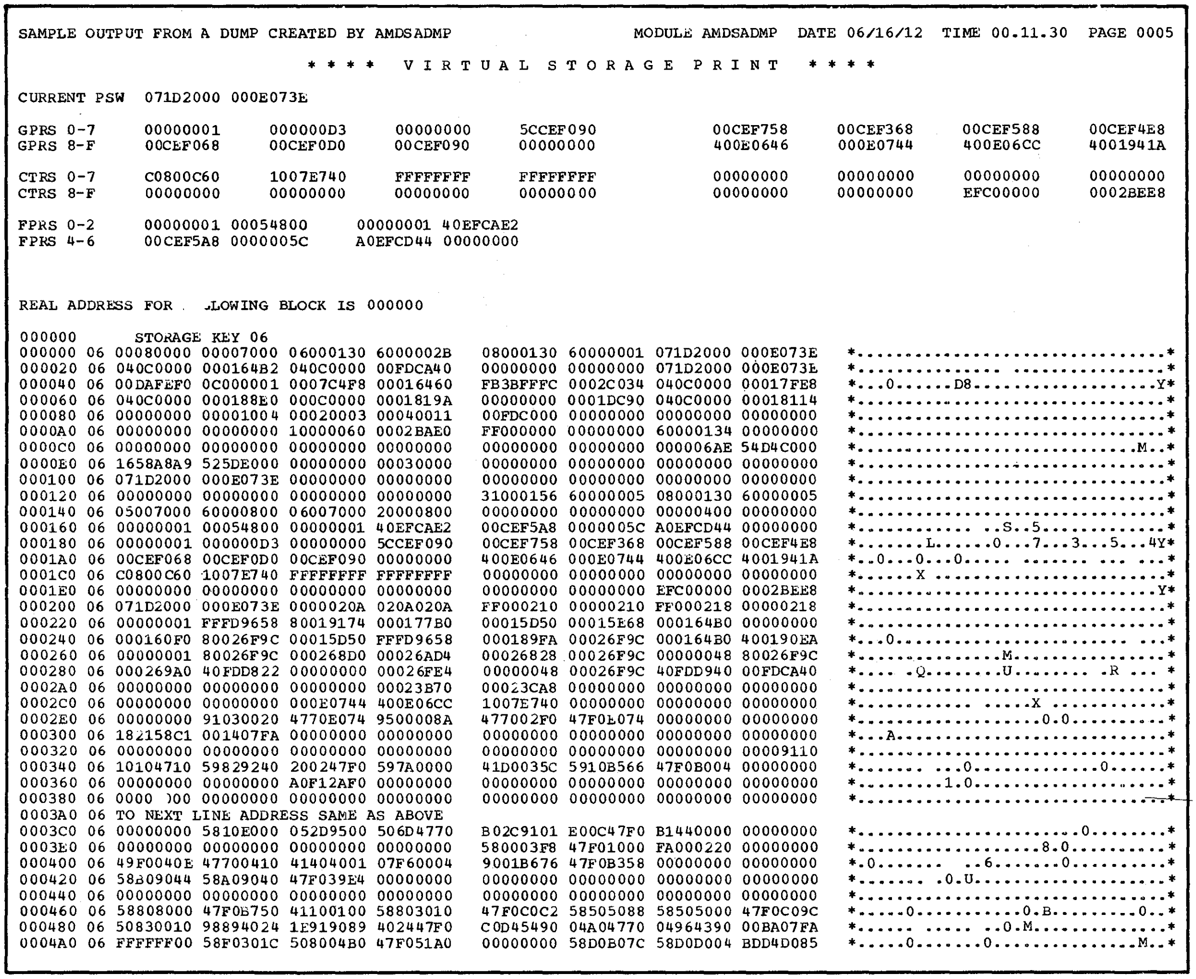
Figure PRDMP-16. Sample Formatted Data Areas (VS2 Only) (Part 2 of 2).

PRDM



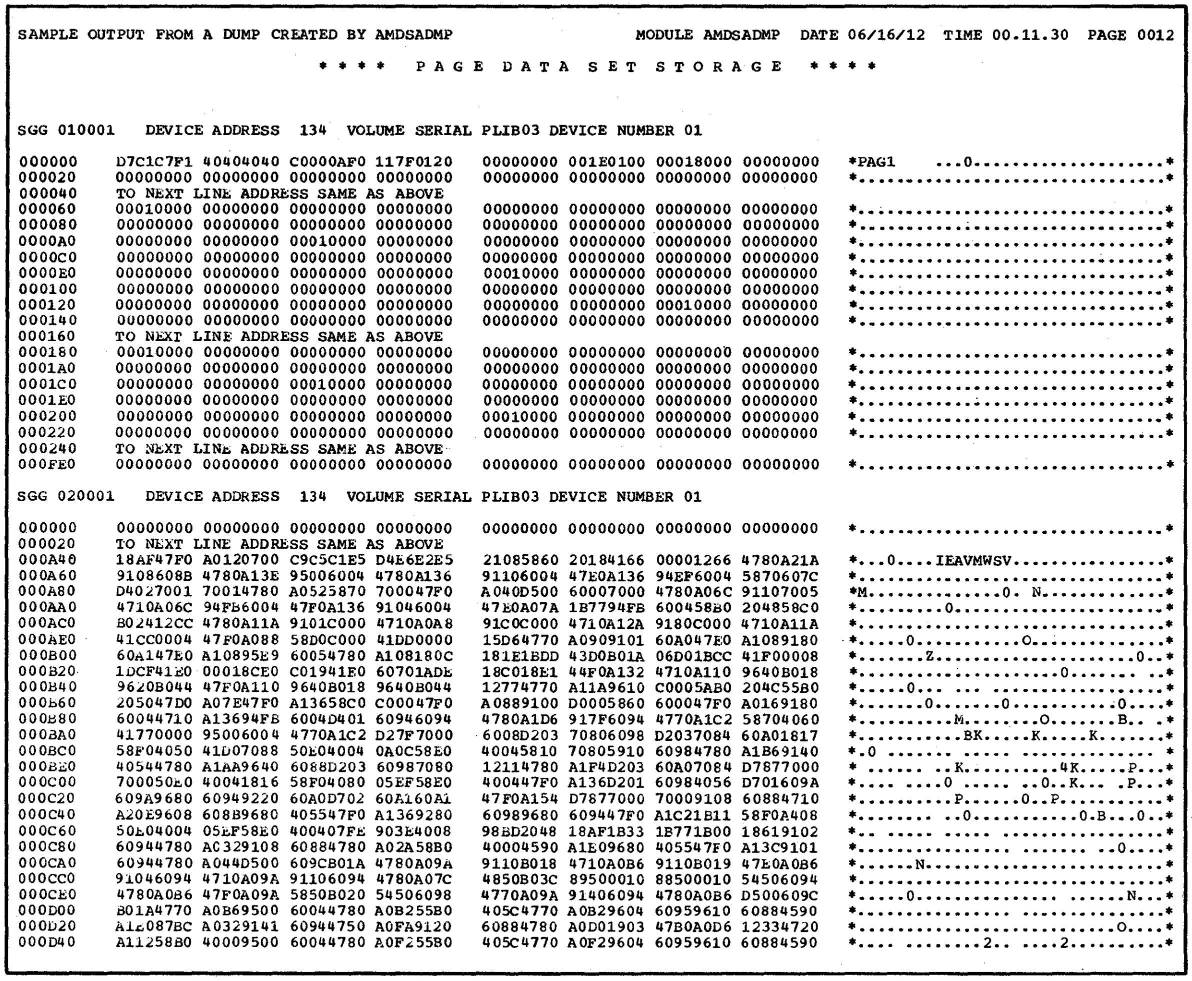
```
SAMPLE OUTPUT FROM A DUMP CREATED BY AMDSADMP                MODULE AMDSADMP  DATE 06/14/12  TIME 07.32.28  PAGE 0012
                                         *****   TSO USER CONTROL BLOCKS   *****

     ******************   USER  USER00       TJID=0001    ********************

TJB 08E6A4     TSB  0008D678     ATTN          00     STAX        02     STAT      08    STAT2    00   EXTNT 00D2FB58
               RCB  0008E91C     RSV     00000000     SPCT 0008E9E0     RSV     0000    RSTOR    08   RSV         00
               USER USER00       IPPB    00000000     NEWID       00     FLUSL     00    TJID  0001   MONI        00
               STAT3       00    LINE        0022

SPCT 08E9E0    FL1          80     SPCA   D2F7F4      FL2              04  ECB     08E5F8     DIR1         00     DIR2          00
               DIR3         00     DIR4       00      NBRT             10  NBRL         02    WKST       0011     PTY           F0
               LTCB    D2FED0      AUX      0011      APCT          0030
               ENT0  D2E80101 0055    ENT1  D2F80102 0055    ENT2  0DE00103 0055    ENT3  0DF00104 0055
               ENT4  0E000105 0055    ENT5  0E100106 0055    ENT6  0E200107 0055    ENT7  0E500108 0055
               ENT8  0E600101 0056    ENT9  0E800102 0056    ENTA  0E900103 0057    ENTB  0EA00104 0056
               ENTC  0EB00105 0056    ENTD  0EC00106 0056    ENTE  0ED00107 0056    ENTF  0EE00108 0056

TSB 08D678   STAT         81  TJB       08E6A4  FLG1          20  WTCB     000000    LNSZ         78    OTBFP    08DB50
             NOBF         05  CBFP      08DDD0  FLG2          00  ITBFP    000000    NIBF         00    IBFP     000000
             FLG3         00  QCB       060FD8  ECB     00000000  WTJID      0000    LNDCC        00    CHDCC        16
             ATNLC      0000  ATNTC     0000    LNNO          00  RSV      00        ASRCE      0003    ATNCC 00000000
             AUTOS 00000000   AUTOI  00000000   ERSDS         00  CTCB     00

     ******************   USER  USER01       TJID=0002    ********************

TJB 08E6D4     TSB  0008D6B8     ATTN          00     STAX        01     STAT      08    STAT2    00   EXTNT 00D3FB58
               RCB  0008E8D0     RSV     00000000     SPCT 0008EA58     RSV     0000    RSTOR    08   RSV         00
               USER USER01       IPPB    00000000     NEWID       00     FLUSL     00    TJID  0002   MONI        00
               STAT3       00    LINE        0023

SPCT 08EA58    FL1          80     SPCA   D3F7F4      FL2              04  ECB     08E5F8     DIR1         00     DIR2          00
               DIR3         00     DIR4       00      NBRT             0A  NBRL         02    WKST       000A     PTY           F6
               LTCB    D3FED0      AUX      0008      APCT          0030
               ENT0  D3E80101 0057    ENT1  D3F80102 0057    ENT2  0C600103 0058    ENT3  0C700104 0057
               ENT4  0C800105 0057    ENT5  0C900106 0057    ENT6  0CB00107 0057    ENT7  0CC00108 0057
               ENT8  0CD00106 0058    ENT9  0CE00102 0058    ENTA  00000000 0000    ENTB  00000000 0000
               ENTC  00000000 0000    ENTD  00000000 0000    ENTE  00000000 0000    ENTF  00000000 0000

TSB 08D6B8   STAT         81  TJB       08E6D4  FLG1          20  WTCB     000000    LNSZ         78    OTBFP    08DAB0
             NOBF         03  OBFP      08DCE0  FLG2          00  ITBFP    000000    NIBF         00    IBFP     000000
             FLG3         00  QCB       061018  ECB     00000000  WTJID      0000    LNDCC        00    CHDCC        16
             ATNLC      0000  ATNTC     0000    LNNO          00  RSV      00        ASRCE      0004    ATNCC 00000000
             AUTOS 00000000   AUTOI  00000000   ERSDS         00  CTCB     00

     ******************   USER  USER03       TJID=0003    ********************

TJB 08E704     TSB  0008D6F8     ATTN          00     STAX        01     STAT      08    STAT2    00   EXTNT 00D4FB58
               RCB  0008E884     RSV     00000000     SPCT 0008EAD0     RSV     0000    RSTOR    08   RSV         00
               USER USER03       IPPB    00000000     NEWID       00     FLUSL     00    TJID  0003   MONI        00
```

Figure PRDMP-17. Sample TSO Formatted User Data Areas (VS2 Only).



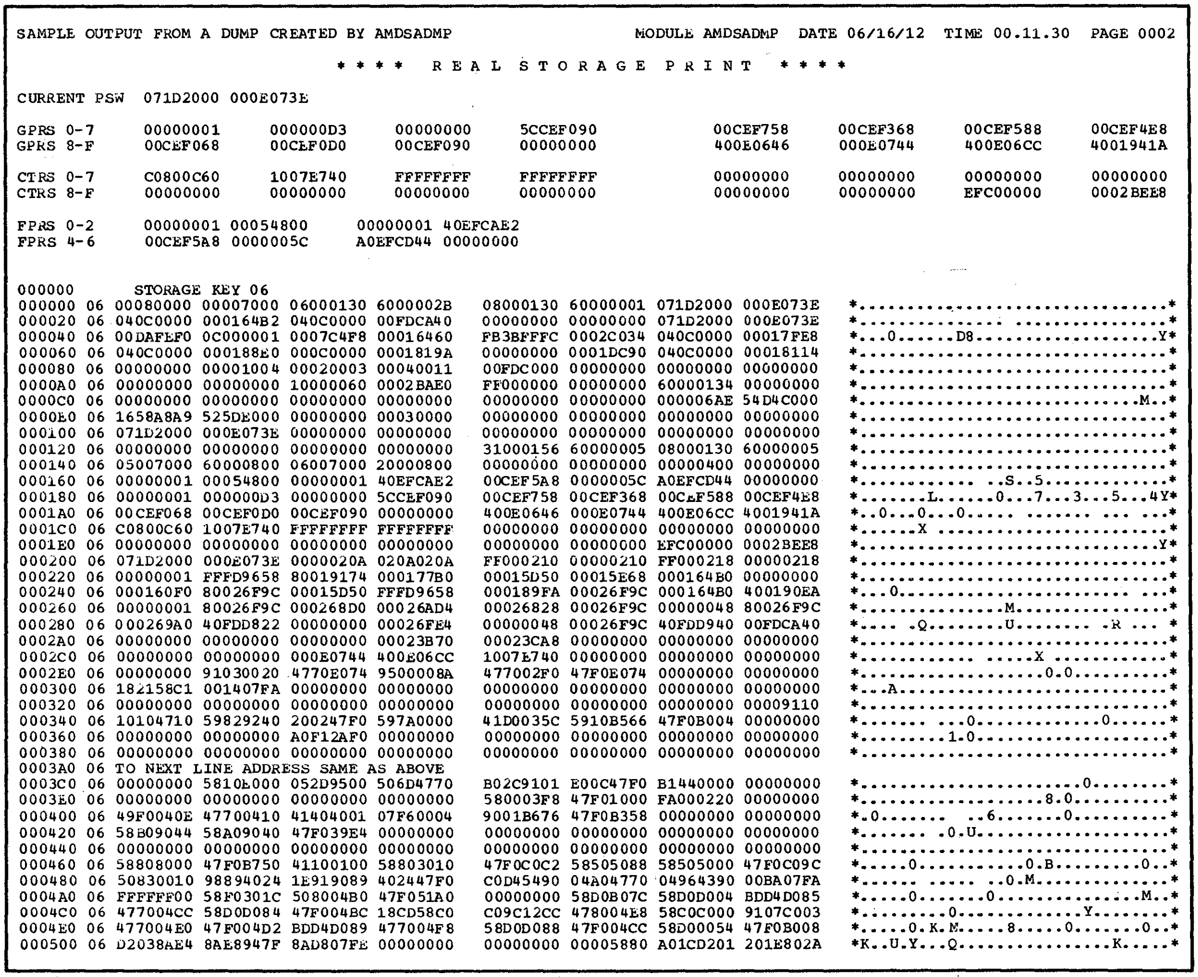
```
SAMPLE OUTPUT FROM A DUMP CREATED BY AMDSADMP                        MODULE AMDSADMP  DATE 06/16/12  TIME 00.11.30  PAGE 0005

                                       * * * *   V I R T U A L   S T O R A G E   P R I N T   * * * *

CURRENT PSW  071D2000 000E073E

GPRS 0-7       00000001     000000D3     00000000     5CCEF090                00CEF758     00CEF368     00CEF588     00CEF4E8
GPRS 8-F       00CEF068     00CEF0D0     00CEF090     00000000                400E0646     000E0744     400E06CC     4001941A

CTRS 0-7       C0800C60     1007E740     FFFFFFFF     FFFFFFFF                00000000     00000000     00000000     00000000
CTRS 8-F       00000000     00000000     00000000     00000000                00000000     00000000     EFC00000     0002BEE8

FPRS 0-2       00000001 00054800     00000001 40EFCAE2
FPRS 4-6       00CEF5A8 0000005C     A0EFCD44 00000000


REAL ADDRESS FOR . .LOWING BLOCK IS 000000

000000       STORAGE KEY 06
000000 06 00080000 00007000 06000130 6000002B  08000130 60000001 071D2000 000E073E   *................................*
000020 06 040C0000 000164B2 040C0000 00FDCA40  00000000 00000000 071D2000 000E073E   *................ ...............*
000040 06 00DAFEF0 0C000001 0007C4F8 00016460  FB3BFFFC 0002C034 040C0000 00017FE8   *...0......D8...................Y*
000060 06 040C0000 000188E0 000C0000 0001819A  00000000 0001DC90 040C0000 00018114   *................................*
000080 06 00000000 00001004 00020003 00040011  00FDC000 00000000 00000000 00000000   *................................*
0000A0 06 00000000 00000000 10000060 0002BAE0  FF000000 00000000 60000134 00000000   *................................*
0000C0 06 00000000 00000000 00000000 00000000  00000000 00000000 000006AE 54D4C000   *.............................M..*
0000E0 06 1658A8A9 525DE000 00000000 00030000  00000000 00000000 00000000 00000000   *................................*
000100 06 071D2000 000E073E 00000000 00000000  00000000 00000000 00000000 00000000   *................................*
000120 06 00000000 00000000 00000000 00000000  31000156 60000005 08000130 60000005   *................................*
000140 06 05007000 60000800 06007000 20000800  00000000 00000000 00000400 00000000   *................................*
000160 06 00000001 00054800 00000001 40EFCAE2  00CEF5A8 0000005C A0EFCD44 00000000   *............ ..S..5.............*
000180 06 00000001 000000D3 00000000 5CCEF090  00CEF758 00CEF368 00CEF588 00CEF4E8   *.......L......0...7...3...5...4Y*
0001A0 06 00CEF068 00CEF0D0 00CEF090 00000000  400E0646 000E0744 400E06CC 4001941A   *..0...0...0..... ....... ... ...*
0001C0 06 C0800C60 1007E740 FFFFFFFF FFFFFFFF  00000000 00000000 00000000 00000000   *......X ........................*
0001E0 06 00000000 00000000 00000000 00000000  00000000 00000000 EFC00000 0002BEE8   *...............................Y*
000200 06 071D2000 000E073E 0000020A 020A020A  FF000210 00000210 FF000218 00000218   *................................*
000220 06 00000001 FFFD9658 80019174 000177B0  00015D50 00015E68 000164B0 00000000   *................................*
000240 06 000160F0 80026F9C 00015D50 FFFD9658  000189FA 00026F9C 000164B0 400190EA   *...0........................ ...*
000260 06 00000001 80026F9C 000268D0 00026AD4  00026828 00026F9C 00000048 80026F9C   *..............M.................*
000280 06 000269A0 40FDD822 00000000 00026FE4  00000048 00026F9C 40FDD940 00FDCA40   *.... .Q........U........ .R ... *
0002A0 06 00000000 00000000 00000000 00023B70  00023CA8 00000000 00000000 00000000   *................................*
0002C0 06 00000000 00000000 000E0744 400E06CC  1007E740 00000000 00000000 00000000   *............ .....X ............*
0002E0 06 00000000 91030020 4770E074 9500008A  477002F0 47F0E074 00000000 00000000   *...................0.0..........*
000300 06 182158C1 001407FA 00000000 00000000  00000000 00000000 00000000 00000000   *...A............................*
000320 06 00000000 00000000 00000000 00000000  00000000 00000000 00000000 00009110   *................................*
000340 06 10104710 59829240 200247F0 597A0000  41D0035C 5910B566 47F0B004 00000000   *....... ...0..............0.....*
000360 06 00000000 00000000 A0F12AF0 00000000  00000000 00000000 00000000 00000000   *.........1.0....................*
000380 06 0000 )00 00000000 00000000 00000000  00000000 00000000 00000000 00000000   *................................*
0003A0 06 TO NEXT LINE ADDRESS SAME AS ABOVE
0003C0 06 00000000 5810E000 052D9500 506D4770  B02C9101 E00C47F0 B1440000 00000000   *.......................0........*
0003E0 06 00000000 00000000 00000000 00000000  580003F8 47F01000 FA000220 00000000   *..................8.0...........*
000400 06 49F0040E 47700410 41404001 07F60004  9001B676 47F0B358 00000000 00000000   *.0....... ..6.......0...........*
000420 06 58B09044 58A09040 47F039E4 00000000  00000000 00000000 00000000 00000000   *....... .0.U....................*
000440 06 00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000   *................................*
000460 06 58808000 47F0B750 41100100 58803010  47F0C0C2 58505088 58505000 47F0C09C   *.....0..........0.B..........0..*
000480 06 50830010 98894024 1E919089 402447F0  C0D45490 04A04770 04964390 00BA07FA   *...... ..... ..0.M..............*
0004A0 06 FFFFFF00 58F0301C 508004B0 47F051A0  00000000 58D0B07C 58D0D004 BDD4D085   *.....0.......0...............M..*
```

Figure PRDMP-18. Sample Dump -- General Format (VS2 Only).



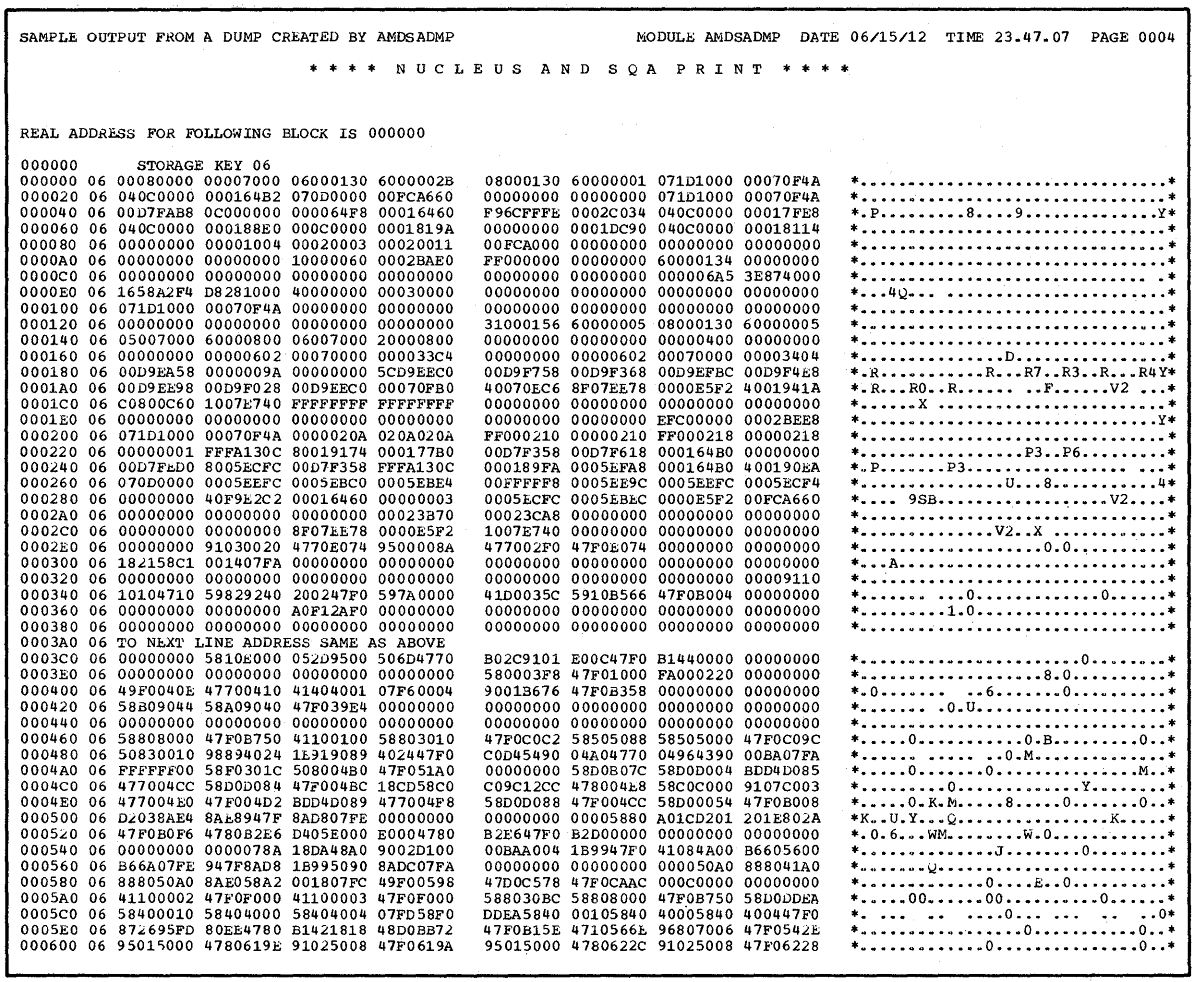
```
SAMPLE OUTPUT FROM A DUMP CREATED BY AMDSADMP                MODULE AMDSADMP  DATE 06/16/12  TIME 00.11.30  PAGE 0012
                                   * * * *   P A G E   D A T A   S E T   S T O R A G E   * * * *


SGG 010001   DEVICE ADDRESS  134  VOLUME SERIAL PLIB03 DEVICE NUMBER 01

000000     D7C1C7F1 40404040 C0000AF0 117F0120   00000000 001E0100 00018000 00000000   *PAG1    ...0....................*
000020     00000000 00000000 00000000 00000000   00000000 00000000 00000000 00000000   *................................*
000040     TO NEXT LINE ADDRESS SAME AS ABOVE
000060     00010000 00000000 00000000 00000000   00000000 00000000 00000000 00000000   *................................*
000080     00000000 00000000 00000000 00000000   00000000 00000000 00000000 00000000   *................................*
0000A0     00000000 00000000 00010000 00000000   00000000 00000000 00000000 00000000   *................................*
0000C0     00000000 00000000 00000000 00000000   00000000 00000000 00000000 00000000   *................................*
0000E0     00000000 00000000 00000000 00000000   00010000 00000000 00000000 00000000   *................................*
000100     00000000 00000000 00000000 00000000   00000000 00000000 00000000 00000000   *................................*
000120     00000000 00000000 00000000 00000000   00000000 00000000 00010000 00000000   *................................*
000140     00000000 00000000 00000000 00000000   00000000 00000000 00000000 00000000   *................................*
000160     TO NEXT LINE ADDRESS SAME AS ABOVE
000180     00010000 00000000 00000000 00000000   00000000 00000000 00000000 00000000   *................................*
0001A0     00000000 00000000 00000000 00000000   00000000 00000000 00000000 00000000   *................................*
0001C0     00000000 00000000 00010000 00000000   00000000 00000000 00000000 00000000   *................................*
0001E0     00000000 00000000 00000000 00000000   00000000 00000000 00000000 00000000   *................................*
000200     00000000 00000000 00000000 00000000   00010000 00000000 00000000 00000000   *................................*
000220     00000000 00000000 00000000 00000000   00000000 00000000 00000000 00000000   *................................*
000240     TO NEXT LINE ADDRESS SAME AS ABOVE
000FE0     00000000 00000000 00000000 00000000   00000000 00000000 00000000 00000000   *................................*

SGG 020001   DEVICE ADDRESS  134  VOLUME SERIAL PLIB03 DEVICE NUMBER 01

000000     00000000 00000000 00000000 00000000   00000000 00000000 00000000 00000000   *................................*
000020     TO NEXT LINE ADDRESS SAME AS ABOVE
000A40     18AF47F0 A0120700 C9C5C1E5 D4E6E2E5   21085860 20184166 00001266 4780A21A   *...0....IEAVMWSV................*
000A60     9108608B 4780A13E 95006004 4780A136   91106004 47E0A136 94EF6004 5870607C   *................................*
000A80     D4027001 70014780 A0525870 700047F0   A040D500 60007000 4780A06C 91107005   *M..............0. N.............*
000AA0     4710A06C 94FB6004 47F0A136 91046004   47E0A07A 1B7794FB 600458B0 204858C0   *.........0......................*
000AC0     B02412CC 4780A11A 9101C000 4710A0A8   91C0C000 4710A12A 9180C000 4710A11A   *................................*
000AE0     41CC0004 47F0A088 58D0C000 41DD0000   15D64770 A0909101 60A047E0 A1089180   *.....0...........O..............*
000B00     60A147E0 A10895E9 60054780 A108180C   181E1BDD 43D0B01A 06D01BCC 41F00008   *.......Z.....................0..*
000B20     1DCF41E0 00018CE0 C01941E0 60701ADE   18C018E1 44F0A132 4710A110 9640B018   *.....................0....... ..*
000B40     9620B044 47F0A110 9640B018 9640B044   12774770 A11A9610 C0005AB0 204C55B0   *.....0... ... ..................*
000B60     205047D0 A07E47F0 A13658C0 C00047F0   A0889100 D0005860 600047F0 A0169180   *.......0.......0...........0....*
000B80     60044710 A13694FB 6004D401 60946094   4780A1D6 917F6094 4770A1C2 58704060   *..........M........O.......B.. .*
000BA0     41770000 95006004 4770A1C2 D27F7000   6008D203 70806098 D2037084 60A01817   *...........BK.....K.....K.......*
000BC0     58F04050 41D07088 50E04004 0A0C58E0   40045810 70805910 60984780 A1B69140   *.0 ....... ..... .............. *
000BE0     40544780 A1AA9640 6088D203 60987080   12114780 A1F4D203 60A07084 D7877000   * ...... ..K..........4K.....P...*
000C00     700050E0 40041816 58F04080 05EF58E0   400447F0 A136D201 60984056 D701609A   *.... ....0 ..... ..0..K... .P...*
000C20     609A9680 60949220 60A0D702 60A160A1   47F0A154 D7877000 70009108 60884710   *..........P......0..P...........*
000C40     A20E9608 608B9680 405547F0 A1369280   60989680 609447F0 A1C21B11 58F0A408   *........ ..0...........0.B...0..*
000C60     50E04004 05EF58E0 400407FE 903E4008   98BD2048 18AF1B33 1B771B00 18619102   *.. ..... ..... .................*
000C80     60944780 AC329108 60884780 A02A58B0   40004590 A1E09680 405547F0 A13C9101   *................ ....... ..0....*
000CA0     60944780 A044D500 609CB01A 4780A09A   9110B018 4710A0B6 9110B019 47E0A0B6   *......N.........................*
000CC0     91046094 4710A09A 91106094 4780A07C   4850B03C 89500010 88500010 54506094   *................................*
000CE0     4780A0B6 47F0A09A 5850B020 54506098   4770A09A 91406094 4780A0B6 D500609C   *.....0............... ......N...*
000D00     B01A4770 A0B69500 60044780 A0B255B0   405C4770 A0B29604 60959610 60884590   *................ ...............*
000D20     A1E087BC A0329141 60944750 A0FA9120   60884780 A0D01903 47B0A0D6 12334720   *...........................O....*
000D40     A11258B0 40009500 60044780 A0F255B0   405C4770 A0F29604 60959610 60884590   *.... ........2.. ....2..........*
```

Figure PRDMP-19. Sample Dump -- Page Data Set (VS2 Only).



```
SAMPLE OUTPUT FROM A DUMP CREATED BY AMDSADMP                    MODULE AMDSADMP  DATE 06/16/12  TIME 00.11.30  PAGE 0002

                                    * * * *   R E A L  S T O R A G E  P R I N T   * * * *

CURRENT PSW   071D2000 000E073E

GPRS 0-7      00000001     000000D3     00000000     5CCEF090          00CEF758     00CEF368     00CEF588     00CEF4E8
GPRS 8-F      00CEF068     00CEF0D0     00CEF090     00000000          400E0646     000E0744     400E06CC     4001941A

CTRS 0-7      C0800C60     1007E740     FFFFFFFF     FFFFFFFF          00000000     00000000     00000000     00000000
CTRS 8-F      00000000     00000000     00000000     00000000          00000000     00000000     EFC00000     0002BEE8

FPRS 0-2      00000001 00054800     00000001 40EFCAE2
FPRS 4-6      00CEF5A8 0000005C     A0EFCD44 00000000


000000       STORAGE KEY 06
000000 06 00080000 00007000 06000130 6000002B   08000130 60000001 071D2000 000E073E   *................................*
000020 06 040C0000 000164B2 040C0000 00FDCA40   00000000 00000000 071D2000 000E073E   *............. ..................*
000040 06 00DAFEF0 0C000001 0007C4F8 00016460   FB3BFFFC 0002C034 040C0000 00017FE8   *...O......D8...................Y*
000060 06 040C0000 000188E0 000C0000 0001819A   00000000 0001DC90 040C0000 00018114   *................................*
000080 06 00000000 00001004 00020003 00040011   00FDC000 00000000 00000000 00000000   *................................*
0000A0 06 00000000 00000000 10000060 0002BAE0   FF000000 00000000 60000134 00000000   *................................*
0000C0 06 00000000 00000000 00000000 00000000   00000000 00000000 000006AE 54D4C000   *.............................M..*
0000E0 06 1658A8A9 525DE000 00000000 00030000   00000000 00000000 00000000 00000000   *................................*
000100 06 071D2000 000E073E 00000000 00000000   00000000 00000000 00000000 00000000   *................................*
000120 06 00000000 00000000 00000000 00000000   31000156 60000005 08000130 60000005   *................................*
000140 06 05007000 60000800 06007000 20000800   00000000 00000000 00000400 00000000   *................................*
000160 06 00000001 00054800 00000001 40EFCAE2   00CEF5A8 0000005C A0EFCD44 00000000   *........... ..S..5..............*
000180 06 00000001 000000D3 00000000 5CCEF090   00CEF758 00CEF368 00CEF588 00CEF4E8   *.......L......O...7...3...5...4Y*
0001A0 06 00CEF068 00CEF0D0 00CEF090 00000000   400E0646 000E0744 400E06CC 4001941A   *..O...O...O..... ....... ... ...*
0001C0 06 C0800C60 1007E740 FFFFFFFF FFFFFFFF   00000000 00000000 00000000 00000000   *......X ........................*
0001E0 06 00000000 00000000 00000000 00000000   00000000 00000000 EFC00000 0002BEE8   *...............................Y*
000200 06 071D2000 000E073E 0000020A 020A020A   FF000210 00000210 FF000218 00000218   *................................*
000220 06 00000001 FFFD9658 80019174 000177B0   00015D50 00015E68 000164B0 00000000   *................................*
000240 06 000160F0 80026F9C 00015D50 FFFD9658   000189FA 00026F9C 000164B0 400190EA   *...O........................ ...*
000260 06 00000001 80026F9C 000268D0 00026AD4   00026828 00026F9C 00000048 80026F9C   *...............M................*
000280 06 000269A0 40FDD822 00000000 00026FE4   00000048 00026F9C 40FDD940 00FDCA40   *.... .Q........U........ .R ... *
0002A0 06 00000000 00000000 00000000 00023B70   00023CA8 00000000 00000000 00000000   *................................*
0002C0 06 00000000 00000000 000E0744 400E06CC   1007E740 00000000 00000000 00000000   *........... .....X .............*
0002E0 06 00000000 91030020 4770E074 9500008A   477002F0 47F0E074 00000000 00000000   *.................O.O............*
000300 06 182158C1 001407FA 00000000 00000000   00000000 00000000 00000000 00000000   *...A............................*
000320 06 00000000 00000000 00000000 00000000   00000000 00000000 00000000 00009110   *................................*
000340 06 10104710 59829240 200247F0 597A0000   41D0035C 5910B566 47F0B004 00000000   *....... ...O.............O......*
000360 06 00000000 00000000 A0F12AF0 00000000   00000000 00000000 00000000 00000000   *.........1.O....................*
000380 06 00000000 00000000 00000000 00000000   00000000 00000000 00000000 00000000   *................................*
0003A0 06 TO NEXT LINE ADDRESS SAME AS ABOVE
0003C0 06 00000000 5810E000 052D9500 506D4770   B02C9101 E00C47F0 B1440000 00000000   *.......................O........*
0003E0 06 00000000 00000000 00000000 00000000   580003F8 47F01000 FA000220 00000000   *..................8.O...........*
000400 06 49F0040E 47700410 41404001 07F60004   9001B676 47F0B358 00000000 00000000   *.O....... ..6......O............*
000420 06 58B09044 58A09040 47F039E4 00000000   00000000 00000000 00000000 00000000   *....... .O.U....................*
000440 06 00000000 00000000 00000000 00000000   00000000 00000000 00000000 00000000   *................................*
000460 06 58808000 47F0B750 41100100 58803010   47F0C0C2 58505088 58505000 47F0C09C   *.....O...........O.B.........O..*
000480 06 50830010 98894024 1E919089 402447F0   C0D45490 04A04770 04964390 00BA07FA   *...... ..... ..O.M..............*
0004A0 06 FFFFFF00 58F0301C 508004B0 47F051A0   00000000 58D0B07C 58D0D004 BDD4D085   *.....O.......O...............M..*
0004C0 06 477004CC 58D0D084 47F004BC 18CD58C0   C09C12CC 478004E8 58C0C000 9107C003   *.........O............Y.........*
0004E0 06 477004E0 47F004D2 BDD4D089 477004F8   58D0D088 47F004CC 58D00054 47F0B008   *.....O.K.M.....8.....O.......O..*
000500 06 D2038AE4 8AE8947F 8AD807FE 00000000   00000000 00005880 A01CD201 201E802A   *K..U.Y...Q................K.....*
```

Figure PRDMP-20. Sample Dump -- Real Storage (VS2 Only).



```
SAMPLE OUTPUT FROM A DUMP CREATED BY AMDSADMP                    MODULE AMDSADMP  DATE 06/15/12  TIME 23.47.07  PAGE 0004
                                   * * * *  N U C L E U S   A N D   S Q A   P R I N T  * * * *


REAL ADDRESS FOR FOLLOWING BLOCK IS 000000

000000       STORAGE KEY 06
000000 06 00080000 00007000 06000130 6000002B   08000130 60000001 071D1000 00070F4A   *................................*
000020 06 040C0000 000164B2 070D0000 00FCA660   00000000 00000000 071D1000 00070F4A   *................................*
000040 06 00D7FAB8 0C000000 000064F8 00016460   F96CFFFE 0002C034 040C0000 00017FE8   *.P.........8....9..............Y*
000060 06 040C0000 000188E0 000C0000 0001819A   00000000 0001DC90 040C0000 00018114   *................................*
000080 06 00000000 00001004 00020003 00020011   00FCA000 00000000 00000000 00000000   *................................*
0000A0 06 00000000 00000000 10000060 0002BAE0   FF000000 00000000 60000134 00000000   *................................*
0000C0 06 00000000 00000000 00000000 00000000   00000000 00000000 000006A5 3E874000   *.............................. .*
0000E0 06 1658A2F4 D8281000 40000000 00030000   00000000 00000000 00000000 00000000   *...4Q... .......................*
000100 06 071D1000 00070F4A 00000000 00000000   00000000 00000000 00000000 00000000   *................................*
000120 06 00000000 00000000 00000000 00000000   31000156 60000005 08000130 60000005   *................................*
000140 06 05007000 60000800 06007000 20000800   00000000 00000000 00000400 00000000   *................................*
000160 06 00000000 00000602 00070000 000033C4   00000000 00000602 00070000 00003404   *...............D................*
000180 06 00D9EA58 0000009A 00000000 5CD9EEC0   00D9F758 00D9F368 00D9EFBC 00D9F4E8   *.R...........R...R7..R3..R...R4Y*
0001A0 06 00D9EE98 00D9F028 00D9EEC0 00070FB0   40070EC6 8F07EE78 0000E5F2 4001941A   *.R...R0..R...... ..F......V2 ...*
0001C0 06 C0800C60 1007E740 FFFFFFFF FFFFFFFF   00000000 00000000 00000000 00000000   *......X ........................*
0001E0 06 00000000 00000000 00000000 00000000   00000000 00000000 EFC00000 0002BEE8   *...............................Y*
000200 06 071D1000 00070F4A 0000020A 020A020A   FF000210 00000210 FF000218 00000218   *................................*
000220 06 00000001 FFFA130C 80019174 000177B0   00D7F358 00D7F618 000164B0 00000000   *................P3..P6..........*
000240 06 00D7FED0 8005ECFC 00D7F358 FFFA130C   000189FA 0005EFA8 000164B0 400190EA   *.P.......P3................. ...*
000260 06 070D0000 0005EEFC 0005EBC0 0005EBE4   00FFFFF8 0005EE9C 0005EEFC 0005ECF4   *...............U...8...........4*
000280 06 00000000 40F9E2C2 00016460 00000003   0005ECFC 0005EBEC 0000E5F2 00FCA660   *.... 9SB.................V2.....*
0002A0 06 00000000 00000000 00000000 00023B70   00023CA8 00000000 00000000 00000000   *................................*
0002C0 06 00000000 00000000 8F07EE78 0000E5F2   1007E740 00000000 00000000 00000000   *..............V2..X ............*
0002E0 06 00000000 91030020 4770E074 9500008A   477002F0 47F0E074 00000000 00000000   *...................0.0..........*
000300 06 182158C1 001407FA 00000000 00000000   00000000 00000000 00000000 00000000   *...A............................*
000320 06 00000000 00000000 00000000 00000000   00000000 00000000 00000000 00009110   *................................*
000340 06 10104710 59829240 200247F0 597A0000   41D0035C 5910B566 47F0B004 00000000   *....... ...0.............0......*
000360 06 00000000 00000000 A0F12AF0 00000000   00000000 00000000 00000000 00000000   *.........1.0....................*
000380 06 00000000 00000000 00000000 00000000   00000000 00000000 00000000 00000000   *................................*
0003A0 06 TO NEXT LINE ADDRESS SAME AS ABOVE
0003C0 06 00000000 5810E000 052D9500 506D4770   B02C9101 E00C47F0 B1440000 00000000   *.......................0........*
0003E0 06 00000000 00000000 00000000 00000000   580003F8 47F01000 FA000220 00000000   *...................8.0..........*
000400 06 49F0040E 47700410 41404001 07F60004   9001B676 47F0B358 00000000 00000000   *.0.......  ..6.......0..........*
000420 06 58B09044 58A09040 47F039E4 00000000   00000000 00000000 00000000 00000000   *....... .0.U....................*
000440 06 00000000 00000000 00000000 00000000   00000000 00000000 00000000 00000000   *................................*
000460 06 58808000 47F0B750 41100100 58803010   47F0C0C2 58505088 58505000 47F0C09C   *.....0...........0.B.........0..*
000480 06 50830010 98894024 1E919089 402447F0   C0D45490 04A04770 04964390 00BA07FA   *...... ..... ..0.M..............*
0004A0 06 FFFFFF00 58F0301C 508004B0 47F051A0   00000000 58D0B07C 58D0D004 BDD4D085   *.....0.......0...............M..*
0004C0 06 477004CC 58D0D084 47F004BC 18CD58C0   C09C12CC 478004E8 58C0C000 9107C003   *.........0.............Y........*
0004E0 06 477004E0 47F004D2 BDD4D089 477004F8   58D0D088 47F004CC 58D00054 47F0B008   *.....0.K.M.....8.....0.......0..*
000500 06 D2038AE4 8AE8947F 8AD807FE 00000000   00000000 00005880 A01CD201 201E802A   *K..U.Y...Q................K.....*
000520 06 47F0B0F6 4780B2E6 D405E000 E0004780   B2E647F0 B2D00000 00000000 00000000   *.0.6...WM........W.0............*
000540 06 00000000 0000078A 18DA48A0 9002D100   00BAA004 1B9947F0 41084A00 B6605600   *..............J........0........*
000560 06 B66A07FE 947F8AD8 1B995090 8ADC07FA   00000000 00000000 000050A0 888041A0   *.........Q......................*
000580 06 888050A0 8AE058A2 001807FC 49F00598   47D0C578 47F0CAAC 000C0000 00000000   *.............0....E..0..........*
0005A0 06 41100002 47F0F000 41100003 47F0F000   588030BC 58808000 47F0B750 58D0DDEA   *.....00......00..........0......*
0005C0 06 58400010 58404000 58404004 07FD58F0   DDEA5840 00105840 40005840 400447F0   *. ...  ..  ....0... ...   .. ..0*
0005E0 06 872695FD 80EE4780 B1421818 48D0BB72   47F0B15E 4710566E 96807006 47F0542E   *.................0...........0..*
000600 06 95015000 4780619E 91025008 47F0619A   95015000 4780622C 91025008 47F06228   *.............0...............0..*
```

Figure PRDMP-21. Sample Dump -- Nucleus and SQA (VS2 Only).

JCL and Control Statement Examples

The following examples illustrate some of the functions that PRDMP can perform.

Example 1: Using the Cataloged Procedure

IBM supplies a cataloged procedure, called PRDMP, that defines the input and output data sets and a work data set for PRDMP. This example shows how to use the cataloged procedure.

```
//PROCDMP      JOB       MSGLEVEL=(1,1)[,REGION=128K]
//STEP1        EXEC      PROC=PRDMP,PARM.DMP=T
//DMP.SYSIN    DD        *
    GO
    END
/*
```

In this example:

JOB Statement

initiates the job. The REGION= parameter applies only to VS2.

EXEC Statement

calls the cataloged procedure, and requests prompting for a dump title.

DMP.SYSIN DD Statement

defines the data set that contains the PRDMP control statements. The data set follows immediately.

GO Control Statement

requests formatting and printing according the the QCBTRACE, LPAMAP, FORMAT, EDIT, and PRINT ALL control statements.

END Control Statement

terminates PRDMP processing.

Example 2: Processing a TSO Dump Using the Cataloged Procedure (VS2 only)

This example shows how to limit PRDMP processing if an error occurs during initialization.

```
//FMTTSO1      JOB  MSGLEVEL=(1,1),REGION=128K
//             EXEC PROC=PRDMP,PARM=0
//TAPE         DD   UNIT=2400,VOL=SER=TSDUMP,LABEL=(,NL),DISP=OLD
//SYSIN        DD   *
         LPAMAP
         FORMAT
         TSO
         END
/*
```

In this example:

EXEC Statement

calls the cataloged procedure PRDMP. The PARM=0 parameter requests PRDMP to print only the nucleus and the SQA if a format error occurs during initialization.

TAPE DD Statement

defines the input data set, which in this case resides on tape.

SYSIN DD Statement

defines the data set containing the PRDMP control statements.

LPAMAP Control Statement

requests a map of the link pack area active queue.

FORMAT Control Statement

requests formatting and printing of the following system data areas from the dumped system:

Task Control Blocks (TCBs)
Request Blocks (RBs)
Problem Program Boundaries
Load List
Job Pack Queue
Data Extent Blocks (DEBs)
Task Input/Output Table (TIOTs)

TSO Control Statement (with no parameters)

requests formatting and printing of all TSO system and user control blocks and the TSO user regions.

END Control Statement

terminates processing.

Example 3: Transferring a Dump Data Set and Processing It in the Later Job

If you need to clear the SYS1.DUMP data set quickly to make room for more dump information, you can use PRDMP to transfer its contents to another data set. This new data set is not formatted or printed during this execution of PRDMP, but it can be used as input later.

This example shows how to transfer the SYS1.DUMP data set, which ordinarily is a cataloged data set on direct access storage, to a tape volume described by the SYSUT2 DD statement. It also shows how to refer to the transferred data set in a later job.

NOTES: 1. When transferring SYS1.DUMP to a SYSUT2 data set, do not use the cataloged procedure PRDMP; the cataloged procedure contains a SYSUT1 DD statement, and the SYSUT1 and SYSUT2 DD statements may never be used in the same step.

2. If the SYS1.DUMP data set is date protected, the operator will receive message IEC107D requesting permission to proceed. You must respond by entering r 00,'U' to allow PRDMP to continue processing.

3. This example does not include any PRDMP control statements except END. If other control statements were included, they would be ignored.

```
//CLEAR        JOB        MSGLEVEL=(1,1)[,REGION=128K]
//STEP1        EXEC       PGM=xMDPRDMP
//SYSPRINT     DD         SYSOUT=A
//PRINTER      DD         SYSOUT=A
//TAPE         DD         DSNAME=SYS1.DUMP,DISP=OLD
//SYSUT2       DD         DSN=DUMP1,UNIT=2400,VOL=SER=DUMP,LABEL=(,NL),
//         DISP=(NEW,KEEP)
//SYSIN        DD         *
         END
/*
************************************************************
//PROCESS      JOB        MSGLEVEL=(1,1)
//STEP         EXEC PGM=xMDPRDMP
//TAPE         DD         DSN=DUMP1,VOL=SER=DUMP,LABEL=(,NL),
//         DISP=OLD,UNIT=2400
//SYSUT1 DD         UNIT=3330,SPACE=(2056,(257,1)),
//         DISP=(NEW,DELETE),VOL=SER=111111
//PRINTER      DD         SYSOUT=A
//SYSPRINT     DD         SYSOUT=A
//SYSIN  DD          *
         FORMAT
         PRINT ALL
         END
/*
```

This example consists of two separate jobs. In the first job:

JOB Statement

initiates the job. The REGION= parameter applies to VS2 only.

EXEC Statement

calls for the execution of HMDPRDMP (VS1 only) or AMDPRDMP (VS2 only).

SYSPRINT DD statement

defines the message data set.

PRINTER DD Statement

defines the data set to which PRDMP ordinarily directs its output. This statement must be included, even though its function is not used in this application.

TAPE DD Statement

defines the input data set, SYS1.DUMP.

SYSUT2 DD Statement

defines the data set to which the contents of SYS1.DUMP will be transferred

SYSIN DD Statement

defines the data set that contains the PRDMP control statements.

The data set follows immediately.

END Control Statement

terminates PRDMP processing. Note that this is the only PRDMP control statement needed.

Note: If one of the format control statements (such as QCBTRACE, FORMAT, PRINT, etc.) is included in the input stream, the data transfer will take place, but no formatting will be done; any subsequent statements will be ignored.

In the second job:

TAPE DD Statement

defines the input data set, which in this case is the transferred dump data set processed in the first job.

SYSUT1 DD Statement

defines a work data set into which the input data will be collected and from which it will be processed.

PRINTER DD Statement

defines the output data set.

SYSPRINT DD Statement

defines the message data set.

SYSIN DD Statement

defines the data set that contains the PRDMP control statements. The data set follows immediately.

FORMAT Control Statement

requests formatting of important system data areas.

PRINT ALL Control Statement

requests printing of the resident nucleus, the system queue area, the pageable nucleus, all virtual storage allocated to partitions in the input data set, and the dumped system's registers and current PSW.

Example 4: Transferring the SYS1.DUMP Data Set and Processing it in the Same Step

If you want to transfer the contents of SYS1.DUMP data set to another data set and process the dump immediately, you can use a job stream like the one shown here. Note that the dump is directed to a data set defined by the SYSUT1 DD Statement, and the the SYSUT2 DD statement is not used.

If the SYS1.DUMP data set is date protected, the operator will recieve message IEC107D requesting permission to proceed. You must respond by entering r 00'U' to allow PRDMP to continue.

```
//TRANS     JOB     MSGLEVEL=(1,1)[,REGION=128K]
//STEP1     EXEC    PGM=xMDPRDMP
//SYSPRINT DD SYSOUT=A
//PRINTER  DD SYSOUT=A
//TAPE      DD      DSNAME=SYS1.DUMP,DISP=OLD
//SYSUT1    DD      DSN=DUMP2,UNIT=3330,VOL=SER=666666,DISP=(NEW,KEEP),
//        SPACE=(2056,(257,1))
//SYSIN     DD      *
          TITLE SYS1.DUMP THURSDAY PM
          GO
          END
/*
```

<u>In this example:</u>

JOB Statement

initiates the job. The REGION= parameter applies only to VS2.

EXEC Statement

calls for the execution of HMDPRDMP (VS1 only) or AMDPRDMP (VS2 only).

SYSPRINT DD Statement

defines the data set to which PRDMP directs its output, which in this case is the processed dump.

TAPE DD Statmenet

defines the input data set, SYS1.DUMP.

SYSUT1 DD Statement

defines a direct access data set to which the contents of SYS1.DUMP will be transferred, and from shich PRDMP will process the transferred dump. In this example, the SYSUT1 data set is to be kept, to allow further processing a a later time; when you keep the SYSUT1 data set, do not direct more than one dump data set to it.

SYSIN DD Statement

defines the data set that contains the PRDMP control statements. The data set follows immediately.

TITLE Control Statement

supplies a title for the processed dump.

GO Control Statement

instructs PRDMP to process the data set defined by the SYSUT1 DD statement.

END Control Statement

terminates PRDMP processing.

If you want to process the transferred dump data set again later, define it using a TAPE DD statement and treat it like any direct access input data set.

Example 5: Processing Multiple Data Sets

PRDMP can process any number of input data sets in a single execution, provided that each data set is properly defined by both DD statements and control statements. This example shows how to process three data sets in the same execution, two of which are on the same tape volume.

```
//NOLINK       JOB        MSGLEVEL=(1,1)[,REGION=128K]
//STEP1        EXEC       PGM=xMDPRDMP,PARM='T'
//SYSPRINT     DD         SYSOUT=A
//PRINTER      DD         SYSOUT=A,SPACE=(121,(1600,100))
//TAPE         DD         UNIT=2400,VOL=SER=DPTAPE,
//         LABEL=(,NL),DISP=OLD
//TODAYDMP     DD         UNIT=SYSDA,VOL=SER=DADUMP,
//         DSNAME=DMPDS,DISP=OLD
//SYSUT1       DD         UNIT=SYSDA,DISP=(NEW,DELETE),
//         SPACE=(2056,(257,10))
//SYSIN        DD         *
    ONGO              Q,F,P A
    GO
    NEWDUMP           FILESEQ=2
    GO
    NEWDUMP           DDNAME=TODAYDMP
    ONGO
    GO
    END
/*
```

In this example:

JOB Statement

initiates the job. The REGION= parameter applies only to VS2.

EXEC statement

calls for the execution of HMDPRDMP (VS1 only) or AMDPRDMP (VS2 only) and requests that the operator be prompted for a dump title.

SYSPRINT DD statement

defines the message data set.

PRINTER DD statement

defines the output data set.

TAPE DD statement

defines two input data sets on the same tape volume.

TODAYDMP DD statement

identifies an input data set on a direct access volume.

SYSUT1 DD statement

defines the PRDMP work data set; it is required in this example because one of the input data sets is on a direct access volume. For VS2, the SPACE= parameter should be coded: SPACE=(2056,(257,10)).

SYSIN DD statement

defines the data set containing the control statements. The data set follows immediately.

ONGO control statement with Q, F, and P A parameters

alters the default parameters for all subsequent GO statements by deleting the LPAMAP and EDIT parameters.

GO control statement #1

instructs PRDMP to process the first data set on the volume described by the TAPE DD statement.

NEWDUMP control statement with FILESEQ=2

identifies the second data set to be processed. Since no DDNAME= parameter is specified, PRDMP assumes that the data set resides on the volume described by the TAPE DD statement. FILESEQ=2 specifies that the second data set on the volume should be processed.

GO control statement #2

instructs PRDMP to process the data set described by the NEWDUMP control statement.

NEWDUMP control statement with DDNAME=TODAYDMP

PRDM

identifies the third data set to be processed. DDNAME=TODAYDMP specifies that the data set is the one described by the TODAYDMP DD statment.

ONGO control statement with no parameters

restores the original default parameters for the GO control statement.

GO control statement #3

instructs PRDMP to process the data set described by the last NEWDUMP control statement. The original default parameters will be used.

END statement

terminates PRDMP processing.

Example 6: Editing GTF Trace Data from Buffers in a Dump

This example shows how to edit GTF trace buffers from a dump of main storage.

```
//EDIT              JOB       MSGLEVEL=(1,1)[,REGION=128K]
//STEP1             EXEC      PGM=xMDPRDMP
//SYSPRINT          DD        SYSOUT=A
//PRINTER           DD        SYSOUT=A
//TAPE              DD        UNIT=2400,VOL=SER=DUMP,LABEL=(,NL),
//       DISP=OLD
//SYSUT1            DD        UNIT=SYSDA,SPACE=(2056,(257,10))
//SYSIN             DD        *
         EDIT
         END
/*
```

In this example:

JOB Statement

initiates the job. The REGION= parameter applies to VS2 only.

EXEC statement

invokes HMDPRDMP (VS1 only) or AMDPRDMP (VS2 only).

SYSPRINT DD statement

defines the message data set.

PRINTER DD statement

defines the output data set.

TAPE DD statement

defines the input data set.

SYSUT1 DD statement

defines the PRDMP work data set. Although it is not required unless the input data set is on a direct access volume or a multi-volume tape, it should be included to reduce PRDMP processing time. When it is included, it must specify enough space to contain the entire dump.

SYSIN DD statement

defines the data set containing the PRDMP control statements. The data set follows immediately.

EDIT control statement with no parameters

instructs PRDMP to format and print GTF trace buffers in the input data set, according to the default options SYS and USR=ALL. (Note that user records will not be present in the input dump data set except under the following conditions: GTF was started with MODE=EXT, and the GTRACE macro was being used to write user records in GTF's buffers; GTF terminated abnormally and was dumped before the buffers could be written to the designated external output device.)

END control statement

terminates PRDMP processing.

Example 7: Editing a GTF Trace Data Set

When GTF trace data is recorded in an external data set, you can specify editing of only selected records. This example shows how to edit trace records associated with two specific jobs.

```
//EDIT               JOB       MSGLEVEL=(1,1)[,REGION=128K]
//STEP1              EXEC      PGM=xMDPRDMP,PARM='ER=0'
//SYSPRINT           DD        SYSOUT=A
//PRINTER            DD        SYSOUT=A
//TRACE              DD        UNIT=2400,LABEL=(,NL),VOL=SER=TRACE,
//        DISP=OLD,DCB=(BLKSIZE=2048,BUFNO=10)
//SYSIN              DD        *
          EDIT             DDNAME=TRACE,JOBNAME=X57A
          EDIT             DDNAME=TRACE,JOBNAME=X56B,
                     SIO=IO=(190,191)
          END
/*
```

In this example:

JOB Statement

initiates the job. The REGION= parameter applies to VS2 only.

EXEC Statement

invokes PRDMP and specifies the action that PRDMP should take if a program interruption occurs in a user program.

SYSPRINT DD Statement

defines the message data set.

PRINTER DD Statement

defines the output data set.

TRACE DD Statement

defines the input trace data set. Subparameters of the DCB parameter are used to specify the maximum trace block size and to request that ten input buffers be used to process the trace data.

SYSIN DD Statement

defines the data set containing the PRDMP control statements. The data set follows immediately.

EDIT Control Statement #1

instructs PRDMP to edit trace records in the data set defined by the TRACE DD statement. The JOBNAME=X57A parameter requests editing for only those records associated with job X57A.

EDIT Control Statement #2

instructs PRDMP to edit trace records from the data set defined by the TRACE DD statement; that is, the same data set referred to in the first EDIT statement. This time, however, only records associated with job X56B are to be processed; of those, only SIO and I/O interrupt traces for devices 190 and 191 are edited.

END Control Statement

terminates PRDMP processing.

Example 8: Processing a Multi-Volume Page Data Set Dump (VS2 Only)

This example shows how to process an input data set that spans two tape volumes. In this case, the input data set is an AMDSADMP high-speed dump that includes the page data set.

```
//PAGEPRT      JOB  MSGLEVEL=(1,1),REGION=128K
//STEP         EXEC      PROC=PRDMP
//TAPE         DD        UNIT=2400,VOL=SER=(TAPE1,TAPE2),LABEL=(,NL),
//        DISP=OLD
//SYSIN        DD        *
          GO
          PRINT PAGE=131=(020032,020033)
          PRINT PAGE=02=(010002,030001)
          END
/*
```

In This example:

EXEC Statement

calls the cataloged procedure to execute AMDPRDMP, and specifies a region size of 128K.

TAPE DD Statement

defines the input dump data set, which is contained on two tape volumes.

SYSIN DD Statement

defines the data set containing the PRDMP control statements.

GO Control Statement

requests PRDMP to process the dump data set using the default parameters, which are QCBTRACE, LPAMAP, FORMAT, EDIT, PRINT ALL.

PRINT PAGE Control Statmeent #1

requests PRDMP to print the range of pages from the SYS1.PAGE data set represented by slot group numbers 020032 through 010033. The SYS1.PAGE data set resided on the device whose address is 131 when the dump was taken.

PRINT PAGE Control Statmeent #2

requests PRDMP to print two 4K pages of storage represented by slot group numbers 010002 and 040001, respectively. When the dump was taken, the page data set resided on the device represented by relative device number 02.

END Control Statement

terminates processing.

Example 9: Processing a High-Speed SADMP Dump that Includes the Page Data Set (VS2 Only)

This example shows how to print pages from the page data set and format TSO control blocks, when input is a high-speed dump taken by AMDSADMP.

```
//FMTTSO2      JOB        MSGLEVEL=(1,1),REGION=128K
//STEP         EXEC       PGM=AMDPRDMP
//SYSPRINT     DD         SYSOUT=A
//PRINTER      DD         SYSOUT=A,DCB=(BLKSIZE=1210)
//TAPE         DD         UNIT=2400,VOL=SER=SADUMP,LABEL=(,NL),DISP=OLD
//SYSUT1       DD         UNIT=SYSDA,SPACE=(2056,(257,10))
//SYSTSO       DD         UNIT=SYSDA,SPACE=(2056,(100,10))
//SYSIN        DD         *
         FORMAT
         PRINT PAGE=1=(030020,010005,040025)
         TSO SYSTEM=USER,USER=FORMAT
/*
```

In this example:

EXEC Statement

invokes AMDPRDMP.

SYSPRINT DD Statmenet

defines the message data set.

PRINTER DD Statement

defines the output data set.

SYSUT1 DD Statement

defines the PRDMP work data set, This data set will contain only the virtual storage dump spooled from the data set defined by the TAPE DD statement; the page data set dumps are not spooled.

SYSTSO DD Statement

defines the work data set that PRDMP uses when processing TSO user control blocks and storage. This data set must contain enough space to hold the Time Sharing Task's region, the paged-in TSO users'regions and LSQA, and the paged-out TSO users' modified LSQA pages.

SYSIN DD Statment

defines the data set containing the PRDMP control statements.

FORMAT Control Statment

instructs PRDMP to format the following system data areas from the dumped system.

Task Control Blocks (TCBs)
Request Blocks (RBs)
Problem Program Boundaries
Load List
Job Pack Queue
Data Extent Blocks (DEBs)
Task Input/Output Blocks (TIOTs)

PRINT PAGE Control Statmenet

instructs PRDMP to print pages of storage from the page data set represented by the relative device number 1. The pages to be dumped are represented by slot group numbers 030020, 010005, and 040025.

TSO Control Statment

instructs PRDMP to format TSO data areas. SYSTEM=USER requests that only TSO system data areas associated with TSO users be formatted. USER=FORMAT requests that all data areas associated with TSO users be formatted.

Chapter 6: PTFLE

Application function: Applies PTF by generating input to the linkage editor, then invoking the linkage editor. Generate function: Generates JCL and control statements needed to apply PTFs or ICRs in a later step.

Contents

Figures

Introduction

PTFLE is a problem program that you can use to update an operating system without performing another system generation. It has two different, but related, functions:

The generate function produces a job stream which, when executed, will update an operating system by replacing existing load modules with new load modules consisting of PTFs (program temporary fixes) or ICRs (independent component releases). Note that the generate function does not actually apply PTFs or ICRs; it only produces a job stream, which you must then execute.

The application function updates an operating system by replacing existing load modules with new load modules containing PTFs (program temporary fixes). It does this in a single operation, by generating control statements and dynamically invoking the linkage editor. Note that the application function cannot be used to apply ICRs (independent component releases) that require an assembly to be performed before invoking the linkage editor.

This chapter tells how to use both functions.

Application Function

In the application function, PTFLE produces control statements needed to apply PTFs, and invokes the linkage editor to apply the PTFs, all in one operation. The application function requires the following input:

- JCL to invoke the program HMAPTFLE or AMAPTFLE. IBM provides a cataloged procedure called PTFLE that includes most of the required JCL.
- A PTFLE control statement for each CSECT to be updated with a PTF.
- An object deck for each PTF to be applied.
- An IDENTIFY statement to flag each changed CSECT.
- The Stage I SYSGEN output from the generation of the operating system to be updated.

Figure PTFLE-1 shows how PTFLE uses this input to apply PTFs.

The application function requires a partition or region size of 26K, plus the blocksize in bytes for the data set defined by the PCHF DD statement, plus the storage required for the linkage editor.

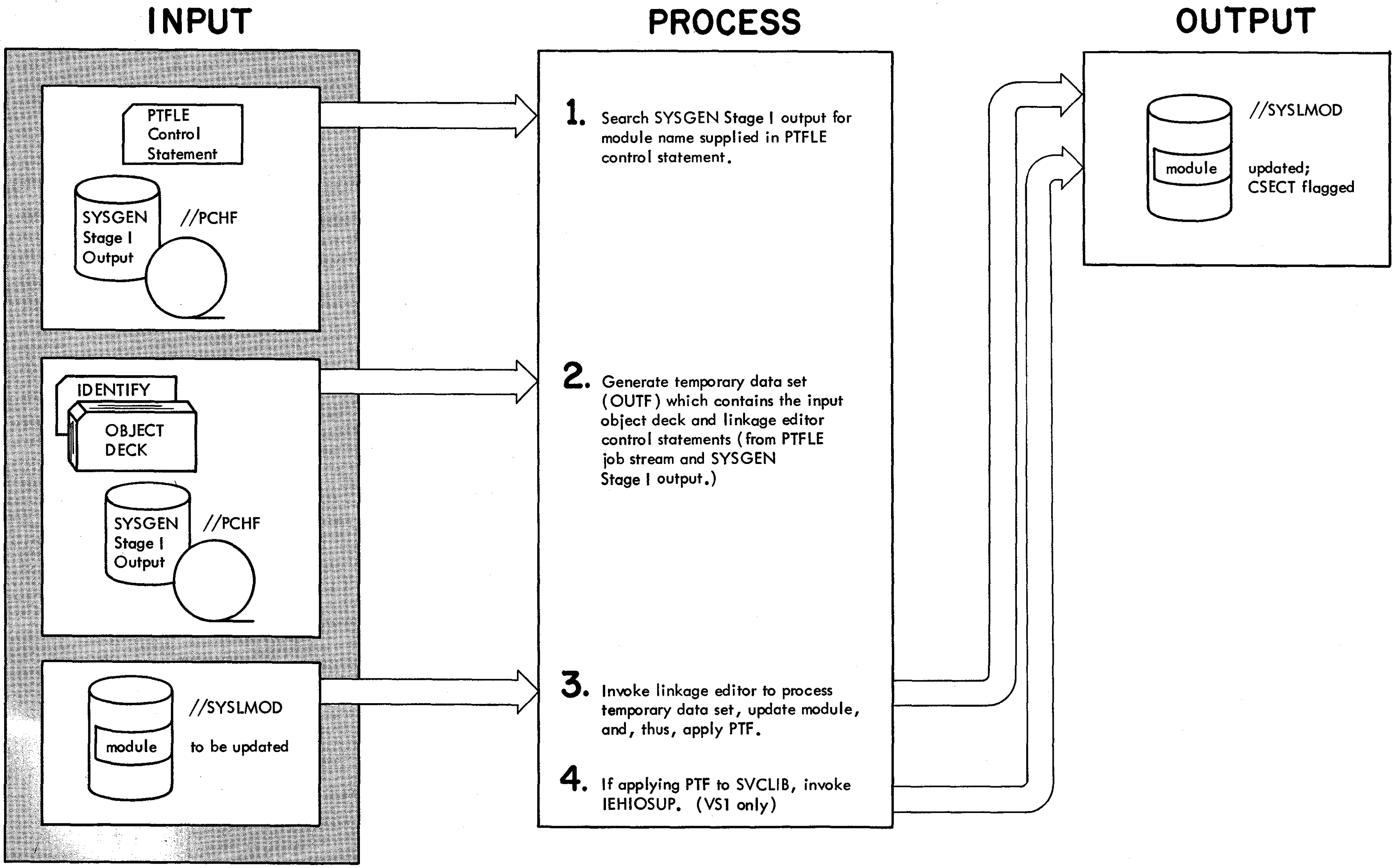

INPUT
PROCESS
OUTPUT
PTFLE Control Statement
SYSGEN Stage I Output
//PCHF
IDENTIFY
OBJECT DECK
SYSGEN Stage I Output
//PCHF
module
//SYSLMOD
to be updated
1. Search SYSGEN Stage I output for module name supplied in PTFLE control statement.
2. Generate temporary data set (OUTF) which contains the input object deck and linkage editor control statements (from PTFLE job stream and SYSGEN Stage I output.)
3. Invoke linkage editor to process temporary data set, update module, and, thus, apply PTF.
4. If applying PTF to SVCLIB, invoke IEHIOSUP. (VS1 only)
//SYSLMOD
module
updated;
CSECT flagged

Figure PTFLE-1. Flow of Processing for the Application Function.

Using the PTFLE Cataloged Procedure

Figure PTFLE-2 shows the PTFLE cataloged procedure. This cataloged procedure assumes that all system libraries are cataloged and that the Stage I output data set consists of unblocked 80-byte records on a non-labeled tape.

```
//                 PROC      USE='IEWL',LIB1=LINKLIB[,REG=68K]
//PTF              EXEC      PGM=xMAPTFLE,PARM=&USE[,REGION=&REG]
//PRINT            DD        SYSOUT=A
//PCHF             DD        UNIT=SYSSQ,LABEL=(,NL),DISP=OLD,
//            VOL=SER=STAG1,DCB=(BLKSIZE=80)
//OUTF             DD        UNIT=SYSDA,SPACE=(TRK,(20,20))
//SYSUT1           DD        UNIT=SYSDA,SPACE=(TRK,(20,20))
//SYSUT2           DD        UNIT=SYSDA,SPACE=(TRK,(20,20))
//SYSPRINT         DD        SYSOUT=A
//SYSLMOD          DD        DSNAME=SYS1.&LIB1,DISP=OLD
[//PARMLIB         DD        DSNAME=SYS1.PARMLIB,DISP=SHR]
```

Figure PTFLE-2. PTFLE Cataloged Procedure -- Application Function Only

PROC statement

assigns default values for symbolic parameters in the EXEC and SYSLMOD DD statements.

EXEC Statement

invokes HMAPTFLE or AMAPTFLE. The PARM= parameter supplies a symbolic name for the linkage editor that PTFLE will use. The default value assigned in the PROC statement is IEWL. The REGION= parameter applies only to VS2.

PRINT DD Statement

defines the message data set for PTFLE.

PCHF DD Statement

defines the Stage I SYSGEN output data set from the generation of the system to be updated. This output must not contain machine control characters.

OUTF DD Statement

defines a temporary sequential data set used by PTFLE and the linkage editor. This data set may reside on either a magnetic tape or direct access volume. Do not attempt to specify the blocksize.

SYSUT1 DD Statement

defines a work data set for the linkage editor. This data set must reside on a direct access device.

SYSUT2 DD Statement

defines a work data set for PTFLE. This data set must reside on a direct access device. Do not attempt to specify the blocksize.

SYSPRINT DD Statement

defines the message data set for the linkage editor.

SYSLMOD DD Statement

defines the library that contains the modules to be updated. The DSNAME= parameter supplies a symbolic name for the library; the default value assigned in the PROC statement is LINKLIB. If you attempt to override the default value, be careful.

PARMLIB DD Statement (VS2 Only)

defines the SYS1.PARMLIB data set which must contain the DSS member IQAORDER. PTFLE requires this statement whenever it must update the nucleus.

Executing the Application Function

Figure PTFLE-3 is an example of a jobstream used to execute the application function of PTFLE.

```
//PTFPROC           JOB       MSGLEVEL=(1,1)
//                  EXEC      PTFLE
//PTF.MODF          DD        *
IEFSD082 01117251 FIRST PTF
         Insert PTF object deck here
    IDENTIFY CSECT1('LEVEL1PTF'),CSECT5('LEVEL3PTF')
IEFSD085 01117251 SAME PTF
         Insert PTF object deck here
    IDENTIFY CSECT10('HERETOO')
/*
```

Figure PTFLE-3. Sample Jobstream for Executing the Application Function of PTFLE.

Application Function Output

When the application function of PTFLE finishes processing, all load modules requiring fixes are updated. No further processing is necessary.

Note, however, that the application function can be used before PTFs have been applied to a distribution library; to avoid having to re-apply a PTF after system generation, be sure you update the distribution libraries with all PTFs applied to the system.

Generate Function

The generate function of PTFLE produces, but does not execute, a job stream needed to apply PTFs (program temporary fixes) and ICRs (independent component releases). The job stream must be executed in a later, separate step.

The generate function requires the following input:

- JCL to invoke the program HMAPTFLE or AMAPTFLE. Since IBM does not provide a cataloged procedure for this purpose, you must supply your own JCL. The next section will show you how to write PTFLE JCL.
- A PTFLE control statement for each CSECT to be updated. (See "Control Statements" in this chapter.)
- (Optional) An IDENTIFY statement to flag each changed CSECT. (See "Control Statements" in this chapter.)
- The Stage I SYSGEN output from the generation of the operating system to be updated.

Note that the generate function does not require a PTF object deck.

The generate function also requires that the distribution libraries be updated to contain all PTFs and ICRs that are to be applied to the system. The distribution libraries are input not to PTFLE, but to the program that executes the JCL produced by PTFLE and applies the PTF and ICRS. Use the linkage editor to include PTFs and ICRs in the distribution libraries; for information about using the linkage editor for this purpose, see the publication OS/VS Linkage Editor and Loader, GC26-3803.

Figure PTFLE-4 shows how the generate function uses this input.

The generate function requires a partition or region size of 47K plus the blocksize in bytes for the data set defined on the PCHF DD statement.

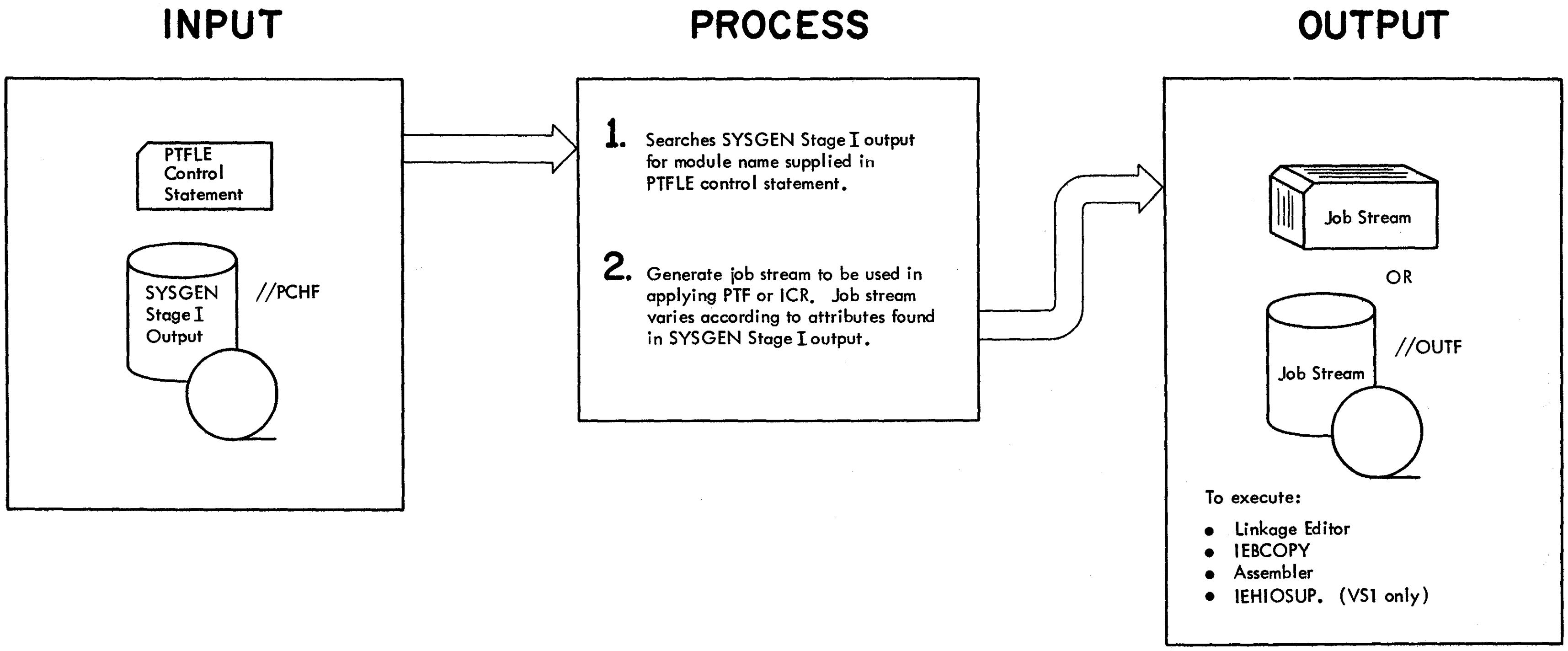

INPUT
PROCESS
OUTPUT
PTFLE Control Statement
SYSGEN Stage I Output
//PCHF
1. Searches SYSGEN Stage I output for module name supplied in PTFLE control statement.
2. Generate job stream to be used in applying PTF or ICR. Job stream varies according to attributes found in SYSGEN Stage I output.
Job Stream
OR
Job Stream
//OUTF
To execute:
• Linkage Editor
• IEBCOPY
• Assembler
• IEHIOSUP. (VS1 only)

Figure PTFLE-4. Flow of Processing for the Generate Function.

Writing JCL for the Generate Function

Figure PTFLE-5 shows the JCL statements needed to execute the generate function of PTFLE.

```
//GENER        JOB      MSGLEVEL=(1,1)[,REGION=64K]
//             EXEC     PGM=xMAPTFLE
//PRINT        DD       SYSOUT=A
//OUTF         DD       UNIT=2400,LABEL=(,NL),
//        DISP=(NEW,KEEP),VOL=SER=OUTPUT
//PCHF         DD       UNIT=2400,LABEL=(,NL),
          DISP=OLD,VOL=SER=SYSGEN,DCB=(LRECL=80,BLKSIZE=80)
//MODF         DD       *
          Insert Control statements here
```

Figure PTFLE-5. Sample JCL Used to Execute PTFLE, Generate Function.

JOB Statement

initiates the job. The REGION= parameter applies only to VS2.

EXEC statement

invokes HMAPTFLE or AMAPTFLE. Do not code any other parameters on this statement.

PRINT DD Statement

defines the PTFLE message data set.

OUTF DD Statement

defines the output data set, which may be directed to a card punch, a direct access device or a tape device. Do not specify a block size.

PCHF DD Statement

defines the Stage I output from the generation of the system to be updated.

MODF DD statement

defines the input stream, which contains control statements.

Executing the Generate Function

Figure PTFLE-6 is an example of a job stream used to execute the generate function of PTFLE.

```
//PTFJCL         JOB      MSGLEVEL=(1,1)[,REGION=64K]
//               EXEC     PGM=xMAPTFLE
//PRINT          DD       SYSOUT=A
//OUTF           DD       UNIT=SYSDA,VOL=SER=OUTPUT,DISP=(NEW,KEEP),
               DSNAME=DAOUTPUT,SPACE=(TRK,(20,10))
//PCHF           DD       UNIT=2314,DISP=OLD,VOL=SER=SYSGEN,
               DCB=(LRECL=80,BLKSIZE=160)
//MODF           DD       *
IEBGEN03 05199133
IEX51    02150191
   IDENTIFY IEX51000('PTF20191')
IGE0000A 03144004
IGE0000D 02155012
/*
```

Notes:

- The REGION= parameter in the JOB statement applies only to VS2.
- The PGM= parameter in the EXEC statement invokes HMAPTFLE in VS1, and AMAPTFLE in VS2.

Figure PTFLE-6. Sample Jobstream for Executing the Generate Function of PTFLE.

Notice that in this example only one module is flagged with an IDENTIFY statement. For the generate function you may omit the IDENTIFY statement; however, the information you supply with the IDENTIFY statement is a valuable diagnostic aid, and it is wise to take full advantage of it.

Generate Function Output

The output of the generate function is a jobstream consisting of JCL and control statements. This jobstream invokes a program, either the linkage editor, the assembler, or IEBCOPY, to update the target module with a PTF or ICR from the distribution library. If the target module was link edited into the operating system during system generation, the linkage editor is invoked to apply the PTF. If the target module was assembled, first the assembler and then the linkage editor is invoked. If the target module was copied, IEBCOPY is invoked.

In VS1 only, PTFLE also generates JCL to invoke IEHIOSUP. This program updates any TTR entries in the transfer control tables of the supervisor call library that may requre a change as a result of updating.

In VS2 only, PTFLE also provides a listing of the job stream that it produces.

Figure deleted because LINKS catalogued procedures are the same for VS1 and VS2. (See Figure PTFLE-7.)

Figure PTFLE-7 is an example of linkage editor-type output produced by the generate function of PTFLE.

```
//SYSGENS     JOB  1,'SYSTEM GENERATION',MSGLEVEL=(1,1)
//SG5    EXEC LINKS,PARM='NCAL,LIST,XREF,OVLY,XCAL,LET',
//  UNIT='2314',SER=SYRES,N=SYS1,NAME=LINKLIB,P1=' ',
//  MOD=,P2=' ',OBJ=OBJPDS,CLASS=A
//AOS04  DD   DISP=SHR,VOLUME=(,RETAIN),DSNAME=SYS1.AOS04
//SYSLIN DD   *
  INCLUDE AOS04(AEWLFMAP)
 ENTRY AEWLFROU
 ALIAS IEWL,AEWL
 ALIAS HEWL,HEWLF064
 ALIAS LINKEDIT
  INCLUDE  SYSLMOD(AEWLF064)
 OVERLAY ONE  *** VALID EXCLUSIVE CALL TO AEWLFINP ***
 INSERT AEWLFINT,AEWLFOPT
 OVERLAY ONE **VALID EXCL. CALLS TO AEWLFADA,IEWLCFNI,AEWLFADA
 INSERT AEWLFINP,AEWLFESD,AEWLFEND,AEWLFSYM,AEWLFRCG
 INSErT AEWLFSCN,AEWLFRAT,AEWLFTXT,AEWLFINC,AEWLFIDR
 OVERLAY ONE
 INSERT AEWLFMAP
 OVERLAY TWO **VALID EXCL. CALLS TO AEWLFFNL,AEWLFSCD,AEWLFFNL
 INSERT AEWLFADA,AEWLFENT,AEWLFENS,AEWLFOUT
 OVERLAY TWO  *** VALID EXCLUSIVE CALL TO AEWLFFNL ***
 INSERT AEWLFREL,AEWLFSCD,AEWLFSIO
 OVERLAY TWO   *** VALID EXCLUSIVE CALL TO IEWLENAM ***
 INSERT AEWLFFNL,AEWLFBTP
  SETSSI   99999999
 NAME AEWLF064(R)
/*
```

Figure PTFLE-7. Linkage Editor JCL and Control Statments Produced by PTFLE (Generate Function)

Figure PTFLE-8 shows an example of IEBCOPY-type output produced by the generate function of PTFLE.

```
//SG44        EXEC      PGM=IEBCOPY,COND=(8,LT)
//SYSUT3      DD        DISP=SHR,DSNAME=SYS1.UT3
//SYSPRINT    DD        SPACE=(121,(500,100),RLSE),
//       DCB=(RECFM=FB,LRECL=121,BLKSIZE=121),
//       SYSOUT=A
//CI505       DD        DISP=SHR,VOLUME=(,RETAIN),DSNAME=SYS1.CI505
//SVCLIB      DD        DSNAME=SYS1.SVCLIB,VOLUME=(,RETAIN,SER=SYSRES),
//       UNIT=2314,DISP=OLD
//SYSIN       DD        *
    COPY OUTDD=SVCLIB,INDD=CI505
    SELECT MEMBER=((IGE0000A,,R))
    SELECT MEMBER=((IGE0000D,,R))
    SELECT MEMBER=((IGE0000G,,R))
/*
```

Figure PTFLE-8. IEBCOPY JCL and Control Statement Produced by PTFLE (Generate Function)

Figure deleted because ASMS catalogued procedures are the same for VS1 and VS2. (See Figure PTFLE-9.)

Figure PTFLE-9 is an example of Assembler and Linkage Editor output produced by the generate function of PTFLE.

```
//SYSGENS      JOB  1,'SYSTEM GENERATION',MSGLEVEL-(1,1)
//SG8     EXEC ASMS,OBJ=OBJPDS,MOD=DCM009,CLASS=A
//SYSIN   DD   *
          PRINT ON,NODATA
DCM009    CSECT
          IEECDCM DEVICE=,USE=FC
          END
/*
//SG2     EXEC LINKS,PARM='NCAL,LIST,XREF',
//  UNIT='2314',SER=SYSRES,N=SYS,NAME=LPALIB,P1=' ',
//  MOD=,P2=' ',OBJ=OBJPDS,CLASS=A
//SYSLIN DD   *
  INCLUDE SYSPUnCH(DCM009)
  INCLUDE  SYSLMOD(DCM009)
  SETSSI   33333333
  NAME DCM009(R)
/*
```

Figure PTFLE-9. Example of Assembler and Linkage Editor Output Produced by PTFLE (Generate Function)

PTFL

Figure PTFLE-10 is an example of IEHIOSUP output produced by the generate function of PTFLE, in VS1 only. In VS1, IEHIOSUP is always invoked as part of PTFLE output, whether or not the SVC library requires updating.

```
//SG79         EXEC       PGM=IEHIOSUP
//SYSPRINT     DD         SPACE=(121,(500,1000),RLSE),
//        DCB=(RECFM=FB,LRECL=121,BLKSIZE=121),SYSOUT=A
//SYSUT1       DD         DSNAME=SYS1.SVCLIB,DISP=(OLD,PASS),
//        VOLUME=(,RETAIN,SER=111111),UNIT=2311
/*
```

Figure PTFLE-10. IEHIOSUP JCL Produced by PTFLE in VS1 only (Generate Function)

Control Statements

Both functions of PTFLE require a PTFLE control statement for each module to be updated. The application function also requires a Linkage Editor IDENTIFY statement for each module. The IDENTIFY statement is optional in the generate function. The following sections describe how to code these control statements.

PTFLE Control Statement

The PTFLE control statement consists simply of a module name from 1 to 8 characters long, an 8-character system status information (SSI) number, and any comments you may wish to add. The module name must begin in column 1 and be followed by one or more blanks. The SSI number must begin in column 10 and be followed by one or more blanks. Only blanks may be inserted between the module name and the SSI number. Here are several examples of PTFLE control statements coded correctly:

```
IEBGEN04 05199134 THIS IS a MODULE TO BE UPDATED

IEBGEN05 05199135 THIS IS ANOTHER MODULE TO BE UPDATED

MYMOD    06123487 THIS MODULE NAME HAS ONLY FIVE CHARACTERS

MOD1     06134567 NOTICE THAT THE MODULE NAME CAN BE 1 TO 8 CHARACTERS

MOD2     06145678 THE SSI NUMBER HOWEVER MUST ALWAYS START IN COLUMN 10
```

Module Name Parameter

You must supply a PTFLE control statement for each module that you want to update. For modules that have alias names and that were copied rather than link edited during system generation, you must supply a separate control statement for each alias name. Alias name control statements need not contain SSI numbers. Here is an example of control statements defining a single module with many alias names.

```
MODULE22 05167788 THIS IS THE TRUE MODULE NAME
ALIAS1
ALIAS2
ALIAS3
ALIAS4
```

In any one execution of PTFLE, you may include up to 150 control statements. For the Generate function, you must count all alias statements toward this maximum.

If any module to be updated has both a component library name and a system library name, include only the component library name in a PTFLE control statement.

With one exception, you can use PTFLE to update a module whose name in the distribution library differs from the CSECT name in the module. The exception is any module which was link edited rather than copied during system generation and whose overlay structure was defined using INCLUDE statements rather than INSERT statements. The FORTRAN H Compiler is an example of such a module.

Modules copied from the distribution library during system generation may be updated using PTFLE provided the SELECT statement was used in the copy operation.

SSI Number Parameter

The number you specify in the SSI field of a PTFLE control statement should be the number that is listed under the heading "Status Info" on the PTF cover letter. This number will be placed in the library directory entry for the updated module to indicate that the PTF was changed. If you omit the SSI field from a control statement containing a true module name, the SSI field in the module will be set to zeroes; you can, however, omit the SSI field from alias control statements without altering the SSI.

IDENTIFY Control Statement

The IDENTIFY control statement allows you to flag the specific CSECT within a module that is to be updated with a PTF or ICR. PTFLE does not use the IDENTIFY statement directly, but passes it to the linkage editor for processing. For the application function, you must include an IDENTIFY statement for each module that is to be updated; if you omit the IDENTIFY statement for one module, PTFLE wiil issue an error message and terminate processing. For the generate function, the IDENTIFY control statement is optional.

Code the IDENTIFY statement according to the following rules:

- Always begin the IDENTIFY statement in or after column 2.
- You may specify as many as 40 characters of identifying information for each CSECT name.
- To continue the IDENTIFY statement close the first card with a delimiting comma and a nonblank character in column 72, and start the next card in column 16. Note, however, that PTFLE allows a maximum of 150 IDENTIFY statements in a single execution, and all IDENTIFY continuation statements must be counted toward this total.

Here are some examples of IDENTIFY control statements:

```
 IDENTIFY MYCSECT('PTF41392547'),YOURCSCT('PTF12345678')

 IDENTIFY CSECT1('***THIS IS A 40 CHARACTER IDENTIFIER****')

    IDENTIFY     CSECT2('PTF1'),CSECT3('PTF2'),CSECT4('PTF3')

 IDENTIFY CSECT1('PTFA'),CSECT2('PTFB'),CSECT3('PTFC'),CSECT4('PTFD'), x

               CSECT5('PTFE'),CSECT6('PTFF')
```

Chapter 7: SADMP

Operates as a stand-alone program to produce a high-speed or low-speed dump of real storage. The high-speed version also dumps the page data set.

Contents

Figures

Introduction

SADMP is a stand-alone program that can operate at high speed or low speed to produce a dump of real storage. The high-speed version of SADMP can also dump the page data set.

Low-speed SADMP output can be directed either to a printer or to a tape volume, from which you can print it using PRDMP or IEBPTPCH. High-speed SADMP output must be written to a tape volume, from which you can format it and print it using PRDMP.

SADMP is supplied as a macro definition in the system library SYS1.MACLIB. To get from this macro definition to the executable stand-alone dump program, you must code and assemble a macro instruction and initialize a residence volume with the resulting job stream. The following is a summary of the steps you must take to generate and execute the SADMP stand-alone dump program:

1. Code the HMDSADMP (for VS1) or the AMDSADMP (for VS2) macro instruction to define the type of dump program you want.

2. Assemble the HMDSADMP or AMDSADMP macro instruction. Output from this step is JCL and control statements needed to create the stand-alone dump program and place it on the residence volume.

3. Initialize the SADMP residence volume by executing the job stream produced in the previous step. Output from this step is the SADMP program in executable form.

4. Execute the SADMP stand-alone program.

Notice that steps 1, 2, and 3 can all be performed under the operating system. Step 4, on the other hand, is a stand-alone operation.

Coding the Macro Instruction

The SADMP program has four basic variations:

- High-speed, residing on a direct access device, with output directed to a tape volume.

- High-speed, residing on a tape device, with output directed to a tape volume.

- Low-speed, residing on a direct access device, with output directed to a tape volume

- Low-speed, residing on a direct access device, with output directed to a printer.

The following sections describe how to code the HMDSADMP or AMDSADMP macro instruction to produce these four versions of the dump program.

High-Speed, Direct Access Resident

Figure SADMP-1 shows how to code the HMDSADMP (for VS1) or AMDSADMP (for VS2) macro instruction to produce a high-speed dump program residing on a direct access volume.

```
[symbol] {HMDSADMP}  [TYPE=HI][,IPL=Dunit][,VOLSER=volser ]
         {AMDSADMP}

                    [ULABEL={PURGE  }][,CONSOLE=(devaddr,devtyp)]
                            {NOPURGE}

                    [,SYSUT= devtyp ][,OUTPUT=Tunit]
```

Figure SADMP-1. Format of HMDSADMP or AMDSADMP Macro Instruction Used to Generate a High-Speed, Direct Access Resident Dump Program.

symbol

an arbitrary name you can assign to the HMDSADMP or AMDSADMP macro instruction. SADMP will use this symbol to create a jobname for use in the initialization step.

HMDSADMP
AMDSADMP

the name of the macro instruction. Use HMDSADMP for VS1 and AMDSADMP for VS2.

TYPE=HI

specifies the high-speed version of the dump program. If you omit this parameter, TYPE=HI is assumed as the default.

IPL=Dunit

specifies the unit address (for example, IPL=D131) or the device type (for example, IPL=D2305-2) of the device on which the dump program should reside during the initialization stage. The dump program need not reside on the same unit after initialization. If you omit this parameter, IPL=D3330 is assumed as the default.

VOLSER=volser

identifies a specific direct access volume on which the dump program should reside. If you omit this parameter, VOLSER=SADUMP is assumed as the default.

ULABEL=$\left\{\begin{array}{l}\text{NOPURGE}\\ \text{PURGE}\end{array}\right\}$

Specifies whether existing user labels on the specified residence volume should be deleted (PURGE) or retained (NOPURGE). If you specify NOPURGE, the SADMP program will be written on cylinder 0 track 0 of the residence volume, immediately following all user labels. If the user labels occupy so much space that the SADMP program will not fit on track 0, the initialization program will issue an error message and terminate.

If you omit this parameter, ULABEL=NOPURGE will be assumed as the default. Note that you must specify ULABEL=PURGE if the residence volume is a 2314 volume that contains user labels.

CONSOLE=(devaddr,devtyp)

specifies the device address and device type of the primary system console. The following device types are valid:

1052

2150

3066 (vs2 only)

3210

3215

If you omit this parameter, CONSOLE=(01F,3215) is assumed as the default.

SYSUT= devtyp

specifies the type of device to be used for workfiles during the initialization stage. The device may be specified as a group name (for example, SYSDA), a device type (for example, 3330), or a unit address (for example, 131). If you omit this parameter, SYSUT=SYSDA will be assumed as the default.

OUTPUT=Tunit

specifies the unit address of the output device. High-speed dump output must always be directed to a tape device. Tape output is always written at the highest density of the tape drive. If you omit this parameter, OUTPUT=T282 is assumed as the default.

High-Speed, Tape Resident

Figure SADMP-2 shows how to code the HMDSADMP or AMDSADMP macro instruction to produce a high-speed dump program residing on a tape volume.

```
[symbol] {HMDSADMP}  [TYPE=HI][,IPL=Tunit][,VOLSER= volser ]
         {AMDSADMP}

                   [,CONSOLE=(devaddr,devtyp)][,SYSUT= devtyp ]

                   [,OUTPUT=Tunit]
```

Figure SADMP-2. Format of HMDSADMP or AMDSADMP Macro Instruction Used to Generate a High-Speed, Tape Resident Dump Program.

symbol

an arbitrary name you can assign to the HMDSADMP or AMDSADMP macro instruction. SADMP will use this symbol to create a jobname for use in the initialization step.

{HMDSADMP}
{AMDSADMP}

the name of the macro instruction. Use HMDSADMP for VS1 and AMDSADMP for VS2.

TYPE=HI

specifies the high-speed version of the dump program. If you omit this parameter, TYPE=HI is assumed as the default.

IPL=Tunit

specifies the unit address (for example, IPL=T282) or the device type (for example, IPL=2400 or IPL=T2400-3) of the device on which the dump program should reside during the initialization stage. The dump program need not reside on the same unit after initialization. If you omit this parameter, IPL=D3330 is assumed as the default.

VOLSER=volser

identifies a specific tape volume on which the SADMP program should reside. If you omit this parameter, VOLSER=SADUMP is assumed as the default.

Note that you must include this parameter unless you have a specific volume named SADUMP reserved as the SADUMP residence volume.

CONSOLE=(devaddr,devtyp)

specifies the device address and device type of the primary system console. The following device types are valid:

1052

2150

3066 (VS2 Only)

3210

3215

If you omit this parameter, CONSOLE=(01F,3215) is assumed as the default.

SYSUT=devtyp

specifies the type of device to be used for workfiles during the initialization stage. The device may be specified as a group name (for example, SYSDA), a device type (for example, 2314), or a unit address (for example, 131). If you omit this parameter, SYSUT=SYSDA will be assumed as the default.

OUTPUT=Tunit

specifies the unit address of the output device. High-speed dump output must always be directed to a tape device. The output is always written at the highest density of the tape drive. If you omit this parameter, OUTPUT=T282 is assumed as the default.

Low-Speed, Output to Tape

Figure SADMP-3 shows how to code the HMDSADMP or AMDSADMP macro instruction to produce a low-speed dump program whose output is directed to a tape volume.

```
[symbol] {HMDSADMP}  [TYPE=LO][,IPL=Dunit][,OUTPUT=Tunit
         {AMDSADMP}

                   [,VOLSER= volser ][,CONSOLE=(devaddr,devtyp)]

                   [,SYSUT= devtyp ][,ULABEL={NOPURGE}]
                                            {PURGE  }
```

Figure SADMP-3. Format of HMDSADMP or AMDSADMP Macro Instruction Used to Produce a Low-Speed Dump Program with Output Directed to Tape.

symbol

an arbitrary name you can assign to the HMDSADMP or AMDSADMP macro instruction. SADMP will use this symbol to create a jobname for use in the initialization step.

{HMDSADMP}
{AMDSADMP}

the name of the macro instruction. Use HMDSADMP for VS1 and AMDSADMP for VS2.

TYPE=LO

specifies the low-speed version of the dump program. If you omit this parameter, TYPE=HI is assumed as the default.

IPL=Dunit

specifies the unit address (for example, IPL=D151) or the device type (for example, IPL=D2314) of the device on which the dump program should reside during the initialization stage. The dump program need not reside on the same unit after initialization. If you omit this parameter, IPL=D3330 is assumed as the default.

OUTPUT=Tunit

specifies the tape device to which SADMP output should be written.

Tape output is always written at the highest density of the tape drive. If you omit this parameter, OUTPUT=P00E (that is, a printer) will be assumed as the default.

VOLSER= volser

identifies a specific direct access volume on which the SADMP program should reside. If you omit this parameter, VOLSER=SADUMP is assumed as the default.

CONSOLE=(devaddr,devtyp)

specifies the device address and device type of the primary system console. The following device types are valid:

1052

2150

3066 (VS2 only)

3210

3215

If you omit this parameter, CONSOLE=(01F,3215) is assumed as the default.

SYSUT= devtyp

specifies the type of device to be used for workfiles during the initialization stage. The device may be specified as a group name (for example, SYSDA), a device type (for example, 3330), or a unit address (for example, 131). If you omit this parameter, SYSUT=SYSDA will be assumed as the default.

ULABEL=$\left\{\begin{array}{l}\text{NOPURGE}\\ \text{PURGE}\end{array}\right\}$

Specifies whether existing user labels on the specified residence volume should be deleted (PURGE) or retained (NOPURGE). If you specify NOPURGE, the SADMP program will be written on cylinder 0 track 0 of the residence volume, immediately following all user labels. If the user labels occupy so much space that the SADMP program will not fit on track 0, the initialization program will issue an error message and terminate.

If you omit this parameter, ULABEL=NOPURGE will be assumed as the default. Note that you must specify ULABEL=PURGE if the residence volume is a 2314 volume that contains user labels.

Low-Speed, Output Directed to a Printer

Figure SADMP-4 shows how to code the HMDSADMP or AMDSADMP macro instruction to produce a low-speed dump program whose output is directed to a printer.

```
[symbol] /HMDSADMP\  [TYPE=LO][,IPL=Dunit][,OUTPUT=Punit]
         \AMDSADMP/

                    [,VOLSER= volser][,CONSOLE=(devaddr,devtype)]

                    [,SYSUT=devtyp][,ULABEL=/NOPURGE\]
                                            \ PURGE /
```

Figure SADMP-4. Format of HMDSADMP or AMDSADMP Macro instruction Used to Generate a Low-Speed Dump Program with Output Directed to a Printer.

symbol

an arbitrary name you can assign to the HMDSADMP or AMDSADMP macro instruction. SADMP will use this symbol to create a jobname for use in the initialization step.

```
/HMDSADMP\
\AMDSADMP/
```

the name of the macro instruction. Use HMDSADMP for VS1 and AMDSADMP for VS2.

TYPE=LO

specifies the low-speed version of the dump program. If you omit this parameter, TYPE=HI is assumed as the default.

IPL=Dunit

specifies the unit address (for example, IPL=D151) or the device type (for example,IPL=D2314) of the device on which the dump program shoujld reside during the initialization stage. The dump program need not reside on the same unit after initialization. If you omit this parameter, IPL=D3330 is assumed as the default.

SADM

OUTPUT=Punit

specifies the printer device to which SADMP output should be written. If you omit this parameter, OUTPUT=P00E will be assumed as the default.

VOLSER=volser

identifies a specific direct access volume on which the SADMP program should reside. If you omit this parameter, VOLSER=SADUMP is assumed as the default.

CONSOLE=(devaddr,devtyp)

specifies the device address and device type of the primary system console. The following device types are valid:

1052

2150

3066 (VS2 Only)

3210

3215

If you omit this parameter, CONSOLE=(01F,3215) is assumed as the default.

SYSUT=devtyp

specifies the type of device to be used for workfiles during the initialization stage. The device may be specified as a group name (for example, SYSDA), a device type (for example, 3330), or a unit address (for example, 131). If you omit this parameter, SYSUT=SYSDA will be assumed as the default.

ULABEL= $\left\{\begin{matrix}\text{NOPURGE}\\\text{PURGE}\end{matrix}\right\}$

specifies whether existing user labels on the specified residence volume should be deleted (PURGE) or retained (NOPURGE). If you specify NOPURGE, the SADMP program will be written on cylinder 0 track 0 of the residence volume, immediately following all user labels. If the user labels occupy so much space that the SADMP program will not fit on track 0, the initialization program will issue an error message and terminate.

If you omit this parameter, ULABEL=NOPURGE will be assumed as the default. Note that you must specify ULABEL=PURGE if the residence volume is a 2314 volume that contains user labels.

Assembling the Macro Instruction

The next step in generating the stand-alone dump program is assembling the macro instruction. Figure SADMP-5 is an example of the JCL statements needed for this operation. This example assumes ASMFC as the standard IBM-supplied cataloged procedure for invoking an assembler.

```
//ASMSADMP         JOB        MSGLEVEL=(1,1)
//            EXEC      ASMFC,PARM.ASM='DECK'
//ASM.SYSIN        DD         *
         HMDSADMP  TYPE=HI
         END
/*
```

Note that this example shows how to assemble the VS1 macro instruction, HMDSADMP. The JCL needed to assemble the VS2 macro instruction is identical except that the macro instruction is named AMDSADMP.

Figure SADMP-5. Sample JCL Needed to Assemble the HMDSADMP or AMDSADMP Macro Instruction

JOB statement

initiates the job.

EXEC statement

invokes the cataloged procedure ASMFC, which does the following:

- Invokes an assembler.
- Identifies the system macro library (SYS1.MACLIB), which contains the HMDSADMP or AMDSADMP macro definition.
- Defines work data sets for the assembler's use.
- Defines two output data sets (SYSPRINT and SYSPUNCH).

The EXEC statement also requests that the assembler output be punched as a deck.

ASM.SYSIN DD statement

defines the input stream, which in this case consists of the HMDSADMP or AMDSADMP macro instruction and an END control statement.

Output from this assembly is an object deck and a listing of the statements in the deck. The deck contains JCL and control statements; these constitute a job stream that creates the stand-alone dump program and initializes it on a tape or direct access volume.

The output listing may also contain error messages, which describe errors that you may have made in specifying the HMDSADMP or AMDSADMP macro instruction. To respond to one of these messages, check your specification of the macro instruction and run the assembly step again.

IPL=&IPL IS INVALID, IPL=D3330 IS ASSUMED

Explanation: The IPL operand is invalid. It is greater than 7 characters, or less than 4 characters, or not prefixed with a "T" or a "D".

Severity Code: 4.

CONSOLE ADDR=&CONSOLE (1) IS INVALID, CONSOLE ADDR=01F IS ASSUMED

Explanation: The console address operand is not three characters.

Severity Code: 4.

CONSOLE TYPE=& CONSOLE (2) IS INVALID, CONSOLE TYPE =3215 IS ASSUMED

Explanation: An invalid console type was specified. Only 1052,2150,3210, and 3215 are acceptable. (in VS2, 3066 is also a valid console.) The length of the console type was not equal to 4.

Severity Code: 4.

TYPE=&TYPE IS INVALID, TYPE=HI IS ASSUMED

Explanation: Type operand must be HI or LO.

Severity Code: 4.

OUTPUT=&OUTPUT IS INVALID, OUTPUT=P00E IS ASSUMED

Explanation: For TYPE=LO the output address was not prefixed with a "T" or "P" or the address was not a 3 character address.

Severity Code: 4.

PARAMETERS IPL=&IPL2 AND TYPE=&TYPE ARE INCOMPATIBLE MACRO PROCESSING TERMINATED

Explanation: IPL=Txxx and TYPE=LO are incompatible. A LO speed dump may only reside on direct access device.

Severity Code: 8.

OUTPUT=&OUTPUT IS INVALID, OUTPUT=T282 IS ASSUMED

Explanation: For TYPE=HI the output address was not prefixed by a "T" or the address was not a 3-character address.

Severity Code: 4.

SYSUT=&SYSUT IS INVALID, SYSUT=SYSDA IS ASSUMED

Explanation: The SYSUT operand exceeds 6 characters.

Severity Code: 4.

VOLSER=&VOLSER IS INVALID, VOLSER=SADUMP IS ASSUMED

Explanation: The VOLSER operand exceeds 6 characters.

Severity Code: 4.

ULABEL=&ULABEL IS INVALID, ULABEL=NOPURGE IS ASSUMED

Explanation: The ULABEL operand is not PURGE or NOPURGE.

Severity Code: 4.

Directing Assembly Output to Tape or Direct Access

You can override the cataloged procedure ASMFC to direct the object module output from the assembly to a tape or direct access volume. To direct output to tape, add the following statement to the JCL shown in Figure SADMP-5:

```
//ASM.SYSPUNCH DD UNIT=2400,LABEL=(,NL),DISP=(NEW,KEEP),VOL=SER=SCRTCH
```

To write the output on a direct access device, use the following statement:

```
//ASM.SYSPUNCH DD UNIT=SYSDA,SPACE=(TRK,(2,1)),DSN=DMPPACK,

//        DISP=(NEW,KEEP),VOL=SER=SCRTCH
```

Assembling Multiple Macro Instructions

If you anticipate need for more than one version of the stand-alone dump program in your installation, you can save time by assembling all applicable variations of the HMDSADMP or AMDSADMP macro instruction in the same step. Differentiate between the versions by coding a unique symbol at the beginning of each macro instruction. SADMP will use the symbol you code to create a jobname for the initialization program.

Here is an example of a job stream used to assemble four versions of the HMDSADMP or AMDSADMP macro instruction. Note that you must specify a different residence volume for each program you generate.

```
//ASMSADMP         JOB        MSGLEVEL=(1,1)
//                 EXEC       ASMFC
//ASM.SYSIN        DD         *
HITAPE    HMDSADMP IPL=T2400,VOLSER=SADMP1
HIDISK    HMDSADMP VOLSER=SADMP2
LOTAPE    HMDSADMP TYPE=LO,OUTPUT=T282,VOLSER=SADMP3
LOPTR     HMDSADMP TYPE=LO,VOLSER=SADMP4
          END
/*
```

Note that this example shows how to assemble more than one VS1 macro instruction, HMDSADMP. To assemble multiple macro instructions in VS2, substitute AMDSADMP for HMDSADMP.

Initializing the Residence Volume

To initialize the SADMP residence volume, make sure the residence volume is properly prepared, and execute the job stream produced in the previous (assembly) step. When execution is complete, the SADMP program is ready to use at any time.

You must also make sure that the SADMP residence device does not contain a SYS1.PAGEDUMP data set if you are generating a high-speed, direct access resident dump program. If SADMP finds such a data set on the device to be initialized as the residence device, initialization will terminate.

Physical output from the initialization step is a listing, which may contain the following error messages. To respond to one of these messages, make sure that the input to the assembly step, output of the assembly step, and input to the initialization step are all correct and that all three correspond. Then run the initialization step again.

TYPE2=&TYPE2 INVALID; MACRO PROCESSING TERMINATED

Explanation: The TYPE2 operand is not HI or LO.

Severity Code: 12.

OUTPUT2=&OUTPUT2 FOR TYPE=&TYPE2 INVALID; MACRO PROCESSING TERMINATED

Explanation: For TYPE2=HI OUTPUT2=Pxxx was specified. OUTPUT2 must be Txxx for HI dumps.

Severity Code: 12.

OUTPUT2=&OUTPUT2 INVALID; MACRO PROCESSING TERMINATED

Explanation: For TYPE2=HI the OUTPUT2 operand is not of the form Txxx. For Type2=LO the OUTPUT2 operand is not of the form Txxx or Pxxx.

Severity Code: 12.

CONADDR=&CONADDR INVALID; MACRO PROCESSING TERMINATED

Explanation: The CONADDR operand is not three characters.

Severity Code: 12.

CONTYPE=&CONTYPE INVALID; MACRO PROCESSING TERMINATED

Explanation: An invalid console type was specified. Only 1052, 2150, 3210,and 3215 are acceptable. (in VS2, 3066 is also a valid console.)

Severity Code: 12.

IPL=&IPL2 INVALID; MACRO PROCESSING TERMINATED

Explanation: The IPL2 operand is invalid, it must be "D" or "T".

Severity Code: 12.

IPL2=&IPL2 AND TYPE2=&TYPE2 INCOMPATIBLE; MACRO PROCESSING TERMINATED

Explanation: IPL2=Txxx and TYPE2=LO are incompatible. A LO speed dump must reside on a direct access device.

Severity Code: 12.

SADM

Executing the Stand-Alone Dump Program

Whenever you need to use the stand-alone dump program, follow this procedure:

- Let system activitiy come to a halt.
- IMPORTANT: Perform the STORE STATUS operation as described in the System/370 Operating Procedures manual for your model.
- Mount the volume that contains the SADMP program and ready the device. (IMPORTANT: If IPL= Tunit or OUTPUT= Tunit, make sure the file protect ring it in place on the tape volumes.)
- Set the Load Unit dials on the system control panel to the address of the device where the SADMP volume is mounted.
- Press the LOAD button.

Notes:

- When you are dumping the page data set, do not be concerned if the output tape stops periodically during execution. This is due to channel contention between the input and output devices. To avoid channel contention and ensure fast operation, make sure your input and output devices are on different channels.
- SADMP execution will be unpredictable if any device from which SADMP is reading for writing data is shared by another CPU. To avoid this problem, stop the other CPU(s) or disable that device on the other CPU(s) while running SADMP.

Operator Communication During Execution

As soon as the Stand-alone dump program begins processing, SADMP may begin to send messages; you must reply to these messages before processing can continue. The nature of the messages in some cases depends on the version of the program that is being executed.

Note that if the console is unavailable, the dump program will bypass operator communication and attempt to dump real storage to the unit address specified in the HMDSADMP or AMDSADMP macro instruction.

If the dump program is low-speed with output directed to tape or high-speed, you will receive this message:

$\left\{\begin{array}{l}\text{HMD001A}\\\text{AMD001A}\end{array}\right\}$ TAPE=

This message allows you to accept the tape device specified in the macro instruction or specify a different tape device. SADMP will check the output volume to make sure it is non-labeled. If a label is present, SADMP will issue error message HMD002I and re-issue message HMD001A requesting that you identify the address of a tape device that has an unlabeled tape.

If the dump program is low-speed with output directed to a printer, you will receive this message:

$\left\{\begin{array}{l}\text{HMD001A}\\\text{AMD001A}\end{array}\right\}$ PTR=

This message allows you to accept the printer device specified in the macro instruction or specify a different printer.

Once SADMP has accepted an output device specification, it will issue message HMD011A. This message requests that you supply up to 100 characters to be used as a dump title. You should use this title to indicate why the dump is required.

When SADMP finishes dumping real storage, it issues this message:

$\left\{\begin{array}{l}\text{HMD005I}\\\text{AMD005I}\end{array}\right\}$ REAL DUMP DONE

If the dump program is high-speed, you will then receive this message:

$\left\{\begin{array}{l}\text{HMD012D}\\\text{AMD012D}\end{array}\right\}$ ENTER Y OR N FOR PAGEDUMP=

This message allows you to specify whether or not you want to dump the page data set. If you reply Y (that is, yes), SADMP issues message HMD021A to request the address of the page device. When the page dump is completed, SADMP issues this message:

$\left\{\begin{array}{l}\text{HMD023I}\\\text{AMD023I}\end{array}\right\}$ PAGE DUMP COMPLETE FOR DEVICE xxx

SADMP then issues this message:

$\left\{\begin{array}{l}\text{HMD024D}\\\text{AMD024D}\end{array}\right\}$ ENTER Y OR N - PAGE DUMP CONTINUE=

to allow you to go on to the next page data set or terminate. If you reply Y, SADMP re-issues message HMD021A to obtain the new page data set address, followed by messages HMD023I and HMD024D each time a page dump is completed. If you reply N, SADMP terminates.

Here is a sample exchange between SADMP and an operator during execution of a high-speed SADMP program.

```
HMD001A TAPE=282
HMD011A TITLE=tuesdaydump - to trace cascading error in job F3153647
HMD005I REAL DUMP DONE
HMD012D ENTER Y OR N FOR PAGEDUMP=y
HMD021A PAGE DEVICE ADDRESS=132
HMD023I PAGE DUMP COMPLETE FOR DEVICE 132
HMD024D ENTER Y OR N - PAGE DUMP CONTINUE=y
HMD021A PAGE DEVICE ADDR=191
HMD023I PAGEDUMP COMPLETE FOR DEVICE 191
HMD024D ENTER Y OR N - PAGE DUMP CONTINUE=n
```

In this example, the underlined characters represent the operator's replies.

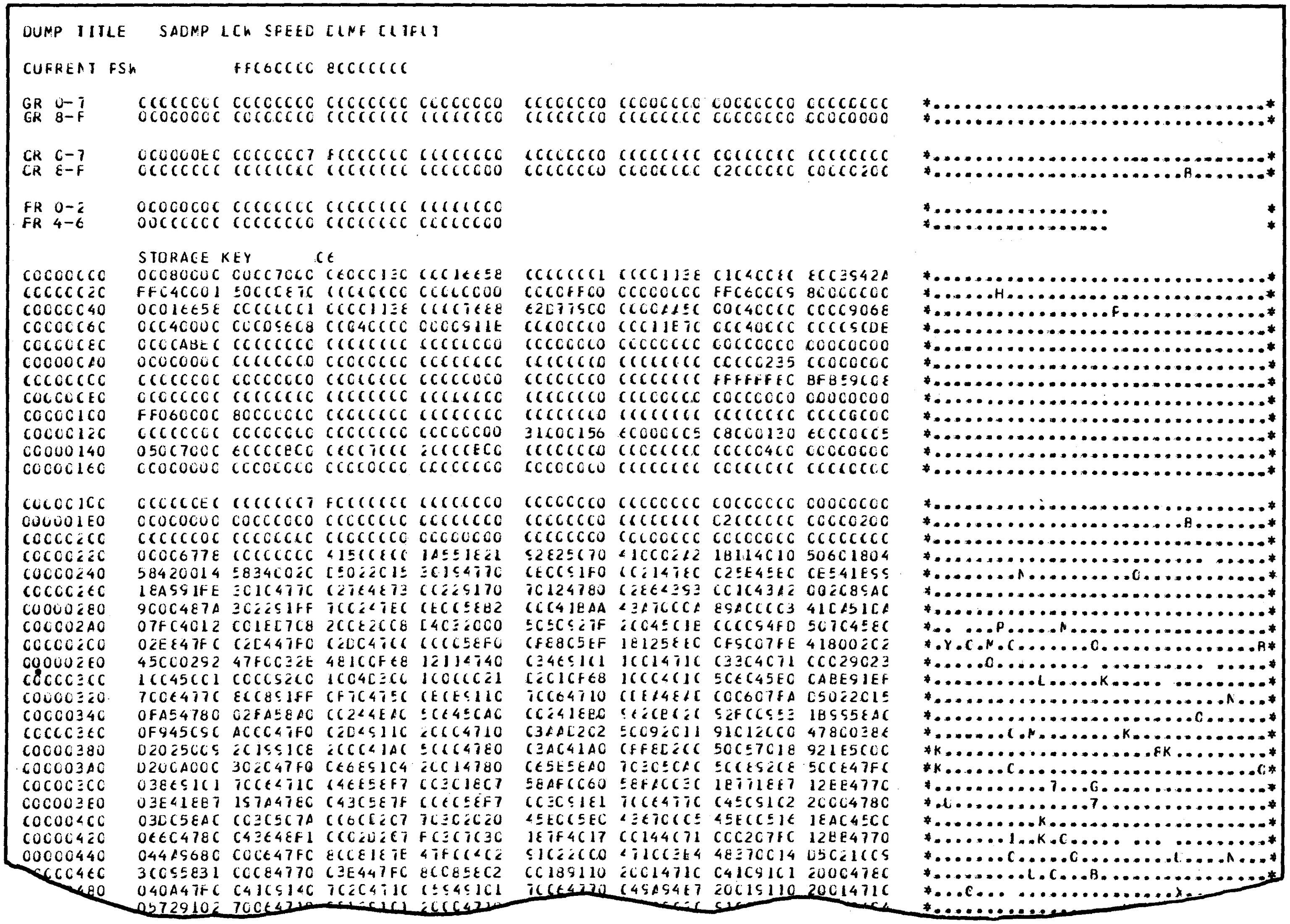
SADMP also uses wait state codes to communicate with the operator. These are described in Appendix B: Console Reference Summary for SADMP.

SADMP Output

The format of SADMP output depends on the version of the stand-alone program that generated it.

Low-Speed Output

Low-speed SADMP output, if directed to a printer, can be used immediately as a diagnostic aid. Figure SADMP-6 shows an example of SADMP low-speed output directed to a printer. For a full description of the fields, refer to the publication OS/VS1 Debugqinq Guide, GC26-5903 or OS/VS2 Debugging Guide, GC28-6203.



```
DUMP TITLE   SADMP LCW SPEED DUMP OUTPUT

CURRENT PSW          FFC6CCCC 8CCCCCCC

GR 0-7       CCCCCCCC CCCCCCCC CCCCCCCC CCCCCCC0  CCCCCCC0 CCC0CCCC C0CCCCC0 CCCCCCCC   *................................*
GR 8-F       0C0C000C CCCCCCCC CCCCCCCC CCCCCCC0  CCCCCCC0 CCCCCCCC CCCCCCCC CC0C0000   *................................*

CR 0-7       0C0000EC CCCCCCC7 FCCCCCCC CCCCCCCC  CCCCCCC0 CCCCCCCC CCCCCCCC CCCCCCCC   *................................*
CR 8-F       0CCCCCCC CCCCCCCC CCCCCCCC CCCCCC00  CCCCCCC0 CCCCCCCC C2CCCCCC C0CCC20C   *.........................B......*

FR 0-2       0C0C0C0C CCCCCCCC CCCCCCCC CCCCCCC0                                        *................                *
FR 4-6       00CCCCCC CCCCCCC0 CCCCCCCC CCCCCCC0                                        *................                *

             STORAGE KEY       C6
C0C00CC0     0C080C0C C0CC70CC C60CC13C CCC1EE58  CCCCCCCL CCCC113E C1C4CCEC EC03942A   *................................*
CCCCCC2C     FFC4CC01 50CCCE7C CCCCCCCC CCCCC000  CCCCFFC0 CCCC0CCC FFC6CCC9 8C0CCC0C   *......H.........................*
C000CC40     0C01E65E CCCCCCC1 CCCC113E CCCC7EE8  62D779C0 CCC0A45C C0C40CCC CCCC9068   *...................F............*
CCC0CC6C     0CC4C00C C0C09E08 CC04CCCC 0CC0911E  CCC0CCC0 CCC11E7C CCC40CCC CCCC9CDE   *................................*
CCCC0CEC     0CCCABEC CCCCCCCC CCCCCCCC CCCCCCC0  CCCCCCC0 CCCCCCCC C0CCCCCC C00C0C00   *................................*
C0000CA0     0C0C000C CCCCCCC0 CCC0CCCC CCCCCCCC  CCCCCCC0 CCCCCCCC CCCCC235 CC0C0C0C   *................................*
CCCC0CCC     CCCCCC0C CCCCCCC0 CCC0CCCC CCCCC0C0  CCCCCCC0 CCCCCCCC FFFFFFEC BFB59C0E   *................................*
C0CC0CEC     0CCCCCCC CCCCCCCC CCCCCCCC CCCCCCCC  CCCCCCC0 CCCCCCCC C0CCCCC0 00000C00   *................................*
C0C0C1C0     FF060C0C 80CCCCCC CCCCCCCC CCCCCCCC  CCCCCCC0 CCCCCCCC CCCCCCCC CCCCGC0C   *................................*
C000C12C     CCCCCCCC CCCCCCCC CCCCCCCC CCCCCC00  31C0C156 6C000CC5 C8CC0130 6CCCCCC5   *................................*
C0000140     050C700C 6CCCCBCC CECC7CCC 2CCCCECC  CCCCCCC0 CCCCCCCC CCCCC4CC CCCC0CCC   *................................*
C000C160     CC0C000C CCC0CCCC CCCC0CCC CCCCCCC0  CCC0CCC0 CCCCCCCC CCCCCCCC CCCCCCCC   *................................*

C0C0C1CC     CCCCCCEC CCCCCCC7 FCCCCCCC CCCCCCC0  CCCCCCC0 CCCCCCCC C0CCCCCC C00C0C0C   *................................*
000001E0     0C0C000C C0CCCCC0 CCCCCCCC CCCCCCC0  CCCCCCC0 CCCCCCCC C2CCCCCC CCCC020C   *.........................B......*
C0CCC2CC     CCCCCC0C CCCCCCCC CCCCCCC0 CCCC0C00  CCCCCCC0 CCC0CCCC CCCCCCCC CCCCCCCC   *................................*
C0C0022C     0C0C677E CCCCCCCC 415CCECC 1A551E21  92E25C70 41CC02A2 18114C10 506C1804   *.......................... .....*
C0000240     58420014 5834C02C D5022C15 3C194770  CECC91F0 CC2147EC C25E45EC CE541E99   *.........N...........O..........*
CCCCC26C     18A591FE 3C1C477C C27E4E73 CC229170  7C124780 C2E64393 CC1C43A2 002C89AC   *................................*
C0000280     9C0C487A 3C2291FF 7CC247EC CECC5EB2  CCC41BAA 43A7CCCA 89ACCCC3 41CA51CA   *................................*
C00002A0     07FC4012 C01ED7C8 2CCE2CC8 D4C32000  5C5C927F 2C045C1E CCCC94FD 507C458C   *.. ...P.....N...................*
C0CC02C0     02EE47FC C2D447F0 C2DC47CC CCCC58F0  CFE8C5EF 18125EEC CF9C07FE 418002C2   *.Y.C.M.C........O..............B*
000002E0     45C00292 47FCC32E 481CCF68 12114740  C34691C1 1CC1471C C33C4C71 CCC29023   *.....O......... .......... .....*
C0CCC3CC     1CC45CC1 CCCC92CC 1C04D3CC 1C0CCC21  C2C1CF68 1CCC4C1C 5C6C45E0 CABE91EF   *...........L.....K..... ........*
C0000320     7C06477C ECC891FF CF7C475C CECE911C  7CC64710 CCEA4EAC C0C607FA D5022C15   *............................N...*
C0000340     0FA54780 02FA58A0 CC244EAC 5C645CAC  CC241EB0 9E2CBC2C 92FCC953 1B955EAC   *.........................O......*
CCCCC36C     0F945C9C ACCC47F0 C2D4911C 2CCC4710  C3AAD2C2 5C092C11 91C12CC0 47800386   *.......C.M.........K............*
C0000380     D2025CC9 2C1991C8 2CCC41AC 5CCC4780  C3AC41A0 CFF8D2CC 50C57C18 921E5C0C   *K.......................BK .....*
C00003A0     D20CA00C 302C47F0 C66E91C4 2CC14780  C65E5EA0 7C3C5CAC 5CCE92CE 5CCE47FC   *K......C.......................C*
C0C0C3CC     038E91C1 7CCE471C C46E5EF7 CC3C18C7  58AFCC60 58FACC3C 1E771EE7 12EE477C   *............7...G...............*
C00003E0     03E418B7 197A478C C43C5E7F CC6C5EF7  CC3C91E1 7CCE477C C45C91C2 2C004780   *.U..............7...............*
C0C004C0     03DC58AC CC3C5C7A CC6CE2C7 7C3C2C20  45ECC5EC 43E70CC5 45ECC516 1EAC45CC   *...........K....................*
C0C00420     0E6C478C C43E48F1 CCC2D2E7 FC3C7C3C  1E7F4C17 CC144C71 CCC207FC 12BE4770   *........1..K.C..... ... ........*
00000440     044A968C C0CE47FC 8CCE1E7E 47FCC4C2  91C22CC0 471CC3E4 48370C14 D5C21CC9   *........C.....C.........C....N..*
 CCC04EC     3C095831 CCC84770 C3E447FC 8CC85EC2  CC189110 2CC1471C C41C91C1 2000478C   *..........L.C...B...............*
     480     040A47FC C41C914C 7C2C471C C59491C1  7CC64770 C49A94E7 20C19110 2001471C   *...C... ................X..
             05729102 700647                                                            *..............
```

Figure SADMP-6. Sample Low-Speed Dump

Low-speed SADMP output directed to tape can be printed using either the IEBPTPCH utility or PRDMP. Figures SADMP-7 and SADMP-8 show how to use IEBPTPCH and PRDMP, respectively, to print low-speed SADMP output. Note: You can also use the IEBGENER utility program to print low-speed SADMP output. For information about the IEBGENER program, refer to the publication OS/VS Utilities, GC35-0005.

```
//PRINTLO     JOB       MSGLEVEL=(1,1)
//            EXEC      PGM=IEBPTPCH
//SYSPRINT    DD        SYSOUT=A
//SYSUT1      DD        UNIT=2400,VOL=SER=DUMPTP,LABEL=(,NL),DISP=OLD,
//       DCB=(BLKSIZE=121,RECFM=F)
//SYSUT2      DD        SYSOUT=A
//SYSIN       DD        *
         PRINT     PREFORM=A
/*
```

Figure SADMP-7. Sample JCL Used to Invoke IEBPTPCH to Print Low-Speed SADMP Output.

```
//PTLODUMP    JOB       MSGLEVEL=(1,1)
//            EXEC      PROC=PRDMP
//DMP.SYSIN   DD        *
    PRINT          STORAGE
    END
/*
```

Figure SADMP-8. Sample JCL Used to Invoke PRDMP to Print Low-Speed SADMP Output

High-Speed Output

High-speed SADMP output must be printed using PRDMP. For full information, refer to chapter 5 in this publication.

SADMP Examples

The following examples show how to code the HMDSADMP or AMDSADMP macro instruction to create various kinds of stand-alone dump programs. In all the examples, the name of the macro instruction is represented by xMDSADMP. For VS1, replace this symbol with HMDSADMP; for VS2, replace it with AMDSADMP.

Example 1: Accepting All Defaults

In this example, the HMDSADMP or AMDSADMP macro instruction is used with no parameters to generate a high-speed, direct access resident dump program.

```
DUMP1    xMDSADMP
```

This is equivalent to coding the following parameters:

```
         TYPE=HI
         IPL=D3330
         VOLSER=SADUMP
         ULABEL=NOPURGE
         CONSOLE=(01F,3215)
         SYSUT=SYSDA
         OUTPUT=T282
```

Example 2: Generating a High-Speed, Tape Resident Dump Program

In this example, the IPL= parameter is coded to specify that the residence volume be a tape, and the VOLSER= parameter is coded to identify that tape. All other parameters are allowed to default.

```
         xMDSADMP  IPL=T2400-2,VOLSER=SATAPE
```

The implied defaults are:

```
         TYPE=HI
         CONSOLE=(01F,3215)
         SYSUT=SYSDA
         OUTPUT=T282
```

Example 3: Generating a Low-Speed Dump with Defaults

In this example. only the TYPE= parameter is coded to specify a low-speed dump. All other parameters are allowed to default.

```
         xMDSADMP  TYPE=LO
```

The implied defaults are:

```
         IPL=D3330
         OUTPUT=P00E
         VOLSER=SADUMP
         CONSOLE=(01F,3215)
         ULABEL=NOPURGE
         SYSUT=SYSDA
```

Example 4: Generating a Low-Speed Dump Program with Output Directed to Tape

In this example, only the TYPE= and OUTPUT= parameters are coded. All other parameters are allowed to default.

```
DUMP2     xMDSADMP  TYPE=LO,OUTPUT=T282
```

The implied defaults are:

```
          IPL=D3330
          VOLSER=SADUMP
          CONSOLE=(01F,3215)
          ULABEL=NOPURGE
          SYSUT=SYSDA
```

Chapter 8: SPZAP

Verifies and/or replaces data in a load module.

Contents

Figures

SPZAP

Introduction

SPZAP is a service aid program that operates as a problem program. It is designed to enable authorized personnel to:

- Inspect and modify instructions and data in any load module that is a member of a partitioned data set.
- Inspect and modify data in a specific record in a direct access data set.
- Dump an entire data set, a specific member of a partitioned data set, or any portion of a data set residing on a direct access device.
- Update the System Status Index (SSI) in the directory entry for any load module.

Capabilities of SPZAP

The functions of SPZAP provide many capabilities. Three of these are suggested below.

- By using the inspect and modify functions of SPZAP, programming errors that require only the replacement of instructions in a load module can be fixed without recompiling the program.
- The modify function of SPZAP can be used to set traps in a program by inserting invalid instructions. The invalid instructions will force abnormal termination; the dump of storage provided as a result of the abnormal termination is a valuable diagnostic tool, since it shows the contents of storage at a predictable point during execution.
- Since SPZAP can replace data directly on a direct access device, it could be used to reconstruct VTOCs or data records that may have been destroyed as the result of an I/O error or a programming error.

Monitoring the Use of SPZAP

Because SPZAP provides the ability to modify data on a direct access storage device, misuse of this program could result in serious damage to both user and system load modules or data sets. To protect against the occurrence of such damage by SPZAP, two means of controlling its use are suggested below:

- One means of exercising control is the System Management Facility (SMF), which provides a system interface with user exit routines for the purpose of monitoring the job stream. This facility, when incorporated into the system, affords an internal means of checking to see whether a particular user is authorized to execute the program specified on the EXEC job control language statement. (For further information on the SMF facility, refer to the publication OS/VS System Management Facilities (SMF), GC35-0004.)

- A second means of protecting against unauthorized use of SPZAP is to store SPZAP in a "password protected" private library. If SPZAP is located in such a library, any person trying to execute this program would be required to include in his JCL statements a JOBLIB DD statement defining the library, and at initiation time he would be required to give the password associated with the library. Only personnel knowing the password would then be able to execute SPZAP. Note, however, that if SPZAP resides in a private library, the authorized program facility (APF) will prevent it from updating a VTOC. Password protected libraries are discussed in the publication OS/VS Data Management for System Programmers, GC28-0631.

Data Modification and Inspection

SPZAP can be used to inspect and modify data in either a specific record of a direct access data set or a load module that is part of a partitioned data set.

The modification function is controlled by the REP control statement. The REP control statement allows you to replace instructions or data at a specific location in a load module or physical record.

The inspection function is controlled by the VERIFY statement. This function allows you to check the contents of a specific location in a load module or physical record prior to replacing it. If the contents at the specified location do not agree with the contents as specified in the VERIFY statement, subsequent REP operations will not be performed.

To avoid possible errors in replacing data, you should always precede any REP operation with a VERIFY operation.

Inspecting and Modifying a Load Module

To inspect or modify data in a load module, you must use a NAME control statement to supply SPZAP with the member name of the load module. The load module must be a member of the partitioned data set identified by the SYSLIB DD statement included in the execution JCL.

If the load module being inspected or modified contains more than one control section (CSECT), you must also supply SPZAP with the name of the CSECT that is to be inspected or modified. If no CSECT name is given in the NAME statement, SPZAP will assume that the control section to be processed is the first one encountered in searching the load module.

SPZAP will place descriptive maintenance data in the SPZAP CSECT Identification Record (IDR) of the load module whenever a REP operation associated with a NAME statement is performed on a control section contained in that module. This function will be performed automatically after all REP statements associated with the NAME statement have been processed; any optional user data that has to be placed in the IDR will come from the IDRDATA statement (See "SPZAP Control Statements" for an explanation of the IDRDATA statement).

Accessing a Load Module

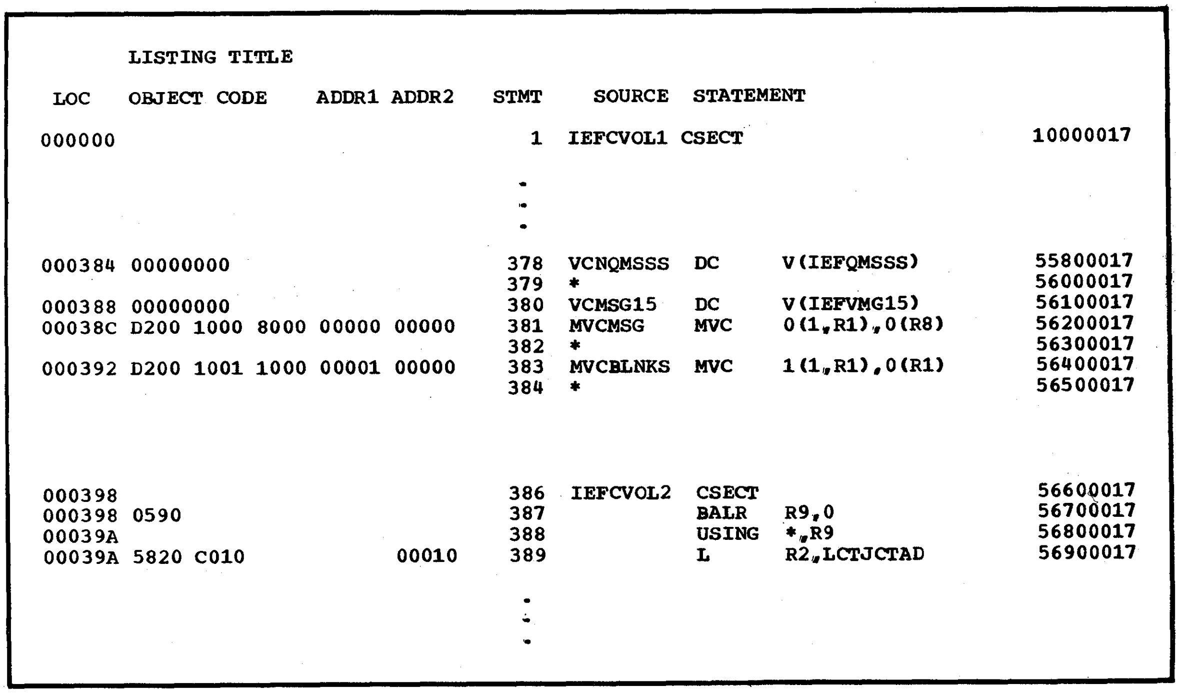
Once the CSECT has been found, SPZAP must locate the data that is to be verified and replaced. This is accomplished through the use of offset parameters in the VERIFY and REP statements. These parameters are specified in hexadecimal notation, and define the displacement of the data relative to the beginning of the CSECT. For example, if a hexadecimal offset of X'40' is specified in a VERIFY statement, SPZAP will find the location that is 64 bytes beyond the beginning of the CSECT identified by the NAME statement, and begin verifying the data from that point.

Normally, the assembly listing address associated with the instruction to be inspected or modified can be used as the offset value in the VERIFY or REP statement. However, if a CSECT has been assembled with other CSECTs so that its origin is not at assembly location zero, then the locations in the assembly listing do not reflect the correct displacements of data in the CSECT. The proper displacements must be computed by subtracting the assembly listing address delimiting the start of the CSECT from the assembly listing address of the data to be referenced.

To eliminate the need for such calculations and allow you to use the assembly listing locations, SPZAP provides a means of adjusting the offset values on VERIFY and REP statements. This is achieved through the use of the BASE control statement. This statement should be included in the input to SPZAP immediately following the NAME statement that identifies the CSECT. The parameter in the BASE statement must be the assembly listing address (in hexadecimal) at which the CSECT beings. SPZAP will then subtract this value from the offset specified on any VERIFY or REP statement that follows the BASE statement, and use the difference as the displacement of the data.

For a complete description of the control statements mentioned in this discussion, see the section "SPZAP Control Statements" in this chapter.

Figure SPZAP-1 is a sample assembly listing showing more than one control section. To refer to the second CSECT (IEFCVOL2), you could include in the input to SPZAP a BASE statement with a location of 0398. Then, to refer to the subsequent LOAD instruction (L R2,LCTJCTAD), you could use an offset of 039A in the VERIFY or REP statements that follow in the SPZAP input stream.



```
        LISTING TITLE

  LOC   OBJECT CODE    ADDR1 ADDR2   STMT    SOURCE  STATEMENT

 000000                                 1  IEFCVOL1 CSECT                            10000017

                                        .
                                        .
                                        .

 000384 00000000                      378  VCNQMSSS  DC       V(IEFQMSSS)            55800017
                                      379  *                                         56000017
 000388 00000000                      380  VCMSG15   DC       V(IEFVMG15)            56100017
 00038C D200 1000 8000 00000 00000    381  MVCMSG    MVC      0(1,R1),0(R8)          56200017
                                      382  *                                         56300017
 000392 D200 1001 1000 00001 00000    383  MVCBLNKS  MVC      1(1,R1),0(R1)          56400017
                                      384  *                                         56500017



 000398                               386  IEFCVOL2  CSECT                           56600017
 000398 0590                          387            BALR     R9,0                   56700017
 00039A                               388            USING    *,R9                   56800017
 00039A 5820 C010           00010     389            L        R2,LCTJCTAD            56900017

                                        .
                                        .
                                        .
```

Figure SPZAP 1. Sample Assembly Listing Showing Multiple Control Sections.

Inspecting and Modifying a Data Record

To inspect or modify a specific data record, you must use a CCHRR control statement to specify its direct access address. This CCHHR address must be within the limits of the direct access data set defined in the SYSLIB DD control statement.

If you request a REP operation for a record identified by a CCHHR control statement, SPZAP will issue message HMA112I or AMA112I to provide a record of your request.

Accessing a Data Record

When you use the CCHRR control statement, SPZAP is able to read directly the physical record you want to inspect or modify. The offset parameters specified in subsequent VERIFY and REP statements are then used to locate the data that is to be verified or replaced within the record. These hexadecimal offsets must define the displacement of data relative to the beginning of the record and include the length of any key field.

SPZAP

Dumping Data

SPZAP's dumping options provide a visual picture of the load module or data record that has been changed, thus allowing you to double check the modifications you have made.

The DUMP and ABSDUMP statements are the control statements used to specify the dumping options. The operation code in the DUMP and ABSDUMP statements indicates the kind of dump you want, a formatted hexadecimal dump or a translated dump; the parameters identify the portion of the data to be dumped. (Use of the DUMP and ABSDUMP statements is discussed in detail under the topic "SPZAP Control Statements.")

Updating System Status Information

The system status index (SSI) is a 4-byte field created by the linkage editor in the directory entry of a load module. It is useful for keeping track of any modifications that are performed on a load module. SPZAP updates the system status index automatically whenever it replaces data in the associated module.

SPZAP also supplies the SETSSI control statement, which you can use to overlay the existing data in the SSI with your own data. For a complete description of the SETSSI control statement, see the section "SPZAP Control Statements" in this chapter.

Not all load modules have system status present, the SSI System Status Index is located in the last four bytes of the user data field in the directory entry for a load module. Figure SPZAP-2 shows the position of the SSI in load module directory entries.

Member Name	TTR	C	User Data Field	SSI
1 8	9 11	12	13 to 70 maximum	variable

Figure SPZAP-2. SSI Bytes in a Load Module Directory Entry

Figure SPZAP-3 shows the composition of the System Status Index field and the flag bits used to indicate the types of changes made to the corresponding load module program. The first byte of SSI information contains the member's change level. When a load module is initially released by IBM, its change level is set at one. Thereafter, the change level is incremented by one for each release that includes a new version of that program. If you make a change to the SSI for any of the IBM-released programs, take care not to destroy this maintenance level indicator unless you purposely mean to do so. To keep the change level byte at its original value, find out what information is contained in the SSI before using the SETSSI function.

Note: Use the LISTLOAD control statement of the LIST service aid to find out what information the SSI contains.

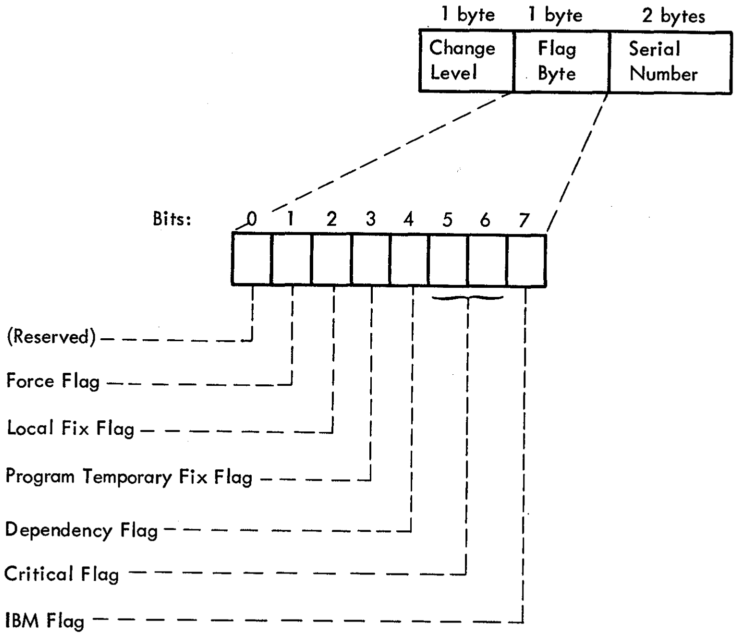


Figure SPZAP-3. Flag Bits in the System Status Index Field

The second byte of the SSI is termed the flag byte. Bits within the flag byte contain information reflecting the member's maintenance status. You need only be concerned with two of the eight bits when you are using SPZAP:

- The local fix flag contained in bit 2 is used to indicate that the user has modified a particular member. (It is not used to reflect modifications made by IBM-supplied PTFs.) SPZAP sets this local fix flag bit to one after successfully modifying to a load module.

- The program temporary fix flag in bit 3 is set to one when an IBM-authorized program temporary fix (PTF) is applied to a system library to correct an error in an IBM module.

All other bits in the flag byte should be retained in the SSI as they appeared before the SETSSI operation was enacted, so as not to interfere with the normal system maintenance procedures.

The third and fourth bytes of the system status index are used to store a serial number that identifies the first digit and the last three digits of a PTF number. SPZAP will not change these bytes unless you request a change by using the SETSSI control statement.

Operational Considerations

Consider the following points when you run SPZAP:

- SPZAP utilizes system OPEN, and therefore cannot modify "read-only" or inspect "write-only" password protected data sets unless the correct password is provided at OPEN.

- Unexpired data sets such as system libraries cannot be modified unless the operator replies r xx,'U' to the expiration message that occurs during OPEN.

- If SPZAP is used to modify an operating system module that is made resident in virtual storage only at IPL time, an additional IPL is required to invoke the new version of the altered module. (Note that for VS2 this includes all modules in SYS1.LPALIB.)

- The SYSLIB DD statement cannot define a concatenated data set.

- SPZAP supports only the following direct access devices: 2314, 2319, 2305, and 3330. One of these devices must be specified in the unit parameter of the SYSLIB DD statement.

- SPZAP is a non-reusable module.

- When modifying a system data set, such as SYS1.LINKLIB, DISP=OLD should be specified on the SYSLIB DD statement.

SPZA

JCL Statements

SPZAP can be executed using the following job control statements. The minimum partition or region for execution is 19K plus the larger of 3K or the blocksize in bytes for the data set specified on the SYSLIB DD statement.

JOB Statement

marks the beginning of the job.

EXEC Statement

invokes the program HMASPZAP (in VS1) or AMPSPZAP (in VS2).

SYSPRINT DD Statement

defines a sequential output message data set, that can be written on a system printer, a magnetic tape volume, or a direct access volume. This statement is required for each execution of SPZAP.

SYSLIB DD Statement (required for each execution)

defines the direct access data set that will be accessed by SPZAP when performing the operations specified on the control statements. The DSNAME parameter and DISP=OLD or DISP=SHR must always be defined. The VOLUME and UNIT parameters are necessary only of the data set is not cataloged. When this data set is the VTOC, DSNAME=FORMAT4.DSCB must be specified. This statement cannot define a concatenated data set.

SYSABEND DD Statement (optional)

defines a sequential output data set to be used in case SPZAP terminates abnormally. This data set can be written to a printer, a magnetic tape volume, or a direct access volume.

SYSIN DD Statement

Defines the input stream data set that contains SPZAP control statements.

SPZAP Control Statements

The SPZAP control statements (entered either through the user's input stream or through the system console) define the processing functions to be performed during a particular execution of SPZAP.

SPZAP control statements must be coded according to the following rules:

- SPZAP control statements may begin in any column, but the operation name must precede the parameters.

- There must be at least one blank between the specified operation name and the first parameter.

- All parameters must also be separated by at least one blank space.

- Data field parameters may be formatted with commas for easier visual check, but embedded blanks within data fields are not permitted.

- Data and offset parameter values must be specified as a multiple of two hexadecimal digits.

- The size of an SPZAP control statement is 80 bytes.

- Following the last required parameter and its blank delimiter, the rest of the control statement space can be used for comments. Exceptions to this are the NAME and DUMP control statements. If the CSECT parameter is omitted from either of these statements, the space following the load module parameter should not be used for comments.

- A record beginning with an asterisk and a blank is considered to be a comment statement.

The control statements are the following:

NAME member [csect]

> used to identify a CSECT in a load module that is to be the object of subsequent VERIFY, REP, or SETSSI operations. The parameters are:
>
> member
>
> > the member name of the load module that contains the control section in which the data to be inspected and/or modified is resident. The load module must be a member of the partitioned data set defined by the SYSLIB DD statement.
>
> csect
>
> > the name of the particular control section that contains the data to be verified or replaced. When this parameter is omitted, it is assumed that the first CSECT contained in the load module is the one to be referenced. If there is only one CSECT in the load module, this parameter is not necessary.
> >
> > Note: More than one NAME statement can be defined in the input to SPZAP. However, the VERIFY, REP and SETSSI statements associated with each NAME statement must immediately follow the NAME statement to which they apply.

CCHHR record address

used to identify a physical record on a direct access device that is to be modified or verified. The record must be in the data set defined by the SYSLIB DD statement. Any immediately following REP or VERIFY statements will reference the data in the specified record. The parameter is:

record address

the actual direct access device address of the record containing the data to be replaced or verified. It must be specified as a 10-digit hexadecimal number in the form ccccchhhhrr, where cccc is the cylinder, hhhh is the track, and rr is the record number. For example, 0001000A01 addresses record 1 of cylinder 1, track 10.

A zero record number is invalid and will default to 1.

Note: More than one CCHHR statement can be defined in the input to SPZAP. However, the VERIFY, REP and SETSSI statements associated with each CCHHR statement must immediately follow the specific CCHHR statement to which they apply.

$\left\{\begin{matrix}\text{VERIFY}\\ \text{VER}\end{matrix}\right\}$ offset expected content

causes the contents at a specified location within a control section or physical record to be compared with the data the user supplies in the statement. If the two fields being compared are not in agreement, that is, if the VERIFY operation is rejected, no succeeding REP or SETSSI operations will be performed until the next NAME or CCHHR control statement is encountered. SPZAP provides a formatted dump of each CSECT or record for which a VERIFY operation failed.

offset

the hexadecimal displacement of the data to be inspected in a CSECT or record. This displacement does not have to be aligned on a fullword boundary, but it must be specified as a multiple of two hexadecimal digits (0D, 021C, 014682, etc.). If this offset value is outside the limits of the CSECT or data record defined by the preceding NAME or CCHHR statement, the VERIFY statement will be rejected. When inspecting a record with a key, the length of the key should be considered in the calculation of the displacement; that is, offset zero is the first byte of the key.

expected content

defines the bytes of data that are expected at the specified location. As with the offset parameter, the number of bytes of data defined must be specified as a multiple of two hexadecimal digits. If desired, the data within the parameters may be separated by commas (never blanks), but again, the number of digits between commas must also be a multiple of two. For example, the data may look like this:

5840C032 (without commas),

or like this:

5840,C032 (with commas)

If all the data will not fit into one VERIFY statement (80-byte logical record), then another VERIFY statement must be defined.

REP offset data

used to modify data at a specified location in a CSECT or physical record that has been previously defined by a NAME or CCHHR statement. The data specified on the REP statement will replace the data at the record or CSECT location stipulated in the offset parameter field. (Note that you should always use the VERIFY function to make sure you know what you are going to change with the REP function.) Message HMA122I or AMA122I will be issued to record the contents of the specified location as they were before the change was made.

offset

is the hexadecimal displacement of the data to be replaced in a CSECT or data record. This displacement need not address a fullword boundary, but it must be specified as a multiple of two hexadecimal digits (0D, 02C8, 001C52). If this offset value is outside the limits of the data record (physical block) or CSECT being modified, the replacement operation will not be performed. When replacing data in a record with a key, the length of the key should be considered in the calculation of the displacement; that is, offset zero is the first byte of the key.

data

defines the bytes of data that are to be inserted at the specified location. As with the offset parameter, the number of bytes of data defined must be specified as a multiple of two hexadecimal digits. If desired, the data within the parameter may be separated by commas (never blanks), but again, the number of digits between commas must also be a multiple of two. For example, a REP data parameter may look like this:

4160B820 (without commas)

or like this:

4160,B820 (with commas).

If all the data to be modified will not fit into one REP statement (80- byte logical record), then another REP statement must be defined.

Remember that SPZAP automatically updates the system status index (SSI) when it successfully modifies the associated load module. For a more complete explanation of the value of the SSI to the maintenance of a load module, refer to "Updating System Status Information" in this chapter.

Two programming notes that are pertinent to this discussion of the REP statement are listed below:

- If multiple VERIFY and REP operations are to be performed on a CSECT, then all the VERIFY statements should precede all the REP statements. This procedure will ensure that all the REP operations are ignored if a VERIFY reject occurs.

- When a record in the VTOC (that is, a DSCB) is accessed for modification, message HMA117D or AMA117D is written to the console. No message is issued, however, when an ABSDUMPT operation is performed on the VTOC.

IDRDATA xxxxxxxx

causes SPZAP to place up to eight bytes of user data into the SPZAP CSECT Identification Record of the load module; this is only done if a REP operation associated with a NAME statement is performed and the load module has been processed by the Linkage Editor to include CSECT Identification Records. The parameter is:

xxxxxxxx

is the eight (or less) bytes of user data (with no embedded blanks) that is to be placed in user data field of the SPZAP IDR of the load module. If more than eight characters are in the parameter field only the first eight characters will be used.

The IDRDATA statement is valid only when used in conjunction with the NAME statement. It must follow its associated NAME statement and precede any DUMP or ABSDUMP statement. IDRDATA statements associated with CCHHR statements will be ignored.

SETSSI xxyynnnn

places user-supplied system status information in the PDS (partitioned data set) directory entry for the library member specified in the preceding NAME statement. The SSI, however, must have been created when the load module was link edited. The parameter is:

xxyynnnn

represents the 4 bytes of system status information the user wishes to place in the SSI field for this member. Each byte is supplied as two hexadecimal digits signifying the following:

xx - change level

yy - flag byte

nnnn - modification serial number

If an error has been detected in any previous VERIFY or REP operation, the SETSSI function will not be performed.

Note: Since all bits in the SSI entry are set (or reset) by the SETSSI statement, extreme care should be exercised in its use to avoid altering information vital to the depiction of the maintenance status of the program being changed. Message HMA122I or AMA122I is issued to record the SSI as it was before the SETSSI operation was performed. (See the discussion in this chapter entitled "Updating System Status Information.")

```
{DUMP  }  member  csect
{DUMPT }          ALL
```

used to dump a specific control section or all control sections in a load module. The format of the output of this dump is hexadecimal (see the discussion in this chapter entitled "SPZAP Output"). The DUMPT statement differs from the DUMP statement in that it also gives the user an EBCDIC and instruction mnemonic translation of the hexadecimal data. The parameters are:

member

the member name of the load module that contains the control section(s) to be dumped. (Note: This load module must be a member of a partitioned data set that is defined by the SYSLIB DD statement.)

csect

defines the name of the particular control section that is to be dumped. To dump all the CSECTs of a load module, code "ALL" instead of the CSECT name; if the CSECT parameter is omitted entirely, it is assumed that the user means to dump only the first control section contained in the load module.

```
{ABSDUMP }  {startaddr stopaddr}
{ABSDUMPT}  {membername        }
            {ALL               }
```

These statements are used to dump a group of data records, a member of a partitioned data set, or an entire data set, as defined in the SYSLIB DD statement. If the key associated with each record is to be formatted, DCB=(KEYLEN=nn), where "nn" is the length of the record key, must also be specified by the SYSLIB DD statement. Note that when dumping a VTOC, DCB=(KEYLEN=44) should be specified; when dumping a PDS directory, DCB=(KEYLEN=8) should be specified. ABSDUMP produces a hexadecimal printout only, while ABSDUMPT prints the hexadecimal data, the EBCDIC translation, and the mnemonic equivalent of the data (see "SPZAP Output"). The parameters are:

startaddr

is the absolute direct access device address of the first record to be dumped. This address must be specified in hexadecimal in the form cccchhhhrr (cylinder, track and record numbers).

stopaddr

is the absolute direct access device address of the last record to be dumped, and it must be in the same format as the start address.

Note: Both addresses must be specified when this method of dumping records is used, and both addresses must be within the limits of the data set defined by the SYSLIB DD statement. The record number specified in the start address must be a valid record number. The record number specified as the stop address need not be a valid record number, but if it is not, the dump will continue until the last record on the track specified in the stop address has been dumped.

membername

is the name of a member of a partitioned data set. The member can be a group of data records or a load module. In either case, the entire member is dumped when this parameter is specified.

ALL

specifies that the entire data set defined by the SYSLIB DD statement is to be dumped.

How much of the space allocated to the data set is dumped depends on how the data set is organized:

For sequential data set, SPZAP dumps until it reaches end of file.

For indexed sequential and direct access data sets, SPZAP dumps all extents.

For partitioned data sets, SPZAP dumps all extents, including all linkage editor control records, if any exist.

BASE xxxxxx

used by SPZAP to adjust offset values that are to be specified in any subsequent VERIFY and REP statements. This statement should be used when the offsets given in the VERIFY and REP statements for a CSECT are to be obtained from an assembly listing in which the starting address of the CSECT is not location zero.

For example, assume that CSECT ABC begins at assembly listing location X'000400', and that the data to be replaced in this CSECT is at location X'000408'. The actual displacement of the data in the CSECT is X'08'. However, an offset of X'0408' (obtained from the assembly listing location X'000408') can be specified in the REP statement if a BASE statement specifying X'000400' is included prior to the REP statement in the SPZAP input stream. When SPZAP processes the REP statement, the base value X'000400' will be subtracted from the offset X'0408' to determine the proper displacement of data within the CSECT. The parameter is:

xxxxxx

is a 6-character hexadecimal offset that is to be used as a base for subsequent VERIFY and REP operations. This value should reflect the starting assembly listing address of the CSECT being inspected or modified.

The BASE statement should be included in the SPZAP input stream immediately following the NAME statement that identifies the control section that is to be involved in the SPZAP operations. The specified base value remains in effect until all VERIFY, REP, and SETSSI operations for the CSECT have been processed.

CONSOLE

indicates that SPZAP control statements are to be entered through the system console.

When this statement is encountered in the input stream, the following message is written to the operator:

$\left\{\begin{array}{l}\text{HMA116A}\\\text{AMA116A}\end{array}\right\}$ ENTER HMASPZAP CONTROL STATEMENT OR END

The operator may then key in any valid SPZAP control statement conforming to the specifications described at the beginning of this control statement discussion. After each operator entry through the console is read, validated, and processed, the message is reissued, and additional input is accepted from the console until "END" is replied. SPZAP will then continue processing control statements from the input stream until an end-of-file condition is detected.

Note: The control statements can be entered through the console in either upper or lower case letters.

* (Comment)

can be used to annotate the SPZAP input stream and output listing. Any number of comment statements can be included in the input stream. When such a statement is encountered, SPZAP writes the entire statement to the data set specified for SYSPRINT.

The asterisk (*) can be specified in any position of the statement, but it must be followed by at least one blank space as a delimiter.

SPZAP Output

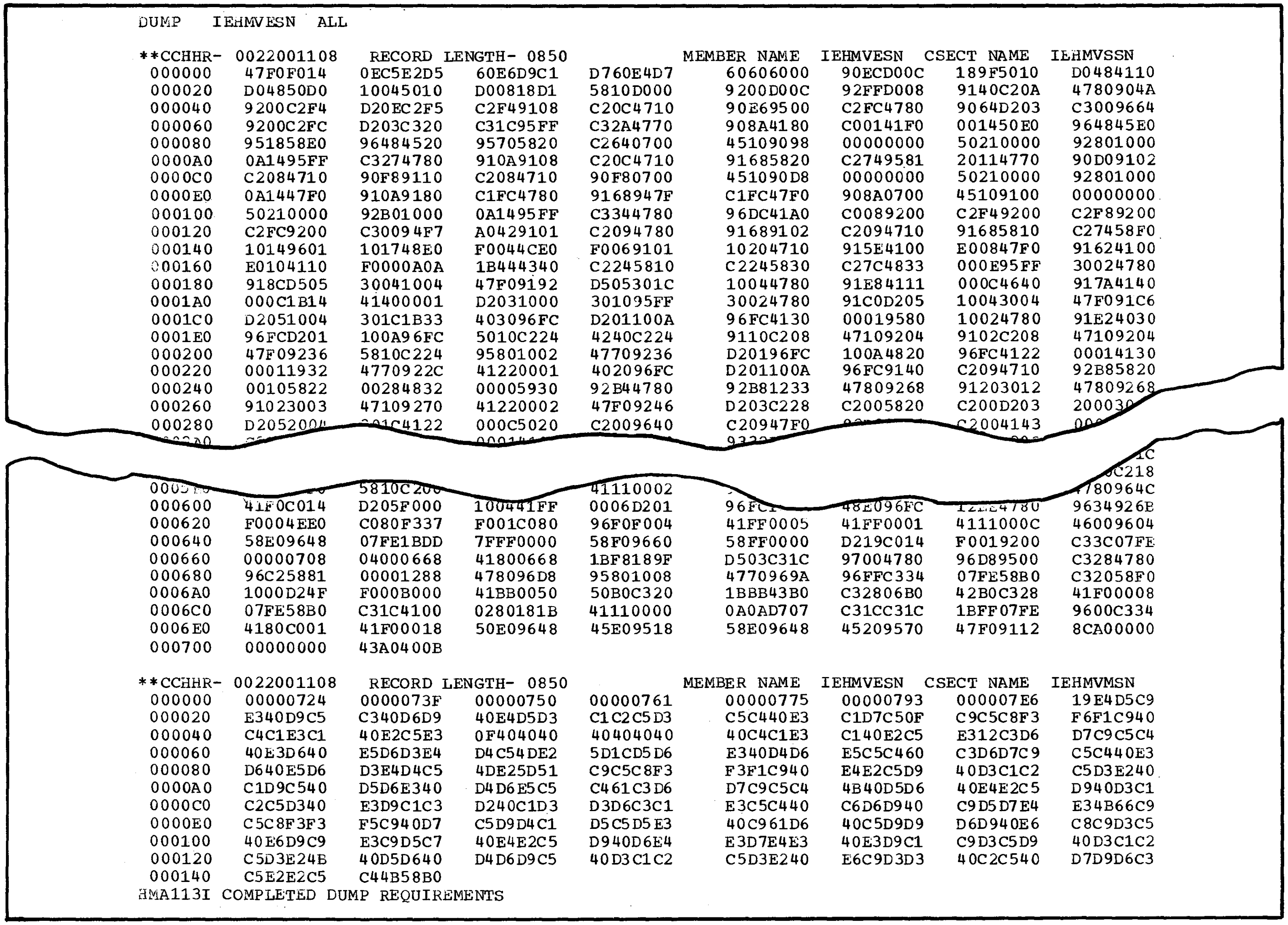
SPZAP provides two different dump formats for the purpose of checking the data that has been verified and/or replaced. These dumps (written to the SYSPRINT data set specified by the user) may be of the formatted hexadecimal type or the translated form. Both formats are discussed below in detail with examples showing how each type will look.

The Formatted Hexadecimal Dump

When DUMP or ABSDUMP is the control statement used, the resulting printout will be a hexadecimal representation of the requested data. Figure SPZAP-4 gives a sample of the formatted hexadecimal dump. A heading line is printed at the beginning of each block. This heading consists of the hexadecimal direct access address of the block, a two-byte record length field, and the names of the member and the control section that contain the data being printed (if the dump is for a specific CSECT or load module). Each printed line thereafter has a three-byte displacement address at the left, followed by eight groups of four data bytes each. The following message:

```
{HMA113I} COMPLETED DUMP REQUIREMENTS
{AMA113I}
```

is printed directly under the last line of the dump printout.

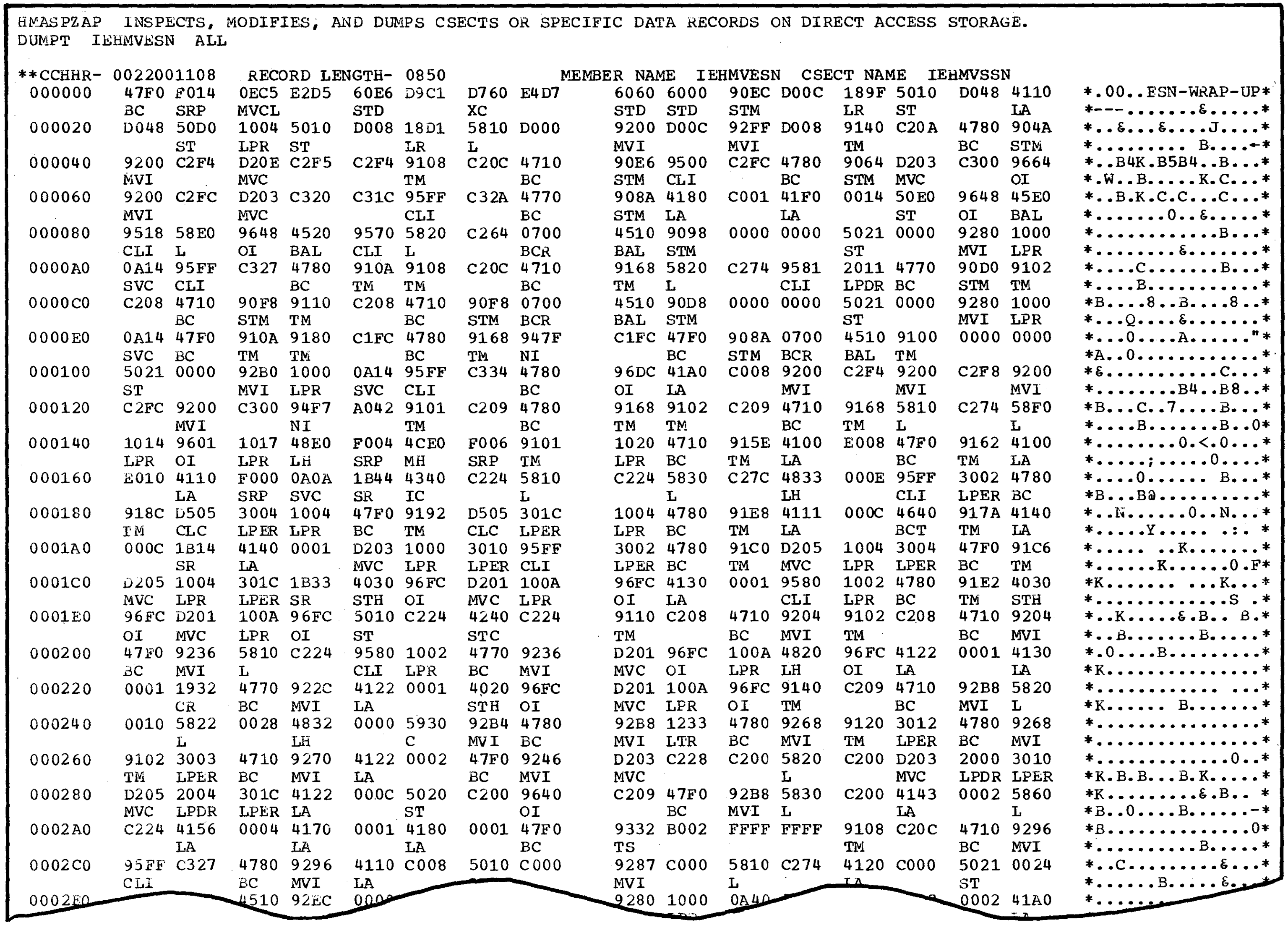


```
DUMP   IEHMVESN  ALL

**CCHHR-  0022001108    RECORD LENGTH- 0850              MEMBER NAME  IEHMVESN  CSECT NAME  IEHMVSSN
 000000    47F0F014    0EC5E2D5    60E6D9C1    D760E4D7     60606000    90ECD00C    189F5010    D0484110
 000020    D04850D0    10045010    D00818D1    5810D000     9200D00C    92FFD008    9140C20A    4780904A
 000040    9200C2F4    D20EC2F5    C2F49108    C20C4710     90E69500    C2FC4780    9064D203    C3009664
 000060    9200C2FC    D203C320    C31C95FF    C32A4770     908A4180    C00141F0    001450E0    964845E0
 000080    951858E0    96484520    95705820    C2640700     45109098    00000000    50210000    92801000
 0000A0    0A1495FF    C3274780    910A9108    C20C4710     91685820    C2749581    20114770    90D09102
 0000C0    C2084710    90F89110    C2084710    90F80700     451090D8    00000000    50210000    92801000
 0000E0    0A1447F0    910A9180    C1FC4780    9168947F     C1FC47F0    908A0700    45109100    00000000
 000100    50210000    92B01000    0A1495FF    C3344780     96DC41A0    C0089200    C2F49200    C2F89200
 000120    C2FC9200    C30094F7    A0429101    C2094780     91689102    C2094710    91685810    C27458F0
 000140    10149601    101748E0    F0044CE0    F0069101     10204710    915E4100    E00847F0    91624100
 000160    E0104110    F0000A0A    1B444340    C2245810     C2245830    C27C4833    000E95FF    30024780
 000180    918CD505    30041004    47F09192    D505301C     10044780    91E84111    000C4640    917A4140
 0001A0    000C1B14    41400001    D2031000    301095FF     30024780    91C0D205    10043004    47F091C6
 0001C0    D2051004    301C1B33    403096FC    D201100A     96FC4130    00019580    10024780    91E24030
 0001E0    96FCD201    100A96FC    5010C224    4240C224     9110C208    47109204    9102C208    47109204
 000200    47F09236    5810C224    95801002    47709236     D20196FC    100A4820    96FC4122    00014130
 000220    00011932    4770922C    41220001    402096FC     D201100A    96FC9140    C2094710    92B85820
 000240    00105822    00284832    00005930    92B44780     92B81233    47809268    91203012    47809268
 000260    91023003    47109270    41220002    47F09246     D203C228    C2005820    C200D203    20003
 000280    D205200      C4122    000C5020    C2009640     C20947F0                C2004143    000

                                                                                                 C
                                                                                               C218
 000         5810C20                41110002                                                 780964C
 000600    41F0C014    D205F000    100441FF    0006D201     96FC        48E096FC                9634926E
 000620    F0004EE0    C080F337    F001C080    96F0F004     41FF0005    41FF0001    4111000C    46009604
 000640    58E09648    07FE1BDD    7FFF0000    58F09660     58FF0000    D219C014    F0019200    C33C07FE
 000660    00000708    04000668    41800668    1BF8189F     D503C31C    97004780    96D89500    C3284780
 000680    96C25881    00001288    478096D8    95801008     4770969A    96FFC334    07FE58B0    C32058F0
 0006A0    1000D24F    F000B000    41BB0050    50B0C320     1BBB43B0    C32806B0    42B0C328    41F00008
 0006C0    07FE58B0    C31C4100    0280181B    41110000     0A0AD707    C31CC31C    1BFF07FE    9600C334
 0006E0    4180C001    41F00018    50E09648    45E09518     58E09648    45209570    47F09112    8CA00000
 000700    00000000    43A0400B

**CCHHR-  0022001108    RECORD LENGTH- 0850              MEMBER NAME  IEHMVESN  CSECT NAME  IEHMVMSN
 000000    00000724    0000073F    00000750    00000761     00000775    00000793    000007E6    19E4D5C9
 000020    E340D9C5    C340D6D9    40E4D5D3    C1C2C5D3     C5C440E3    C1D7C50F    C9C5C8F3    F6F1C940
 000040    C4C1E3C1    40E2C5E3    0F404040    40404040     40C4C1E3    C140E2C5    E312C3D6    D7C9C5C4
 000060    40E3D640    E5D6D3E4    D4C54DE2    5D1CD5D6     E340D4D6    E5C5C460    C3D6D7C9    C5C440E3
 000080    D640E5D6    D3E4D4C5    4DE25D51    C9C5C8F3     F3F1C940    E4E2C5D9    40D3C1C2    C5D3E240
 0000A0    C1D9C540    D5D6E340    D4D6E5C5    C461C3D6     D7C9C5C4    4B40D5D6    40E4E2C5    D940D3C1
 0000C0    C2C5D340    E3D9C1C3    D240C1D3    D3D6C3C1     E3C5C440    C6D6D940    C9D5D7E4    E34B66C9
 0000E0    C5C8F3F3    F5C940D7    C5D9D4C1    D5C5D5E3     40C961D6    40C5D9D9    D6D940E6    C8C9D3C5
 000100    40E6D9C9    E3C9D5C7    40E4E2C5    D940D6E4     E3D7E4E3    40E3D9C1    C9D3C5D9    40D3C1C2
 000120    C5D3E24B    40D5D640    D4D6D9C5    40D3C1C2     C5D3E240    E6C9D3D3    40C2C540    D7D9D6C3
 000140    C5E2E2C5    C44B58B0
HMA113I COMPLETED DUMP REQUIREMENTS
```

Figure SPZAP-4. Sample Formatted Hexadecimal Dump

The Translated Dump

The control statements DUMPT and ABSDUMPT also provide an operation code translation and an EBCDIC representation of the data contained in the dump. Figure SPZAP-5 shows the format of the translated dump. The first byte of each halfword of data is translated into its mnemonic operation code equivalent, provided such a translation is possible. If there is no equivalent mnemonic representational value to be given, the space is left blank. This translated line of codes and blanks is printed directly under the corresponding hexadecimal line. An EBCDIC representation of each byte of data is printed on two lines to the right of the corresponding line of text with periods (.) substituted for those bytes that do not translate to valid printable characters.



```
HMASPZAP  INSPECTS, MODIFIES, AND DUMPS CSECTS OR SPECIFIC DATA RECORDS ON DIRECT ACCESS STORAGE.
DUMPT  IEHMVESN  ALL

**CCHHR- 0022001108   RECORD LENGTH- 0850           MEMBER NAME  IEHMVESN  CSECT NAME  IEHMVSSN
 000000     47F0 F014  0EC5 E2D5  60E6 D9C1  D760 E4D7     6060 6000  90EC D00C  189F 5010  D048 4110   *.00..ESN-WRAP-UP*
            BC   SRP   MVCL       STD        XC            STD  STD   STM        LR   ST         LA     *---.......&.....*
 000020     D048 50D0  1004 5010  D008 18D1  5810 D000     9200 D00C  92FF D008  9140 C20A  4780 904A   *..&...&....J....*
                 ST    LPR  ST         LR    L             MVI        MVI        TM         BC   STM    *......... B....←*
 000040     9200 C2F4  D20E C2F5  C2F4 9108  C20C 4710     90E6 9500  C2FC 4780  9064 D203  C300 9664   *..B4K.B5B4..B...*
            MVI        MVC             TM         BC       STM  CLI        BC    STM  MVC        OI     *.W..B.....K.C...*
 000060     9200 C2FC  D203 C320  C31C 95FF  C32A 4770     908A 4180  C001 41F0  0014 50E0  9648 45E0   *..B.K.C.C...C...*
            MVI        MVC             CLI        BC       STM  LA         LA         ST    OI   BAL    *.......0..&.....*
 000080     9518 58E0  9648 4520  9570 5820  C264 0700     4510 9098  0000 0000  5021 0000  9280 1000   *............B...*
            CLI  L     OI   BAL   CLI  L          BCR      BAL  STM              ST         MVI  LPR    *.........&......*
 0000A0     0A14 95FF  C327 4780  910A 9108  C20C 4710     9168 5820  C274 9581  2011 4770  90D0 9102   *....C.......B...*
            SVC  CLI        BC    TM   TM         BC       TM   L          CLI   LPDR BC    STM  TM     *....B...........*
 0000C0     C208 4710  90F8 9110  C208 4710  90F8 0700     4510 90D8  0000 0000  5021 0000  9280 1000   *B....8..B....8..*
                 BC    STM  TM         BC    STM  BCR      BAL  STM              ST         MVI  LPR    *...Q....&.......*
 0000E0     0A14 47F0  910A 9180  C1FC 4780  9168 947F     C1FC 47F0  908A 0700  4510 9100  0000 0000   *...0....A......"*
            SVC  BC    TM   TM         BC    TM   NI            BC    STM  BCR   BAL  TM                *A..0............*
 000100     5021 0000  92B0 1000  0A14 95FF  C334 4780     96DC 41A0  C008 9200  C2F4 9200  C2F8 9200   *&...........C...*
            ST         MVI  LPR   SVC  CLI        BC       OI   LA         MVI        MVI        MVI    *........B4..B8..*
 000120     C2FC 9200  C300 94F7  A042 9101  C209 4780     9168 9102  C209 4710  9168 5810  C274 58F0   *B...C..7....B...*
                 MVI        NI         TM         BC       TM   TM         BC    TM   L          L      *....B.......B..0*
 000140     1014 9601  1017 48E0  F004 4CE0  F006 9101     1020 4710  915E 4100  E008 47F0  9162 4100   *........0.<.0...*
            LPR  OI    LPR  LH    SRP  MH    SRP  TM       LPR  BC    TM   LA         BC    TM   LA     *.....;.....0....*
 000160     E010 4110  F000 0A0A  1B44 4340  C224 5810     C224 5830  C27C 4833  000E 95FF  3002 4780   *....0...... B...*
                 LA    SRP  SVC   SR   IC         L             L          LH         CLI   LPER BC     *B...B@..........*
 000180     918C D505  3004 1004  47F0 9192  D505 301C     1004 4780  91E8 4111  000C 4640  917A 4140   *..N......0..N...*
            TM   CLC   LPER LPR   BC   TM    CLC  LPER     LPR  BC    TM   LA         BCT   TM   LA     *.....Y..... .:. *
 0001A0     000C 1B14  4140 0001  D203 1000  3010 95FF     3002 4780  91C0 D205  1004 3004  47F0 91C6   *..... ..K.......*
                 SR    LA         MVC  LPR   LPER CLI      LPER BC    TM   MVC   LPR  LPER  BC   TM     *......K......0.F*
 0001C0     D205 1004  301C 1B33  4030 96FC  D201 100A     96FC 4130  0001 9580  1002 4780  91E2 4030   *K....... ...K...*
            MVC  LPR   LPER SR    STH  OI    MVC  LPR      OI   LA         CLI   LPR  BC    TM   STH    *.............S .*
 0001E0     96FC D201  100A 96FC  5010 C224  4240 C224     9110 C208  4710 9204  9102 C208  4710 9204   *..K.....&.B.. B.*
            OI   MVC   LPR  OI    ST         STC           TM         BC   MVI   TM         BC   MVI    *..B.......B.....*
 000200     47F0 9236  5810 C224  9580 1002  4770 9236     D201 96FC  100A 4820  96FC 4122  0001 4130   *.0....B.........*
            BC   MVI   L          CLI  LPR   BC   MVI      MVC  OI    LPR  LH    OI   LA         LA     *K...............*
 000220     0001 1932  4770 922C  4122 0001  4020 96FC     D201 100A  96FC 9140  C209 4710  92B8 5820   *............ ...*
                 CR    BC   MVI   LA         STH  OI       MVC  LPR   OI   TM         BC    MVI  L      *K...... B.......*
 000240     0010 5822  0028 4832  0000 5930  92B4 4780     92B8 1233  4780 9268  9120 3012  4780 9268   *................*
                 L          LH         C     MVI  BC       MVI  LTR   BC   MVI   TM   LPER  BC   MVI    *................*
 000260     9102 3003  4710 9270  4122 0002  47F0 9246     D203 C228  C200 5820  C200 D203  2000 3010   *.............0..*
            TM   LPER  BC   MVI   LA         BC   MVI      MVC             L          MVC   LPDR LPER   *K.B.B...B.K.....*
 000280     D205 2004  301C 4122  000C 5020  C200 9640     C209 47F0  92B8 5830  C200 4143  0002 5860   *K.........&.B.. *
            MVC  LPDR  LPER LA         ST         OI            BC    MVI  L          LA         L      *B..0....B......-*
 0002A0     C224 4156  0004 4170  0001 4180  0001 47F0     9332 B002  FFFF FFFF  9108 C20C  4710 9296   *B..............0*
                 LA         LA         LA         BC       TS                    TM         BC   MVI    *..........B.....*
 0002C0     95FF C327  4780 9296  4110 C008  5010 C000     9287 C000  5810 C274  4120 C000  5021 0024   *..C.........&...*
            CLI        BC   MVI   LA                       MVI        L          LA         ST          *......B.....&...*
 0002E0                4510 92EC  000                      9280 1000  0A40                  0002 41A0   *.......
```

Figure SPZAP-5. Sample Tranlated Dump

SPZAP Examples

Example 1: Inspecting and Modifying a Load Module Containing a Single CSECT

This example shows how to inspect and modify a load module containing a single CSECT.

```
//ZAPCSECT         JOB       MSGLEVEL=(1,1)
//STEP             EXEC      PGM=xMASPZAP
//SYSPRINT         DD        SYSOUT=A
//SYSLIB           DD        DSNAME=SYS1.LINKLIB,DISP=OLD
//SYSIN            DD        *
    NAME       IEEVLNKT
    VERIFY     0018      C9C8,D2D9,D1C2,C7D5
    REP        0018      E5C6,D3D6,E6F0,4040
    SETSSI     01211234
    IDRDATA    71144
    DUMP       IEEVLNKT
/*
```

In this example:

JOB Statement

initiates the job

EXEC Statement

Invokes HMASPZAP (in VS1) or AMASPZAP (in VS2).

SYSPRINT DD Statement

defines the message data set.

SYSLIB DD Statement

defines the system library SYS1.LINKLIB containing the module IEEVLNKT that SPZAP is to process.

SYSIN DD Statement

defines the input stream.

NAME Control Statement

instructs SPZAP that the operations defined by the control statements that follow are to be performed on the module IEEVLNKT.

VERIFY Control Statement

requests that SPZAP check the hexadecimal data at offset X'0018' in the module IEEVLNKT to make sure that it is the same as the hexadecimal data specified in this statement. If the data is the same, SPZAP continues processing the subsequent statements sequentially. If the data is not identical, SPZAP will not perform the REP and SETSSI operations requested for the module. It will, however, perform the requested DUMP operation before discontinuing the processing. It will also dump a hexadecimal image of the module IEEVLNKT to the SYSPRINT data set.

REP Control Statement

causes SPZAP to replace the data at offset X'0018' in module IEEVLNKT with the data given in this control statement, provided the VERIFY statement was successful.

SETSSI Control Statement

instructs SPZAP to replace the system status information in the directory entry for module IEEVLNKT with the SSI data given in the statement, if the VERIFY statement was successful. The new SSI is to contain:

- A change level of 01.
- A flag byte of 21.
- A serial number of 1234.

IDRDATA Control Statement

causes SPZAP to update the IDR in module IEEVLNKT with the data 71144, if the REP operation is successful.

DUMP Control Statement

requests that a hexadecimal image of module IEEVLNKT be dumped to the SYSPRINT data set. Since the DUMP statement follows the REP statement, the image will reflect the changes made by SPZAP if the VERIFY operation was successful.

Example 2: Inspecting and Modifying a CSECT in a Load Module Containing Several CSECTs

This example show how to apply an IBM-supplied PTF in the form of an SPZAP fix, rather than a module replacement PTF.

```
//PTF40228          JOB       MSGLEVEL=(1,1)
//STEP              EXEC      PGM=xMASPZAP
//SYSPRINT          DD        SYSOUT=A
//SYSLIB            DD        DSNAME=SYS1.NUCLEUS,DISP=OLD
//SYSIN             DD        *
    NAME      IEANUC01  IEWFETCH
    IDRDATA   LOCFIX01
    VERIFY    01F0 47F0C018
    VERIFY    0210 5830C8F4
    REP       01F0 4780C072
    REP       0210 4130C8F4
    SETSSI    02114228
    DUMPT     IEANUC01  IEWFETCH
/*
```

JOB Statement

initiates the job.

EXEC Statement

invokes HMASPZAP (in VS1) or AMASPZAP (in VS2).

SYSPRINT DD Statement

defines the message data set.

SYSLIB DD Statement

defines the library (SYS1.NUCLEUS) that contains input module IEANUC01.

SYSIN DD Statmenet

defines the input stream that contains the SPZAP control statements.

NAME Control Statement

instructs SZAP that the operations defined by the control statements that immediately follow this statement are to be performed on the CSECT IEWFETCH contained in the load module IEANUC01.

IDRDATA Control Statement

causes SPZAP to update the IDR in module IEANUC01 for CSECT IEWFETCH with the data LOCIX01, if either of the REP operations is successful.

VERIFY Control Statements

request that SPZAP compare the contents of the locations X'01F0' and X'0210' in the control section IEWFETCH with the data given in the VERIFY control statements. If the comparisons are equal, SPZAP will continue processing subsequent control statements sequentially. However, if the data at the locations does not compare identically to the data given in the VERIFY control statements, SPZAP will dump a hexadecimal image of CSECT IEWFETCH to the SYSPRINT data set; the subsequent REP and SETSSI statements will be ignored. The DUMPT function specified will be performed before SPZAP terminates processing.

REP Control Statements

cause SPZAP to replace the data at offsets X'01F0' and X'0210' from the start of CSECT IEWFETCH with the hexadecimal data specified on the corresponding REP statements.

SETSSI Control Statement

causes SPZAP to replace the system status information in the directory for module IEANUC01 with the SSI data given in the SETSSI statement after the replacement operations have been effected. The new SSI will contain:

- A change level of 02.
- A flag byte of 11.
- A serial number of 4228.

DUMPT Control Statement

causes SPZAP to produce a translated dump for CSECT IEWFETCH of load module IEANUC01.

Example 3: Inspecting and Modifying Two CSECTs in the Same Load Module

This example shows how to inspect and modify two control sections in the same module.

```
//CHANGIT          JOB       MSGLEVEL=(1,1)
//STEP             EXEC      PGM=xMASPZAP
//SYSPRINT         DD        SYSOUT=A
//SYSLIB           DD        DSNAME=SYS1.LINKLIB,DISP=OLD
//SYSIN            DD        *
    NAME      IEFX5000  IEFQMSSS
    VERIFY    0284 4780,C096
    REP       0284 4770,C096
    IDRDATA   PTF01483
    SETSSI    01212448
    DUMPT     IEFX5000  IEFQMSSS
    NAME      IEFX5000  IEFQMRAW
    VERIFY    0154 4780,C042
    REP       0514 4770,C042
    IDRDATA   PTF01483
    SETSSI    01212448
    DUMPT     IEFX5000  IEFQMRAW
/*
```

JOB Statement

initiates the job.

EXEC Statement

invokes HMASPZAP (in VS1) or AMASPZAP (in VS2).

SYSPRINT DD Statement

defines the message data set.

SYSLIB DD Statement

defines the data set to be accessed by SPZAP while performing the operations specified by the control statements. In this case, it defines the system library SYS1.LINKLIB containing the load module IEFX5000 that is to be changed by SPZAP.

SPZA

NAME Control Statement #1

instructs SPZAP that the operations requested via the control statements immediately following it are to be performed on CSECT IEFQMSSS in load module IEFX5000 that resides in the data set defined by the SYSLIB DD statement.

VERIFY Control Statement #1

requests that SPZAP check the hexadecimal data at offset X'0284' in CSECT IEFQMSSS to make sure it is the same as the data specified in this control statement. If the data is identical, SPZAP continues processing the control statements. If the data is not identical, SPZAP will not perform the REP or SETSSI for csect iefqmsss, but it will perform the DUMPT operation. It will also provide a hexadecimal dump of CSECT IEFQMSSS.

REP Control Statement #1

causes SPZAP to replace the data at offset X'0284' in CSECT IEFQMSSS with the hexadecimal data given in this control statement.

IDRDATA Control Statement #1

causes SPZAP to update the IDR in module IEFX5000 for CSECT IEFQMSSS with the data PTF01483, if the first REP operation is successful.

SETSSI Control Statement #1

instructs SPZAP to replace the system status information in the directory entry for module IEFX5000 with the SSI data given. The new SSI will contain:

- A change level of 01.
- A flag byte of 21.
- A serial number of 2448.

DUMPT Control Statement #1

causes SPZAP to provide a translated dump of CSECT IEFQMSSS.

NAME Control Statement #2

indicates that the operations defined by the control statements that immediately follow this statement are to be performed on CSECT IEFQMRAW in the load module IEFX5000.

VERIFY Control Statement #2

requests that SPZAP perform the VERIFY function at offset X'0154' from the start of CSECT IEFQMRAW. If the VERIFY operation is successful, SPZAP will continue processing the subsequent control statements sequentially. If the VERIFY is rejected, however, SPZAP will not perform the following REP or SETSSI operations, but it will dump a hexadecimal image of CSECT IEFQMRAW to the SYSPRINT data set and perform the DUMPT operation as requested.

REP Control Statement #2

causes SPZAP to replace the data at hexadecimal offset X'0154' from the start of CSECT IEFQMRAW with the hexadecimal data that is specified in this control statement.

IDRDATA Control Statement #2

causes SPZAP to update the IDR in module IEFX5000 for CSECT IEFQMRAW with the data PTF01483, if the second REP operation is successful.

SETSSI Control Statement #2

causes SPZAP to perform the same function as the previous SETSSI, but it is performed only if the second VERIFY is not rejected.

DUMPT Control Statement #2

causes SPZAP to perform the DUMPT function on control section IEFQMRAW.

Example 4: Inspecting and Modifying a Data Record

In this example, the data set to be modified is a volume table of contents.

```
//ZAPIT        JOB        MSGLEVEL=(1,1)
//STEP         EXEC       PGM=xMASPZAP
//SYSPRINT     DD         SYSOUT=A
//SYSLIB       DD         DSNAME=FORMAT4.DSCB,DISP=OLD,
//        UNIT=3330,VOLUME=SER=111111,DCB=(KEYLEN=44)
//SYSIN        DD         *
    CCHHR            005000001
    VERIFY           2C   0504
    REP              2C   0A08
    REP              2E   0001,03000102
    ABSDUMPT         ALL
/*
```

JOB Statement

initiates the job.

EXEC Statement

invokes HMASPZAP (in VS1) or AMASPZAP (in VS2).

SYSPRINT DD Statement

defines the message data set.

SYSLIB DD Statement

defines the data set to be accessed by SPZAP in performing the operations specified by the control statements. In this example, it defines the VTOC (a Format 4 DSCB) on a 3330 volume with a serial number of 111111. DCB=(KEYLEN=44) is specified so that the dump produced by the ABSDUMPT control statement will show the dsname which is a 44 byte key. Note that this is not necessary for the VERIFY and REP control statements.

CCHHR Control Statement

indicates that SPZAP is to access the direct access record address "0005000001" in the data set defined by the SYSLIB DD statement while performing the operations specified by the following control statements.

SPZAP

VERIFY Control Statement

requests that SPZAP check the data at hexadecimal displacement X'2C' from the start of the data record defined in the CCHHR statement to make sure it is the same as the hexadecimal data specified in this control statement. If the data is the same, SPZAP continues processing the following control statements sequentially. If the data is not identical, SPZAP will not perform the REP function but will perform the ABSDUMPT operation; it will also dump a formatted hexadecimal image of the data record defined by the CCHHR statement to the SYSPRINT data set.

REP Control Statements

cause the eight bytes of data starting at displacement 2C from the beginning of the record to be replaced with the hexadecimal data in the REP control statements. The 2C displacement value allows for a 44-byte key at the beginning of the record.

ABSDUMPT Control Statement

causes SPZAP to dump the entire data set to the SYSPRINT data set. Since DCB=(KEYLEN=44) is specified on the SYSLIB DD statement, the 44 byte dsname will also be dumped.

Note: If the VTOC is to be modified, message HMA117D or AMA117D will be issued to the operator, requesting permission for the modification.

Example 5: Entering SPZAP Control Statements Through the Console

This example shows how to enter SPZAP control statement through the console.

```
//CONSOLIN          JOB        MSGLEVEL=(1,1)
//STEP              EXEC       PGM=xMASPZAP
//SYSPRINT          DD         SYSOUT=A
//SYSLIB            DD         DSNAME=SYS1.LINKLIB,DISP=OLD
//SYSIN             DD         *
         CONSOLE
/*
```

JOB Statement

initiates the job.

EXEC Statement

invokes HMASPZAP (in VS1) or AMaSPZAP (in VS2).

SYSPRINT DD Statement

defines the message data set.

SYSLIB DD Statement

defines the data set that contains the module to be updated.

SYSIN DD Statement

defines the input stream.

CONSOLE Control Statement

indicates that SPZAP control statements are to be entered through the console.

Example 6: Using the BASE Control Statement for Inspecting and Modifying a Load Module

This example shows how to inspect and modify a CSECT whose starting address does not coincide with assembly listing location zero.

```
//MODIFY      JOB       MSGLEVEL=(1,1)
//STEP        EXEC      PGM=xMASPZAP
//SYSPRINT    DD        SYSOUT=A
//SYSLIB      DD        DSNAME=SYS1.LINKLIB,DISP=OLD
//SYSIN       DD        *
    NAME           IEFMCVOL  IEFCVOL2
    BASE           0398
    IDRDATA        MOD04
    VERIFY         039A 5820C010
    REP            039A 47000000
    DUMP           IEFMCVOL  IEFCVOL2
/*
```

JOB Statement

initiates the job.

EXEC Statement

invokes HMASPZAP (in VS1) or AMASPZAP (in VS2).

SYSPRINT DD Statement

defines the message data set.

SYSLIB DD Statement

defines the data set to be accessed by SPZAP when performing the operations requested via the control statements. In this case, it defines the system library, SYS1.LINKLIB, that contains the module IEFMCVOL in which the CSECT to be changed, IEFCVOL2, resides.)

SYSIN DD Statement

defines the input stream that contains the SPZAP control statements.

NAME Control Statement

instructs SPZAP that the operations defined by the control statements that immediately follow it are to be performed on CSECT IEFCVOL2 in the load module IEFMCVOL.

BASE Control Statement

provides SPZAP with a base value that is to be used to readjust the offsets on the VERIFY and REP statements that follow it.

IDRDATA Control Statement

causes SPZAP to update the IDR in module IEFMCVOL for CSECT IEFCVOL2 with the data MOD04, the the REP operation is successful.

VERIFY Control Statment

requests that SPZAP inspect the data at offset X'039A'. The base value X'0398' given in the previous BASE statement is subtracted from this offset to determine the proper displacement of the data within CSECT IEFCVOL2. Therefore, SPZAP checks the data at the location that is actually displaced X'0002' bytes from the beginning of CSECT IEFCVOL2 to ensure that it is the same as the hexadecimal data specified in this control statement.

If the data is the same, SPZAP continues processing the following statements in the order in which they are encountered. If the data is not identical, will not perform the REP, SETSSI, or IDRDATA functions, but it will perform the DUMP operation; it will also dump a hexadecimal image of CSECT IEFCVOL2 to the SYSPRINT data set.

REP Control Statement

causes SPZAP to replace the data at displacement X'0002' (offset 039A minus base value 0398) into CSECT IEFCVOL2 with the hexadecimal data specified in this control statement.

DUMP Control Statement

requests that SPZAP dump a hexadecimal image of CSECT IEFCVOL2 to the SYSPRINT data set. Since the DUMP statement follows the REP statement, the image will reflect the changes made by SPZAP (assuming no verification has been rejected).

Appendix A: Writing EDIT User Programs ————→

Tells how to write and use EDIT exit routines and format appendages.

APP A

Contents

Figures

APP A

Introduction

You may want to code special programs to supplement GTF and PRDMP/EDIT operation. EDIT allows for two types of user programs: exit routines and format appendages. Neither type may occupy more than 10K bytes of main storage.

- An exit routine allows you to inspect each input trace record before EDIT begins processing it; on the basis of the inspection you must decide whether EDIT should process the record normally or take special action.

- A format appendage allows you to format all user trace records of a specified type. A format appendage must be named HMDUSRxx (for VS1) or AMDUSRxx (for VS2), where xx is the hexadecimal form of the format identifier (FID) specified in the GTRACE macro when the record was created. (Note that in VS1 EDIT will also accept format appendages named IMDUSRxx; thus format appendages that were originally written for use with OS/MFT or OS/MVT need not necessarily be rewritten for use with OS/VS. Similarly, in VS2 EDIT will also accept format appendages named HMDUSRxx as well as those named AMDUSRxx or IMDUSRxx.)

This appendix is designed to help you write efficient, helpful user programs.

Guaranteeing Cross-System Compatibility for Format Appendages

To make sure that an OS/MFT or OS/MVT format appendage is upward compatible with OS/VS operation, or that a VS1 format appendage is upward compatible with VS2 operation, reassemble the module containing the appendage. If your format appendage depends on specific fields in a trace record, be sure that it will not be affected by differences in record format between systems.

User Program Interfaces

A user program interfaces with the EDIT function of PRDMP in the following ways:

Gaining Control

Until EDIT calls them, user programs reside in SYS1.LINKLIB or in a data set defined by the JOBLIB or STEPLIB DD statement. Once a user program is loaded into main storage, it remains there until EDIT processing is complete.

An exit routine is named in the EXIT= parameter of the EDIT control statement. It gets control every time EDIT reads an input trace record, and always completes its examination of the record before EDIT processes it.

A format appendage is invoked only when EDIT encounters a record that contains an FID field corresponding to the name of the format appendage. It remains in main storage until deleted, but only gets control when EDIT encounters a record with the corresponding FID.

Using the Parameter List

When EDIT passes control to a user program, register 1 contains the address of a parameter list. The contents of that parameter list, and its related fields are shown in figure APNDX-1. The exit routine or format appendage uses the parameter list to find the record it is to process, determine how to process it, and decide where to put the processed record.

Input record

As shown in Figure APNDX-1, the first four bytes of the parameter list give the address of the input record. Four-byte fields at offset 12 and 16, respectively, point to the event identifier (EID) field and the data area in the input record.

For a complete description of the input record format, see Figure GTF-8 in Chapter 1: GTF (Generalized Trace Facility).

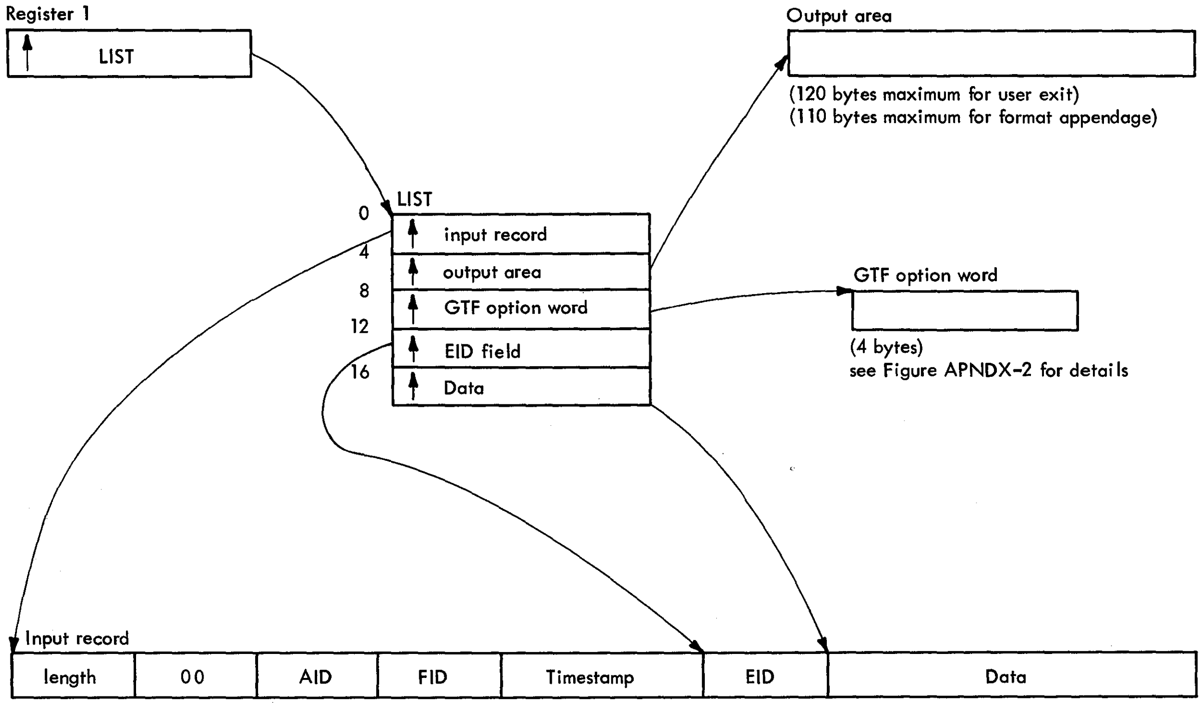


Figure APNDX-1. EDIT Parameter List and Related Fields

APP A

GTF Option Word

A four-byte field at offset 8 in the parameter list gives the address of the GTF option word, a four-byte table that summarizes the GTF options in effect when the input trace records were produced. Figure APNDX-2 lists the contents of the GTF option word.

BYTES	BITS	OPTIONS IN EFFECT DURING TRACE
Byte 1	1...	SYSM-- minimal tracing for system events
	.1..	SYSP-- maximum tracing, prompting requested.
	..1.	SYS-- maximum tracing for system events
	...1	USR -- all GTRACE-generated interrupts traced
	.... 1...	TRC -- all GTF interrupts traced
	.... .1..	DSP -- all task-switches traced
	.... ..x.	Reserved
	1	PCI -- program-controlled interrupts traced
Byte 2	1...	SVC -- all SVC interrupts traced
	.1..	SVCP -- SVC interrupts selected by prompting
	..1.	SIO -- all SIO events traced
	...1	SIOP -- SIO events selected by prompting
	.... 1...	PI -- all program interrupts traced
	.... .1..	PIP -- program interrupts selected by prompting
	.... ..1.	IO -- all I/O interrupts traced
	1	IOP -- I/O interrupts selected by prompting
Byte 3	1...	EXT -- external interrupts traced
	.1..	GTFSYYNC - message HHL118I or AHL118I has been issued
	..1.	GTFAOS1 - OS/VS1 system
	...1	GTFAOS2 - OS/VS2 System
	.... xxx.	reserved bits
	1	IO=SIO -- identical devices selected for IO & SIO
Byte 4	xxxx	Reserved
	.... 1...	Monitor Call instruction (always on)
	.... .x..	Reserved
	.... ..1.	Tracing system has time-of-day clock
	1	user timestamp requested

Figure APNDX-2. Contents of GTF Option Word, showing GTF Options in Effect During Trace

For more information about any of the GTF options, refer to Chapter 1, GTF (Generalized Trace Facility).

Returning to EDIT

A user program must return to EDIT with one of the return codes listed below. If EDIT recieves an invalid return code from a user program, it takes action as specified by the ER= subparameter of the PARM= parameter of the EXEC statement that invokes PRDMP. This parameter, its values and their meanings are described in Chapter 5: PRDMP in the section "Job Control Language Statements".

Exit Routine Return Codes

An exit routine must return to EDIT with one of the following return codes:

Code Meaning

0 EDIT should print the contents of the output area, clear the area, and return immediately to the exit routine. This allows the exit routine to print more than one line of output. (Note that the output buffer may be in a different location when the format appendage receives control again.)

4 EDIT should print the contents of the output area and obtain the next logical record.

8 EDIT should format and print the trace record according to the selectivity specified in the EDIT control statement.

12 EDIT should obtain the next logical input trace record without printing the contents of the output buffer.

16 EDIT should print the contents of the output buffer and no longer invoke the exit routine, which is no longer needed.

20 EDIT should format and print the trace record according to the selectivity specified in the EDIT control statement, and should no longer invoke the exit routine, which is no longer needed.

24 EDIT should terminate processing and return control to PRDMP so that the next PRDMP control statement may be processed.

28 EDIT should format and print this record as though no selectivity had been specified in the EDIT control statement.

Format Appendage Return Codes

A format appendage must return to EDIT with one of the following return codes:

Code Meaning

0 EDIT should print the contents of the output buffer and return immediately to the format appendage. (Note that the output buffer may be in a different location when the format appendage receives control again.)

4 EDIT should print the contents of the output buffer and obtain the next logical input trace record.

8 EDIT should obtain the next logical input trace record without printing the contents of the output buffer.

Handling Errors

EDIT is prepared to handle three types of errors: failures in finding or loading a user program, invalid return codes, and program checks. Other types of errors and their consequences are discussed later in this appendix, in the section "Avoiding Unrecoverable Errors".

Errors in Finding or Loading a User Program

There are three probable reasons why EDIT should fail to find or load a user program:

- The program was incorrectly identified in the EXIT= parameter of the EDIT control statement (for exit routines) or in the FID parameter of the GTRACE macro (for format appendages).

- The program did not reside in the designated library.

- The program was larger than 10K bytes. EDIT will not load an exit routine or format appendage that that exceeds this maximum, but will issue this message:

```
{HMD229I} MODULE mod EXCEEDS 10K LIMIT
{AMD229I}
```

If, for one of these reasons, EDIT cannot find or load a user program, it takes action as shown in Figure APNDX-3.

Error Input Type	Exit Routine		Format Appendage	
	Not Found	Not Loaded	Not Found	Not Loaded
Dump	A	A	B	B
Trace Data Set	A	A	B	A

Action A: EDIT terminates processing and returns control to PRDMP, which obtains the next PRDMP control statement.

Action B: EDIT dumps the associated record in hexadecimal and obtains the next input trace record. Any subsequent records that have the same FID will be dumped in hexadecimal.

Figure APNDX-3. EDIT Actions in Response to Errors in Finding or Loading User Programs.

Invalid Return Codes and Program Checks

EDIT's action in response to invalid return codes and program checks depends on the value for ER= that you specify in the PARM= parameter of the EXEC statement that invokes PRDMP. For an explanation of the valid values for ER=, refer to the section "Job Control Language Statements" in Chapter 5: PRDMP.

Avoiding Unrecoverable Errors

As shown in the previous sections, EDIT can recover from three kinds of errors: failures in finding or loading a user program, invalid return codes, and program checks. EDIT cannot protect you, however, against errors that you may generate, for example by performing I/O operations or issuing GETMAIN macro instructions. In fact, you should avoid issuing any SVCs in your user program. Ordinarily this is not difficult, since EDIT provides you with the ability to examine records, manipulate data, and request formatted output to be printed. If you must issue an SVC, EDIT will permit you to do so; you should be prepared, however, for possibly unpredictable results if an error occurs during an operation that you have requested by issuing an SVC.

Another type of error arises when you invoke more format appendages than EDIT has room for: When EDIT needs more room to load a new format appendage, it deletes all previously loaded format appendages and starts loading format appendages again as needed.

Deletion of a format routine is critical if the deleted program issues an OPEN because the reinitialization that is necessary when the program is reloaded can cause two DCBs to be open at the same time. (Note that you need not worry about deletion of an exit routine. EDIT provides a separate 10K block of storage for an exit routine, so that an exit routine will never be deleted until EDIT processing is terminated.)

To avoid running out of space for your format appendages, increase the size of your partition or region. Remember that EDIT allows you 10K out of the first 128K, and 50K out of all subsequent partition or region size increases of 64K. In other words, if your partition or region size is 128K, you have room for one format appendage. If your partition or region size is 192K, you have room for six format appendages, and so on.

If your format appendage must issue a GETMAIN macro, be sure to specify a paritition or region large enough to include the amount of storage needed. When you no longer need the extra storage, be sure to issue a FREEMAIN macro for all storage that you reserved for your own use. If you do not do this, and your format appendage is deleted, the storage you reserved will remain allocated to you and thus will be unavailable to subsequent users.

A few examples may further clarify the areas in which EDIT does not provide error recovery:

- A user program, known as module A, issues the LINK SVC for module B. A program check occurs in module B. EDIT will attempt error recovery, since the error is a program check, but it knows nothing about module B. Therefore when it produces diagnostic information it will give the entry point of module A as the entry point of the failing module, and attribute the registers at the time of the program check to module A.

- A user program issues the OPEN SVC (SVC X'13') unsuccessfully and is posted with a system completion code of 213. EDIT cannot recover, so EDIT, the user program and IMDPRDMP will all be terminated.

- A user program opens a DCB. Before it can close the DCB, the program is deleted to make room for another user program. When the deleted program is reloaded, it creates a new DCB and opens it. Thus there are two open DCBs with the same name in storage at the same time. The operating system will not tolerate this situation, so the user program is abnormally terminated.

- A user program issues the SPIE SVC, thereby nullifying EDIT's SPIE routine. As a result any program checks in the user program that EDIT would normally handle will go through the user's own SPIE routine, perhaps with unpredictable results.

Sample User Exit

Figure APNDX-4 shows a sample exit routine. This routine, named ABENDXIT, was written to aid diagnosis of an abnormal termination condition in a particular job. It scans each input trace record, suppressing printing until it finds a record with the specified jobname. When it finds such a record, ABENDXIT signals PRDMP to print that record. All subsequent records will be printed until ABENDXIT encounters an SVC 13 record for the specified jobname; then ABENDXIT instructs PRDMP to print that record and terminate.

```
**********************************************************************
*  ABENDXIT IS AN EDIT USER EXIT ROUTINE DESIGNED TO CONTROL PRINTING
*  OF ALL GTF RECORDS ASSOCIATED WITH A PROGRAM THAT HAS
*  PROGRAM CHECKED AND ABENDED
**********************************************************************
ABENDXIT CSECT
*    EQUATE STATEMENTS
FRSTREG        EQU  0
PARMREG        EQU  1
EIDREG         EQU  2
DATAREG        EQU  3
WORKREG        EQU  4
CHAINREG       EQU  9
BASE           EQU  12
SAVEPTR        EQU  13
RETPTR         EQU  14
CODEREG        EQU  15
         STM  RETPTR,BASE,12(SAVEPTR)  STORE REGISTERS
         BALR BASE,0                   ESTABLISH ADDRESSABILITY
         USING *,BASE                  USING REGISTER 12
         ST   SAVEPTR,SAVE+4           BACKWARD CHAINING
         LA   CHAINREG,SAVE            MY SAVE AREA POINTER
         ST   CHAINREG,8(SAVEPTR)      FORWARD CHAINING
         LR   SAVEPTR,CHAINREG         REG 13 ADDRESSES SAVE AREA
         IMDMEDIT                 SYMBOLIC EID MACRO
+*/*******************************************************************/
+*/*  THE IMDMEDIT MACRO MAPS THE EID VALUES ASSOCIATED WITH IBM    */
+*/*  SYSTEM AND SUBSYSTEM EVENTS.  THE STORAGE FOR ANY OR ALL OF   */
+*/*  THE MAPPED VALUES MUST BE CONTAINED IN THE MODULE REFERENCING */
+*/*  THE DESIRED EIDS.  IMDMEDIT IS DESIGNED TO BE USED BY IBM-    */
+*/*  SUPPLIED FORMAT APPENDAGES, AND USER-SUPPLIED USER EXIT       */
+*/*  MODULES.                                                      */
+*/*******************************************************************/
+IMDMPCI EQU  X'2000' PCI I/O INTERRUPT
+IMDMSVC EQU  X'0100' SVC INTERRUPT
+IMDMDSP EQU  X'6000' TASK SWITCH
+IMDMIO1 EQU  X'7100' I/O INTERRUPT
+IMDMIO2 EQU  X'7101' I/O INTERRUPT
+IMDMSIO EQU  X'7000' SIO OPERATION
+IMDSSM1 EQU  X'5100' PI 19
+IMDMSSM EQU  X'0000' SSM INTERRUPT
+IMDPIPG EQU  X'5000' PI 17
+IMDMPI  EQU  X'5101' PROGRAM INTERRUPT
+IMDMEXT EQU  X'5200' EXTERNAL INTERRUPT
+IMDMDMA1 EQU X'EFFF' OPEN/CLOSE/EOV
         TM   TERMSW,X'01'        Q/HAS TERMINATION BEEN REQSTD
         BC   1,FINISH            YES,TELL EDIT TO TERMINATE
         L    EIDREG,12(PARMREG)  GET POINTER TO EID
```

Figure APPNDX-4. Sample Exit Routine (Part 1 of 2)

APP A

```
         L    DATAREG,16(PARMREG) GET POINTER TO DATA(JOBNAME)
         TM   PRINTSW,X'01'       Q/HAS JOBN ALREADY BEEN FOUND
         BC   1,PRINTALL          YES, SO PRINT THIS RECORD
         LA   WORKREG,0           GET ZERO CONSTANT
         C    WORKREG,ECB1        Q/HAS THIS ECB BEEN POSTED
         BC   7,MYJOBLAB          YES, CHECK IF JOBN FOUND
         WTOR 'SPECIFY 8-CHARACTER JOBNAME OF ABENDING PROGRAM',
              MYJOBN,8,ECB1
         WAIT ECB=ECB1
         LA   WORKREG,MYJOBN      ADDRESS OF JOBNAME SELECTED
         OC   0(8,WORKREG),BLANKS CONVERT LOWER-CASE CHARS TO
*                                 UPPER CASE
MYJOBLAB CLC  0(8,DATAREG),MYJOBN Q/IS THIS MY JOBNAME
         BC   7,NOPRINT           NO -- JUST RETURN
*  ONCE JOBNAME FOUND| SET SWITCH AND PRINT ALL RECORDS UNTIL
*  ENCOUNTER AN SVC 13 (ABEND) CONTAINING THIS JOBNAME
         OI   PRINTSW,X'01'       TURN ON JOBNAME FOUND SWITCH
PRINTALL CLC  0(2,EIDREG),SVCEID  Q/ IS THIS AN SVC RECORD
         BC   7,PRINTREC          NO, SO PRINT AND CONTINUE
         CLI  15(DATAREG),X'0D'   Q/IS THIS AN SVC 13 (ABEND)
         BC   7,PRINTREC          NO, SO PRINT AND CONTINUE
         CLC  0(8,DATAREG),MYJOBN Q/IS THIS MY JOBNAME
         BC   7,PRINTREC          NO, SO PRINT AND CONTINUE
EXIT     OI   TERMSW,X'01'        INDICATE THAT THIS IS LAST
*                                 RECORD TO BE PRINTED
PRINTREC LA   CODEREG,8           FORMAT AND PRINT THIS RECORD
RETURN   L    SAVEPTR,4(SAVEPTR)  RESTORE SAVE AREA POINTER
         L    RETPTR,12(SAVEPTR)  RESTORE REGISTER 14
         LM   FRSTREG,BASE,20(SAVEPTR) RESTORE OTHER REGS EXCEPT 15
         BCR  15,RETPTR           RETURN TO EDIT
FINISH   LA   CODEREG,24          TERMINATE EDIT PROCESSING
*                                 SINCE SVC 13 WAS LAST RECORD
         B    RETURN              RESTORE REGISTERS AND RETURN
NOPRINT  LA   CODEREG,12          IGNORE RECORD
         B    RETURN              RESTORE REGISTERS AND RETURN
SAVE     DC   18F'0'              SAVE AREA
SVCEID   DC   AL2(IMDMSVC)        ESTABLISH REAL AREA FOR
*                                 EID FROM IMDMEDIT MAP MACRO
TERMSW   DC   X'00'               INDICATION TO REQUEST TERM
PRINTSW  DC   X'00'               JOBN FOUND, SO PRINT REC IND
ECB1     DC   F'0'                FOR POST
MYJOBN   DC   C'        '         PLACE FOR OPR TO PUT JOBNAME
BLANKS   DC   C'        '         TO CONVERT LOWER TO UPPER CASE
         END
/*
```

Figure APNDX-4. Sample Exit Routine. (Part 2 of 2)

Some instructions in the sample exit routine require special attention. These are shaded in Figure APNDX-4, and they are discussed below.

```
IMDMEDIT
```

This mapping macro expands, as shown, into a list of equate statements that supply symbolic names for the event identifiers (EIDs). You should use the symbolic name in your program; this is your protection against program failure, if for any reason, the EID values are later changed.

```
L    EIDREG,12(PARMREG)
```

```
L    DATAREG,16(PARMREG)
```

These two instructions access the EDIT parameter list. (See Figure APNDX-1.)

```
WTOR 'SPECIFY 8-CHARACTER JOBNAME OF ABENDING PROGRAM', MYJOBN,8,ECB1
```

This instruction requests information that cannot be obtained from the EDIT parameter list. You can use a WTOR to request any information that the operator is likely to have, such as the EDIT options in effect. Note, however, that when you issue an SVC in a user program you risk abnormal termination if an error occurs during the SVC operation. For more information about this point, refer to the section "Avoiding Unrecoverable Errors" eariler in this chapter.

```
SVCEID    DC    AL2(IMDMSVC)
```

This establishes a main storage location for the value equated to IMDMSVC in the expansion of the IMDMEDIT mapping macro.

Sample Format Appendage

Figure APNDX-5 shows how to use the EDIT parameter list and how to handle multiple EIDs. It consists of excerpts from a sample format appendage named HMDUSR01, which formats three different types of user records. For each record HMDUSR01 produces two lines of output. The first line varies according to the record type. The second line is the same for all records.

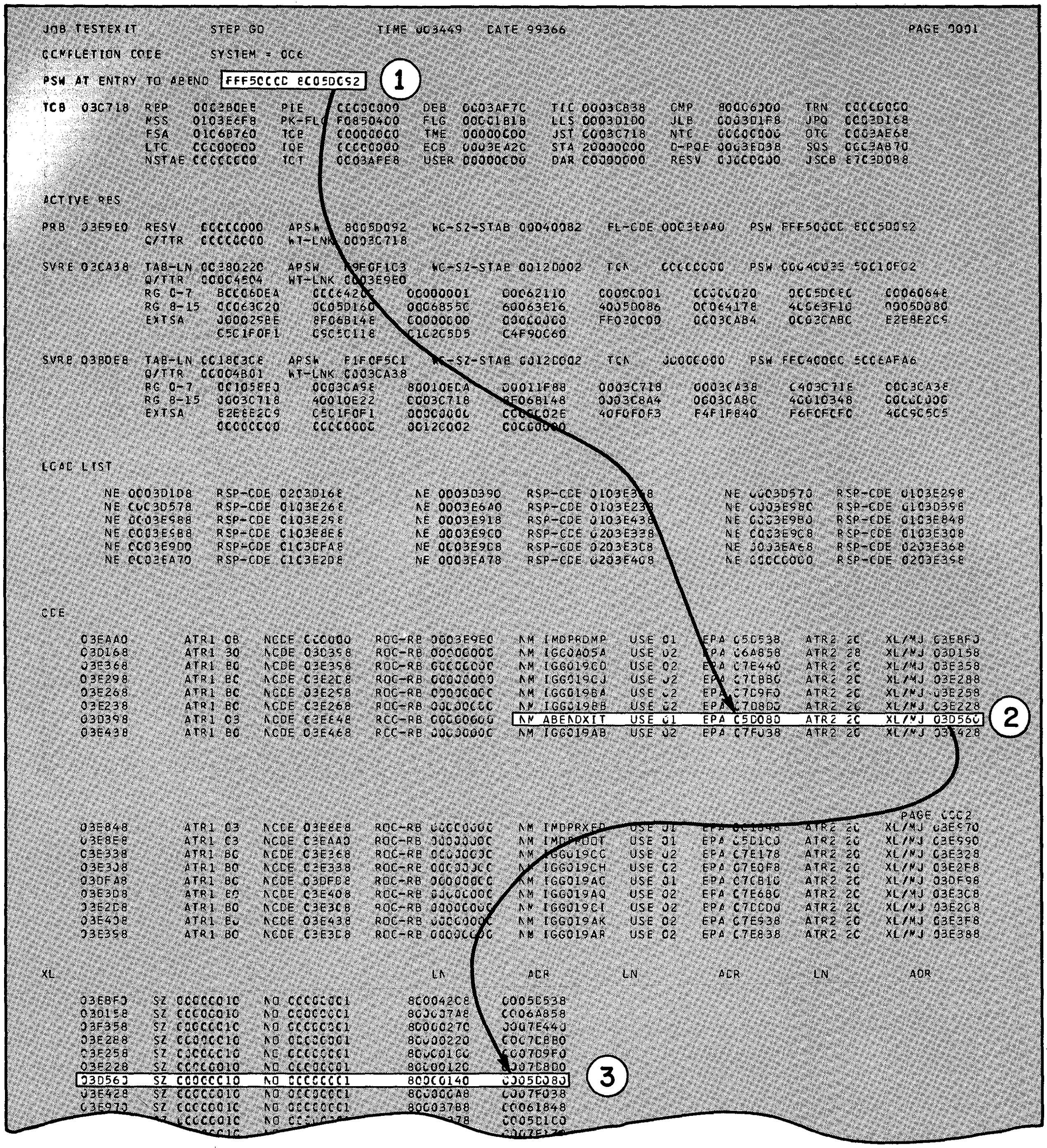
Note that HMDUSR01 is a valid format appendage in VS1 only, because of the H prefix. To write a similar format appendage valid for VS2, simply change the name on the CSECT statement to AMDUSR01. See the section "Guaranteeing Cross-System Compatibility for Format Appendages".

```
*********************************************************************
*   HMDUSR01 IS AN EDIT USER FORMAT APPENDAGE MODULE THAT PROCESSES
*   THREE DIFFERENT TYPES OF INPUT RECORDS, THUS, THREE DIFFERENT EIDS.
*   LINE ONE OF THE FORMATTED OUTPUT VARIES ACCORDING TO THE EID.  LINE
*   TWO OF THE FORMATTED OUTPUT IS THE SAME FOR ALL EIDS, AND IS
*   PRODUCED IN COMMON CODE.
*********************************************************************
HMDUSR01 CSECT
*    EQUATE STATEMENTS
FRSTREG  EQU  0
PARMREG  EQU  1
EIDREG   EQU  2
DATAREG  EQU  3
CHAINREG EQU  9
BASE     EQU  12
SAVEPTR  EQU  13
RETPTR   EQU  14
CODEREG  EQU  15
         STM  RETPTR,BASE,12(SAVEPTR)  STORE REGISTERS
         BALR BASE,0                   ESTABLISH ADDRESSABILITY
         USING *,BASE                  USING REGISTER 12
         ST   SAVEPTR,SAVE+4           BACKWARD CHAINING
         LA   CHAINREG,SAVE            MY SAVE AREA POINTER
         ST   CHAINREG,8(SAVEPTR)      FORWARD CHAINING
         LR   SAVEPTR,CHAINREG         REG 13 ADDREESES SAVE AREA
         L    EIDREG,12(PARMREG)       GET POINTER TO EID
         L    DATAREG,16(PARMREG)      GET POINTER TO FIRST LINE DATA
         TM   SWITCH,X'01'             Q/ HAS FIRST LINE BEEN OUTPUTTED
         BC   1,LINETWO                YES, BRANCH TO FORMAT LINE TWO
*                  WHICH IS COMMON TO ALL THREE EID RTNS
         CLC  0(2,EIDREG),EID1         NO--Q/IS THIS A RECORD WITH EID1
         BC   8,RTN1                   YES--FORMAT LINE ONE
         CLC  0(2,EIDREG),EID2         Q/IS THIS A RECORD WITH EID2
         BC   8,RTN2                   YES--FORMAT LINE ONE
         CLC  0(2,EIDREG),EID3         Q/IS THIS A RECORD WITH EID3
         BC   8,RTN3                   YES--FORMAT LINE ONE
         LA   CODEREG,8                NO--IF NONE OF THESE EIDS, IGNORE
         B    RETURN                   REC, RESTORE REGS, AND RETURN
    .
    .
    .
RTN1
    .
    .
    .
         B    ZEROCODE                 SET ZERO RETURN CODE
```

Figure APNDX-5. Sample Format Appendage (Part 1 of 2)

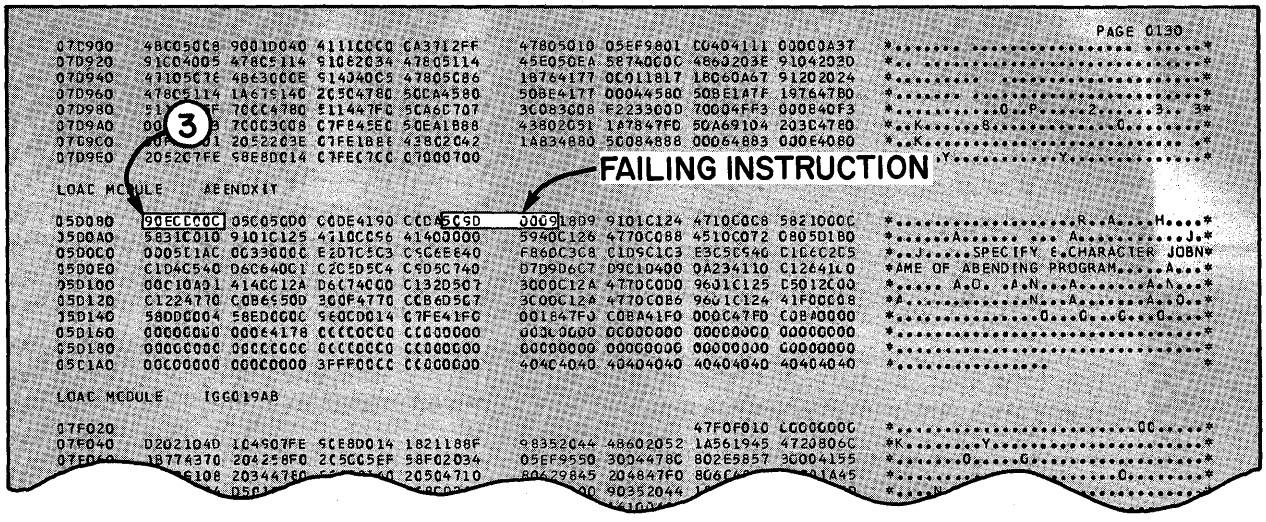
```
RTN2
     .
     .
     .
         B    ZEROCODE                  SET ZERO RETURN CODE
RTN3
     .
     .
     .
ZEROCODE OI   SWITCH,X'01'              FIRST LINE COMPLETE INDICATOR
         SR   CODEREG,CODEREG           OUTPUT THIS LINE AND RETURN
*                  IMMEDIATELY TO THIS FORMAT APPENDAGE
         B    RETURN                    RESTORE REGISTERS AND RETURN
LINETWO
     .
     .
     .
         NI   SWITCH,X'FE'              TURN OFF LINE 2 INDICATOR
         LA   CODEREG,4                 OUTPUT THIS LINE--COMPLETE
RETURN   L    SAVEPTR,4(SAVEPTR)        RESTORE SAVE AREA POINTER
         L    RETPTR,12(SAVEPTR)        RESTORE REGISTER 14
         LM   FRSTREG,BASE,20(SAVEPTR)  RESTORE OTHER REGS EXCEPT 15
         BCR  15,RETPTR                 RETURN TO EDIT

SAVE     DC   18F'0'                    REGISTER SAVE AREA
SWITCH   DC   X'00'                     READY FOR LINE TWO SWITCH
EID1     DC   X'E001'                   EID1
EID2     DC   X'E002'                   EID2
EID3     DC   X'E003'                   EID3
     .
     .
     .
         END
/*
```

Figure APNDX-5.   Sample Format Appendage (Part 2 of 2)

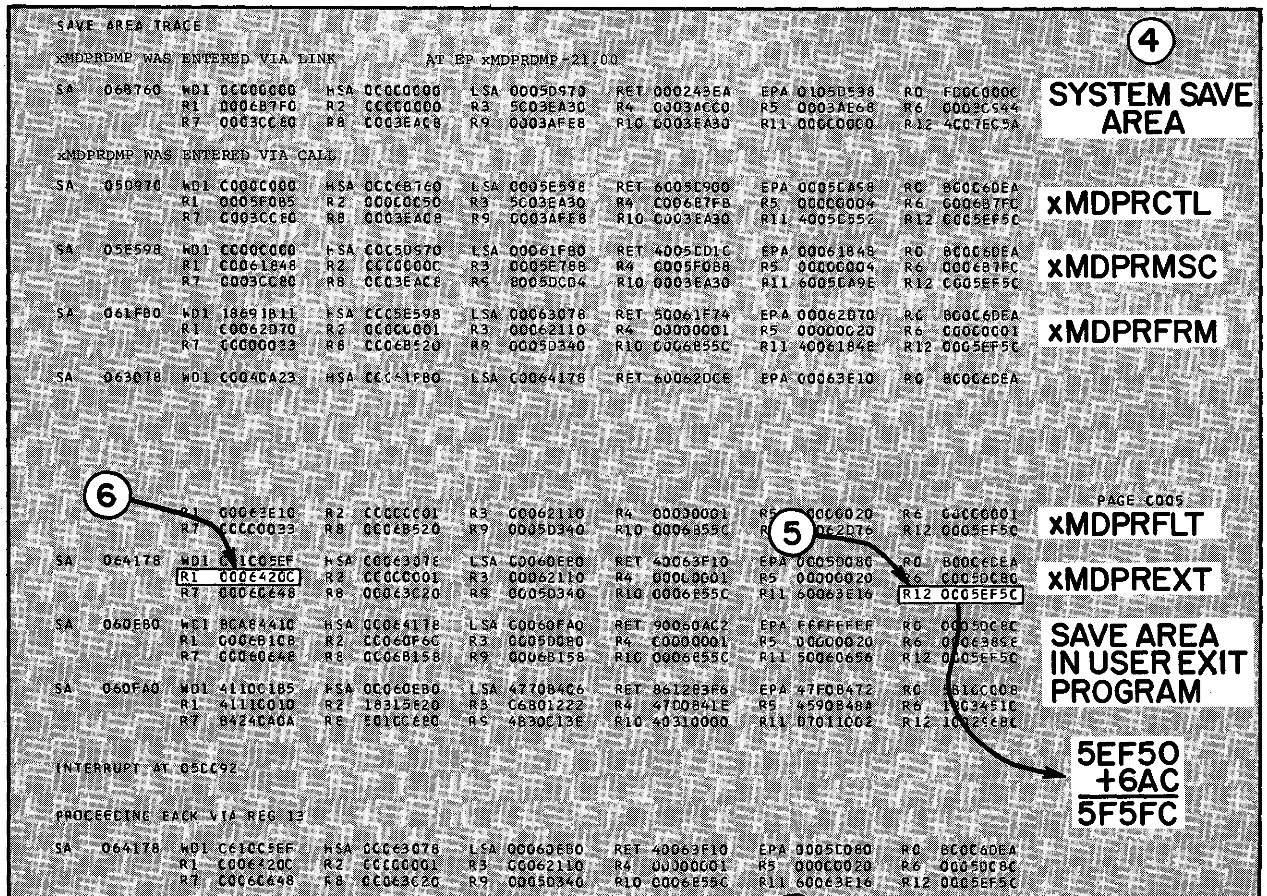


```
JOB TESTEXIT        STEP GO             TIME 003449   DATE 99366                                   PAGE 0001

COMPLETION CODE        SYSTEM = 0C6

PSW AT ENTRY TO ABEND  FFF50CCD 8C05DC92

TCB  03C718  RBP    0003B0E8   PIE     00000000   DEB  0003AF7C   TIC 0003C838   CMP   800C6000   TRN  00000000
             MSS    0103E6F8   PK-FLG  F0850400   FLG  000C1818   LLS 0003D1D0   JLB   0003D1F8   JPQ  0003D168
             FSA    01C6B760   TCB     00000000   TME  00000000   JST 0003C718   NTC   00000000   OTC  0003AE68
             LTC    00000000   IQE     00000000   ECB  0003E42C   STA 20000000   D-PQE 0003ED38   SQS  0003AB70
             NSTAE  00000000   TCT     0003AFE8   USER 00000000   DAR 00000000   RESV  00000000   JSCB 8703D088

ACTIVE RBS

PRB  03E9E0  RESV   00000000   APSW   8005D092   WC-SZ-STAB 00040082   FL-CDE 0003EAA0   PSW FFF500CD 8C05D092
             Q/TTR  00000000   WT-LNK 0003C718

SVRB 03CA38  TAB-LN 00380220   APSW   F9F0F1C3   WC-SZ-STAB 0012D002   TQN    00000000   PSW 0004C033 50C10FC2
             Q/TTR  00004504   WT-LNK 0003E9E0
             RG 0-7    BC006DEA   0C06420C   00000001   00062110   00000001   00000020   0005DCEC   0006064E
             RG 8-15   00063C20   0C05D160   0006855C   60063E16   4005D086   00064178   40063F10   0005D080
             EXTSA     J00029BE   8F06B148   C0000000   00000000   FF030C00   0003CAB4   0003CABC   E2E8E2C9
                       C5C1F0F1   C9C5C118   C1C2C5D5   C4F90C60

SVRB 03B0E8  TAB-LN 0018C3C8   APSW   F1F0F5C1   WC-SZ-STAB 0012C002   TQN    00000000   PSW FFC400C0 5CC6AFA6
             Q/TTR  00004B01   WT-LNK 0003CA38
             RG 0-7    0C1058EJ   0003CA9E   80010ECA   00011F88   0003C718   0003CA38   C403C71E   0003CA38
             RG 8-15   J003C718   40010E22   C003C718   8F068148   0003C8A4   0003CABC   40010348   00000000
             EXTSA     E2E8E2C9   C5C1F0F1   00000000   C000C02E   40F0F0F3   F4F1F840   F6F0F0F0   40C9C5C5
                       00000000   00000000   0012C002   00000000

LOAD LIST

        NE 0003D1D8   RSP-CDE 0203D168        NE 00030390   RSP-CDE 0103E368        NE 0003D570   RSP-CDE 0103E298
        NE 0003D578   RSP-CDE 0103E268        NE 0003E6A0   RSP-CDE 0103E238        NE 0003E98C   RSP-CDE 0103D398
        NE 0003E988   RSP-CDE C103E29E        NE 0003E918   RSP-CDE 0103E438        NE 0003E9B0   RSP-CDE 0103E848
        NE 0003E9B8   RSP-CDE C103E8E8        NE 0003E9C0   RSP-CDE 0203E338        NE 0003E9C8   RSP-CDE 0103E308
        NE 0003E9D0   RSP-CDE C1C3DFA8        NE 0C03E9C8   RSP-CDE 0203E3C8        NE J003EA68   RSP-CDE 0203E368
        NE 0C03EA70   RSP-CDE C1C3E2D8        NE 0003EA78   RSP-CDE 0203E408        NE 00000000   RSP-CDE 0203E398

CDE

     03EAA0       ATR1 0B   NCDE 000000   ROC-RB 0003E9E0   NM IMDPRDMP   USE 01   EPA 05D538   ATR2 2C   XL/MJ 03EBF0
     03D168       ATR1 30   NCDE 03D398   ROC-RB 00000000   NM IGC0A05A   USE 02   EPA 06A858   ATR2 28   XL/MJ 03D158
     03E368       ATR1 B0   NCDE 03E398   ROC-RB 00000000   NM IGG019CD   USE 02   EPA 07E440   ATR2 2C   XL/MJ 03E358
     03E298       ATR1 BC   NCDE C3E2C8   ROC-RB 00000000   NM IGG019CJ   USE 02   EPA 07CBB0   ATR2 2C   XL/MJ 03E288
     03E268       ATR1 BC   NCDE 03E298   ROC-RB 00000000   NM IGG019BA   USE 02   EPA 07D9F0   ATR2 2C   XL/MJ 03E258
     03E238       ATR1 BC   NCDE C3E268   ROC-RB 00000000   NM IGG019BB   USE 02   EPA 07D8D0   ATR2 20   XL/MJ 03E228
     03D398       ATR1 03   NCDE C3E648   RCC-RB 00000000   NM ABENDXIT   USE 01   EPA C5D080   ATR2 2C   XL/MJ 03D560
     03E438       ATR1 B0   NCDE 03E468   RCC-RB 00000000   NM IGG019AB   USE 02   EPA 07F038   ATR2 20   XL/MJ 03E428

                                                                                                     PAGE 0002
     03E848       ATR1 03   NCDE 03E8E8   ROC-RB 00000000   NM IMDPRXED   USE 01   EPA 061848   ATR2 20   XL/MJ 03E970
     03E8E8       ATR1 C3   NCDE C3EAA0   ROC-RB 00000000   NM IMDPROOT   USE 01   EPA 05D1C0   ATR2 20   XL/MJ 03E990
     03E338       ATR1 BC   NCDE C3E368   ROC-RB 00000000   NM IGG019CC   USE 02   EPA 07E178   ATR2 20   XL/MJ 03E328
     03E308       ATR1 B0   NCDE C3E338   ROC-RB 00000000   NM IGG019CH   USE 02   EPA 07E0F8   ATR2 20   XL/MJ 03E2F8
     03DFA8       ATR1 BC   NCDE 03DFD8   ROC-RB 00000000   NM IGG019AC   USE 01   EPA 07CB10   ATR2 2C   XL/MJ 03DF98
     03E3D8       ATR1 E0   NCDE C3E408   ROC-RB 00000000   NM IGG019AQ   USE 02   EPA 07E6B0   ATR2 2C   XL/MJ 03E3C8
     03E2D8       ATR1 B0   NCDE C3E3C8   ROC-RB 00000000   NM IGG019CI   USE 02   EPA 07CCD0   ATR2 2C   XL/MJ 03E2C8
     03E408       ATR1 B0   NCDE 03E438   ROC-RB 00000000   NM IGG019AK   USE 02   EPA 07E938   ATR2 20   XL/MJ 03E3F8
     03E398       ATR1 B0   NCDE C3E3C8   ROC-RB 00000000   NM IGG019AR   USE 02   EPA 07E838   ATR2 2C   XL/MJ 03E388

XL                                            LN          ADR          LN          ADR          LN          ADR

     03EBF0   SZ 0000001C   NO 00000001       8C0042C8    0005D538
     03D158   SZ 00000010   NO 00000001       800007A8    0006A858
     03F358   SZ 0000001C   NO 00000001       80000270    0007E440
     03E288   SZ 00000010   NO 00000001       80000220    0007CBB0
     03E258   SZ 0000001C   NO 00000001       800001C0    0007D9F0
     03E228   SZ 00000010   NO 00000001       80000120    0007D8D0
     03D560   SZ 00000010   NO 00000001       800C0140    0005D080
     03E428   SZ 00000010   NO 00000001       800000A8    0007F038
     03E970   SZ 0000001C   NO 00000001       80003788    00061848
                 0000001C   NO 00                 378     0005D1C0
```

Figure APNDX-6. Sample ABEND Dump Showing Fields Needed for Debugging User Exit Routine ABENDXIT (Part 1 of 3)

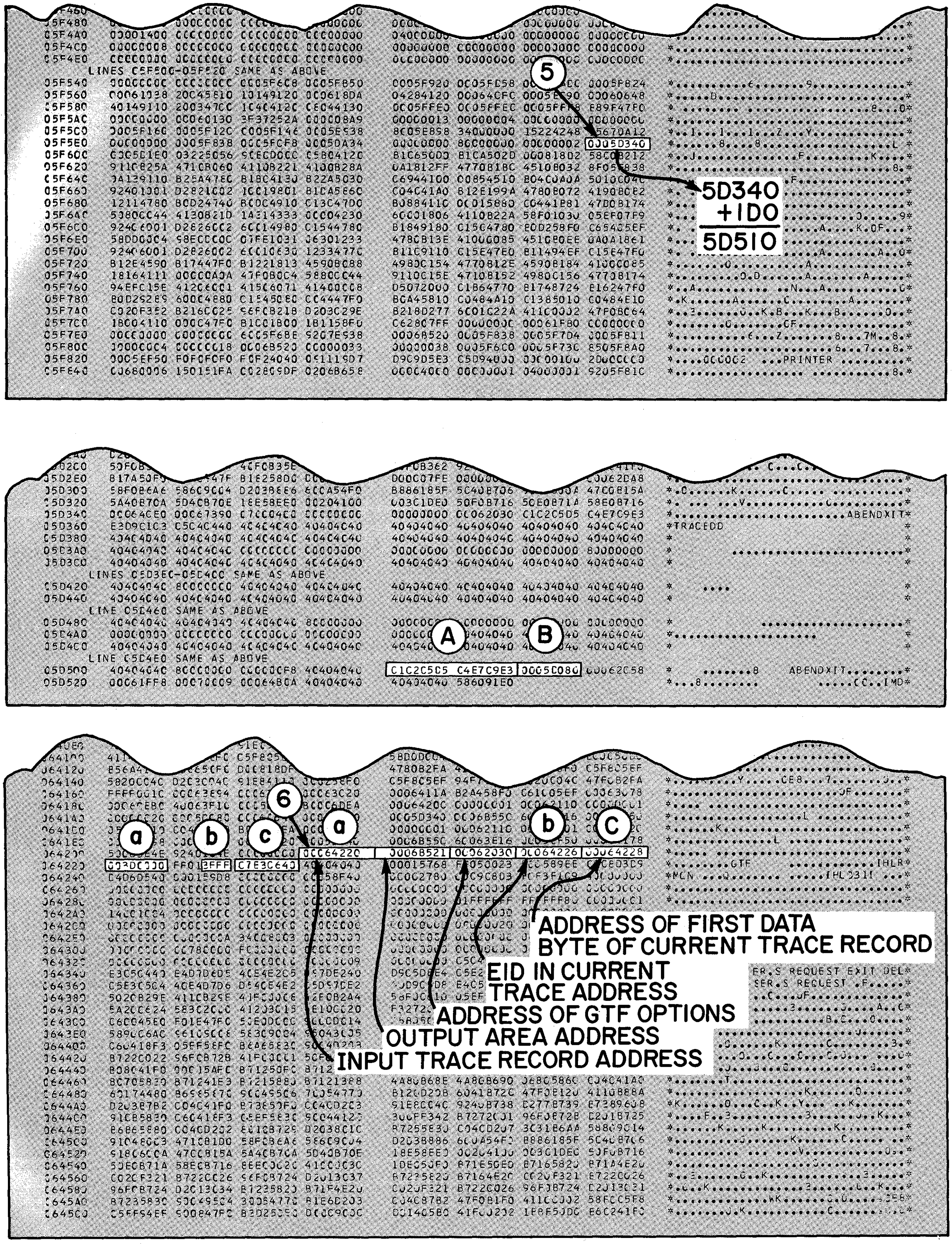


```
                                                                                                        PAGE 0130
07D900    48C050C8 9001D040 4111C0C0 CA3712FF     47805010 05EF9801 C0404111 00000A37    *....... ................. ......*
07D920    91C04005 478C5114 91C82034 47805114     45E050EA 58740C0C 4860203E 91042030    *.. ............................*
07D940    47105C7E 4863000E 91404005 47805C86     18764177 0C011817 18C60A67 91202024    *........  ......................*
07D960    478C5114 1A67S140 2C504780 50CA4580     50BE4177 00044580 50BE1A7F 19764780    *....... ........................*
07D980    51     F 70CC4780 511447FC 5CA6C707     3C083C08 F223300D 70004FF3 000840F3    *...........0..P.....2......3.. 3*
07D9A0    00     3 7C0C3C08 C7F845EC 50EA1888     43802051 1A7847F0 50A69104 203C4780    *..K......8...........0........*
07D9C0    00     1 2052203E C7FE18EE 43802042     1A834880 50084888 00064883 000E4080    *..K.......................... .*
07D9E0    2052C7FE 58E8D014 C7FEC7CC 07000700                                            *  .Y..........Y................*

LOAD MODULE     ABENDXIT

05D080    90ECD00C 05C05GD0 C0DE4190 C0DA5C9D     00091809 9101C124 4710C0C8 5821000C    *..................R..A....H....*
05D0A0    5831C010 9101C125 4710C096 41400000     5940C126 4770C088 4510C072 0805D180    *......A...... ... A...........J.*
05D0C0    0005E1AC 00330000 E2D7C5C3 C9C6E840     F860C3C8 C1D9C1C3 E3C5D940 C1C6C2C5    *..J.....SPECIFY 8.CHARACTER JOBN*
05D0E0    C1D4C540 D6C640C1 C2C5D5C4 C9D5C740     D7D9D6C7 D9C1D400 0A234110 C1264100    *AME OF ABENDING PROGRAM.....A...*
05D100    00C10A01 4140C12A D6C740C0 C132D507     3000C12A 4770C0D0 9601C125 D5012C00    *..... A.O. .A.N...A........A.N...*
05D120    C1224770 C0B6950D 300F4770 C0B6D5C7     3C00C12A 4770C0B6 9601C124 41F000C8    *A.............N...A........A...0..*
05D140    58D0C004 58E0C00C 980CD014 07FE41F0     001847F0 C0BA41F0 000C47F0 C0BA0000    *...............0...0...G....0....*
05D160    00000000 00064178 00000000 00000000     00000000 00000000 00000000 00000000    *................................*
05D180    00000000 00000000 00000000 00000000     00000000 00000000 00000000 00000000    *................................*
05D1A0    00000000 00000000 3FFF0000 00000000     40404040 40404040 40404040 40404040    *...............                *

LOAD MODULE     IGG019AB

07F020                                                                47F0F010 00000000    *.........................00.....*
07F040    D202104D 104907FE 9CE8D014 1821188F     98352044 48602052 1A561945 4720806C    *K........Y......................*
07F060    18774370 204258F0 2C5CC5EF 58F02034     05EF9550 3004478C 802E5857 30004155    *........0.....G.................*
            6108 20344780      20504710     80629845 204847F0 806C     01A45    *..........................0......*
            D50                              00 90352044                          *....N
```



```
SAVE AREA TRACE

xMDPRDMP WAS ENTERED VIA LINK           AT EP xMDPRDMP-21.00

SA   06B760   WD1 00000000    HSA 00000000    LSA 0005D970    RET 000243EA    EPA 0105D538    R0  FD000000C
              R1  0006B7F0    R2  00000000    R3  5C03EA30    R4  0003ACC0    R5  0003AE68    R6  0003C944
              R7  0003CC80    R8  0003EAC8    R9  0003AFE8    R10 0003EA30    R11 00000000    R12 4C07EC5A

xMDPRDMP WAS ENTERED VIA CALL

SA   05D970   WD1 C000C000    HSA 0006B760    LSA 0005E598    RET 6005C900    EPA 0005CA98    R0  B00C6DEA
              R1  0005F0B5    R2  0000C050    R3  5C03EA30    R4  C006B7FB    R5  00000004    R6  0006B7FC
              R7  C003CC80    R8  0003EAC8    R9  C003AFE8    R10 C003EA30    R11 4005C552    R12 C005EF5C

SA   05E598   WD1 C000C000    HSA C0C5D970    LSA 00061FB0    RET 4005CD1C    EPA 00061848    R0  B00C6DEA
              R1  C0061848    R2  C000000C    R3  0005E78B    R4  0005F0B8    R5  00000004    R6  0006B7FC
              R7  C003CC80    R8  0C03EAC8    R9  8005DCD4    R10 0003EA30    R11 6005CA9E    R12 C005EF5C

SA   061FB0   WD1 18691B11    HSA CC05E598    LSA 00063078    RET 50061F74    EPA 00062D70    R0  B00C6DEA
              R1  C0062D70    R2  0C0C0001    R3  00062110    R4  00000001    R5  00000020    R6  00000001
              R7  C0000033    R8  CC06B520    R9  0005D340    R10 C006B55C    R11 4006184E    R12 0005EF5C

SA   063078   WD1 C004CA23    HSA CC061FB0    LSA C0064178    RET 60062DCE    EPA 00063E10    R0  B00C6DEA

                                                                                                    PAGE C005
              R1  G0063E10    R2  CCCCCC01    R3  G0062110    R4  00000001    R5  00000020    R6  G0CC0001
              R7  CCCC0033    R8  0C06B520    R9  0005D340    R10 0006B55C    R   00062D76    R12 0005EF5C

SA   064178   WD1 C61C05EF    HSA C0063078    LSA G0060EB0    RET 40063F10    EPA 0005D080    R0  B00C6DEA
              R1  0006420C    R2  CC0CC001    R3  00062110    R4  00000001    R5  00000020    R6  C005DC8C
              R7  0006C648    R8  00063C20    R9  0005D340    R10 0006B55C    R11 60063E16    R12 0C05EF5C

SA   060EB0   WD1 BCA84410    HSA 00064178    LSA G0060FA0    RET 90060AC2    EPA FFFFFFFF    R0  0005DC8C
              R1  G006B1C8    R2  CC060F6C    R3  0C05D080    R4  C0000001    R5  00000020    R6  0006385E
              R7  C0060648    R8  0C06B158    R9  0006B158    R10 0006B55C    R11 50060656    R12 0005EF5C

SA   060FA0   WD1 4110C185    HSA 0C060EB0    LSA 4770B4C6    RET 861283F6    EPA 47F0B472    R0  5810C008
              R1  4111C010    R2  18315E20    R3  C6801222    R4  47D0B41E    R5  4590B48A    R6  18C3451C
              R7  B424CA0A    R8  5010C680    R9  4B30C13E    R10 40310000    R11 D7011002    R12 10029680

INTERRUPT AT 05CC92

PROCEEDING BACK VIA REG 13

SA   064178   WD1 C61CC5EF    HSA 0C063078    LSA 00060EB0    RET 40063F10    EPA 0005C080    R0  BC0C6DEA
              R1  C006420C    R2  CCCC0001    R3  C0062110    R4  00000001    R5  00000020    R6  0005DC8C
              R7  C0C6C648    R8  0C063C20    R9  0005D340    R10 C006B55C    R11 60063E16    R12 0005EF5C

   63078      WD1 00040A23    HSA CC061FB0    LSA 00064178    RET 60062DCE        63E          B00C6DEA
              R1  00          2 CCC00001                      R4  0000                        C000C001
              R7                CCC6B520                      R10 0                            05EF5C
```

Figure APNDX-6. Sample ABEND Dump Showing Fields Needed for Debugging User Exit Routine ABENDXIT (Part 2 of 3)



```
05F480    0000CCCC CCCC      CC000000    0000                0000000 0000           ...............................*
05F4A0    00001400 CCCC0000 CCCC0000 CC000000    04000000      000 00000000 0000CCCC  *................................*
05F4C0    00000008 CCCC0000 CCCC0000 CC000C00    00000000 C0000000 0000000C C000C000  *................................*
05F4E0    CCCC0000 0CCCC0C0 CCCCGCC0 CCCC0600    0CC00000 00000000 00    0C C0000000  *................................*
      LINES C5F500-05F520 SAME AS ABOVE
05F540    0000CC0C CCCCCCCC CC05F6C8 0C05FB50    0005F920 0C05FC58 00   CC 0005F824  *...........6........9...........*
05F560    00061038 20C45810 10149120 0C0618DA    04284120 00064CFC 0005E 90 00060648  *.....D..........................*
05F580    40149110 200347CC 1C4C412C CE044130    0C05FFE0 0C05FFEC 0C05FF 8 E89F47F0  *...........................8..0*
05F5AC    00CC0000 CC060130 3F37252A C000C8A9    00000013 00000004 00C00000 00000000  *................................*
05F5C0    0005F160 0005F12C C005F146 0C05E538    8C05E858 34000000 15224248 5670A12   *..1...1...1..Z...Y..............*
05F5E0    00C00000 0005F838 0CC5FCF8 0C05DA34    00000000 8C000000 00C00002 0005D340  *......8....8..................L *
05F600    C0C5C1E0 03225056 9CECD0CC C5B0412C    B1C650D0 B1CA502D 000818D2 58C0B212  *..J..............F.........K....*
05F620    911CB25A 471CB060 4110B221 4100B28A    0A1812FF 4770B1BC 4510B032 8F05 838  *..............................8.*
C5F64C    0A139110 B25A478C B1BC4130 B22A5030    C6944100 00854510 B04C0A0A 5010C04C  *...              F..............*
05F660    92401001 D2821C02 1CC19801 B1CA5E6C    C04C41A0 B12E199A 4780B072 4190BCB2  *...                .............*
05F680    12114780 BCD24740 BCDC4910 C13C47DC    B0884110 0C015880 C0441E81 47D0B174  *...                .............*
05F6A0    5080CC44 4130B21D 1A314333 0CC04230    60C01806 4110B22A 58F01030 05EF07F9  *...                ......3....9*
05F6C0    924C6001 D2826CC2 6CC14980 C1544780    B1849180 C15C4780 E0D258F0 C65405EF  *...                ..A....K.0F...*
05F6E0    58DD00C4 98ECCC0C 07FE1031 06301233    478CB13E 41000085 4510B0EE 0A0A1861  *...                .............*
05F700    92406001 D2826002 6CC10630 1233477C    B11C9110 C15E47E0 B11494EF C15E47F0  *...                ..A.........A..0*
05F720    B12E4590 B17447F0 B1221813 4590BC88    4980C154 4770B12E 4590B184 4100C085  *........0..............A.........*
05F740    18164111 0CCC0A0A 47F0B0C4 5880CC44    9110C15E 4710B152 4980C156 4770B174  *.........0.D.......A.......A.....*
05F760    94EFC15E 412C6C01 415C6071 41400CC8    05072000 C1B64770 B1748724 E16247F0  *..A.............. ..N...A.........C*
05F780    B0D29289 600C4880 C1545080 CC4447F0    BCA45810 C0484A10 C1385010 C0484E10  *.K.......A........C........A.......*
05F7A0    C020F352 B216C025 96FCB218 D203C29E    B218D277 6C01C22A 411C0002 47F0BC64  *..3.......0..K.B...K...B.......0..*
05F7C0    18C04110 000C47FC B1C01B00 181158F0    C62807FF 0000000C 00061FB0 CC000000  *.........0.........CF.............*
05F7E0    00CC0000 C00CCCGC 6CC5F6BE 9207E538    0006B520 0005F838 0005F7D4 0005F811  *.............6...Z.........8...7M..8.*
05F80C    0000CCC4 CCCCCC18 0C06B520 CC000033    00000038 0005F6C0 0005F73C 8505F8A0  *.......................6...7...8.*
05F820    00C5EF50 F0FCFCF0 FCF24040 09111 9D7    D9C9D5E3 C5D94000 00C00100 20000CC0  *....0CC0C2   ...PRINTER ..........*
C5F840    C0680006 150151FA C028C9DF 0206B658    0C0C40C0 00C00001 04000001 9205F81C  *.......................  .........8.*
```

```
05D2C0    50F0B      4CF0B35E                  F0B362 92                     41F     ..C...C.
05D2E0    817A50F     47F B1E258D0 D0           D00C07FE 000000                00062DA8  *.................................*
05D300    58F0B6A6 586C9C04 D203B8E6 6CCA54F0    B886185F 5C40B706        00A 47C0815A  *.0......K......C.... ...........*
05D320    5A40B70A 5D4CB70E 18E58EE0 00204100    003C1DE0 50F0B716 5CE0B71A 58E0B716  *. ... ...V..............C.........*
05D340    0C064CE0 00C67390 C7CC04C0 CCCC0C0C    00000000 0C062030 C1C2C5D5 C4E7C9E3  *..............................ABENDXIT*
05D360    E3D9C1C3 C5C4C440 4C4C4C40 40404C40    40404040 40404040 40404040 404C4C40  *TRACEDD                          *
05D380    404C4040 404C4040 4C4C404C 404C4C40    40404040 4040404C 40404040 40404040  *                                *
05D3A0    4040404C 404C404C CCCCCCCC C0000000    00C00000 0C000C00 00000000 80000000  *          ......................*
05D3C0    40404040 404C404C 4C4C4040 4C4C4040    40404040 40404040 40404040 40404040  *                                *
      LINES C5D3EC-05C4CC SAME AS ABOVE
C5D420    4040404C 8CCCCCC0 40404040 4040404C    40404040 4C404040 40404040 40404040  *     ....                       *
05D440    40404C40 404C4040 4C404040 404C4040    40404040 40404040 40404040 404C4040  *                                *
      LINE C5D460 SAME AS ABOVE
05D480    404C4040 40404340 4C404C4C 8CCC0000    000C0C    C000000 00    00 00000000  *                ..................*
05D4A0    00000000 0CCCCCC0 CCC000C0 00C000C0    000C0     0404040 4      40 40404040  *......................          *
05D4C0    40404040 4C404040 4C4C404C 4C40404C    404040     0404040 4      40 40404040  *                                *
      LINE C5D4E0 SAME AS ABOVE
05D500    4040404C 8CCC0000 CCCC0CF8 4C40404C    C1C2C5D5 C4E7C9E3 0005C080 00062C58  *     .......8    ABENDXIT........*
05D520    00C61FF8 00C70C09 0C064B0A 40404040    40404040 586091E0                      *...8........          .....CC..IMD*
```

```
064100    411       C5F8C5                         58D0DCC                   CCJC5CCC
064120    E56A41     C65CFC D0C818DF               4780B2FA 4            F0 C5F8C5EF
064140    5820CC4C D2C3C04C 91E84110 3C0258F0    C5F8C5EF 94F           20C04C 47F0B2FA  *.......Y.....CE8...7.........0..*
064160    FFFF0010 CC063E94 CC06    0CC63C20    0006411A B2A458F0 C61C05EF C0063078  *.....................0F........*
064180    0006CEBC 40063F10 CCC5     BCCC6DEA    0006420C C000C001 0C062110 CC000001  *................................*
0641A0    0C0CCC20 0CC5CE80 CC0         0C        0C05D340 CC06B55C 60      16 00    60  *.................L...............*
0641C0    0     10 CC4        B        EA 0C0         00000C01 00062110 0       01 0      2C  *.......K........................*
0641E0    00     58 00C        0          20 C0         0006B55C 6C063E16 J0       F50 00      178  *..............L.................*
064200    50     E4E 9240     E CCC    CC0 0CC64220    0006B521 0C062030 0C064226 00064228  *......... ......................*
064220    003DC000 FF01 3FFF C7E3C640 40 04040    00015768 F 050023   C589EE C  C8D3C9  *........GTF         .........IHLR*
064240    C4D6D540 C0015908 CCCCCCCC C 58F40    CC0C2780 0C9C8D3  CF3F1C9  0000000  *MCN ...Q........ .....IHL0311 ...*
064260    00CCC0CC CCCCCCCC CCCCCCC0 C 000000    0CCC0CC0 C C0000  00  000 00000000  *................................*
064280    00C0000G 0C0CCCC0 CCCCCCCC 0 0C0C00    00000000 01FFFF FF FFFFFF80 C000C0C1  *................................*
0642A0    14CC1C84 0CCCCCCC CCCC0CC0 C C00C00    C0000000 00 0000 0
0642C0    00000000 0CCC0COC CCCC0CCC C 000CC0    C0000000 0000020 0
0642E0    0CCCCCCC C00C0CCA 34CC8CC3 0 C00000    C000000  0  000  0
064300    00CCCCCC CC78CCCG FC000CC0 0 CCC0C0    000C0000 0C  000 0
064320    0CCCCC00 CCCCCCCC 3CCC0CCC  0C0C09    0C0C0C00 C5C4
064340    E3C5C440 E4D7D6D5 4CE4E2C5  97DE240    D9C5D8E4 C5E2                        ER.S REQUEST EXIT DEL*
064360    C5E3C5C4 40E4D7D6 D54CE4E2  5D97DE2     0D9C D8 E4C5                        SER.S REQUEST .F.....*
064380    5020B29E 411CB29E 41FC00C8  2F0B2A4    58F0C010 05EF                        ..C...0F...............*
0643A0    5A2CC624 583C2C00 41203C15  E10CC20    F3272                              .....3........0....A..C*
0643C0    C6C045E0 F01E47FC 50E0DCCC  0CCDC14     58050                             ..........B.C...0.....*
0643E0    5890C6AC 9610C0C8 5E3C9004  6043C05                                    .....................0......*
064400    C60418F3 05EF58FC B6A65E3C  0C40203                                   .0.......K.......C.........3.*
064420    B722C022 96FCB72B 41FC0CC1 50F                                        ...0...C...C...C.0.........0*
064440    B08C41F0 00C15AFC B71250FC B712                                      ..0...0...0...............0....*
064460    8C705830 B71241E3 B7215880 B7121388    4A80B68E 4A80B690 0680586C C04C41A0  *........T.......................*
064480    60174480 B6985870 9CC495C6 70054770    B120D20B 6041B72C 47F0E120 4110B88A  *....................K......0.....*
0644A0    D203B7B2 C04C41F0 B73E50F0 CC4CD2C3    91E8CC4C 9240B738 C277B739 B7389608  *K......0...C..K..Y... ..K........*
0644C0    91CB5830 C6C418F3 C5EF5E3C 9C04412C    300FF342 B7272C01 96F0B72B D2018725  *.....F..3...........3.......0..K...*
0644E0    B6B65880 C04CD202 8010B729 D2038C1C    B7255E30 C04CD207 3C31B6AA 58809C14  *......K......K..........K..........*
064500    91048CC3 471C81D0 58FCB6A6 586C9C04    D203B886 6C0A54F0 B886185F 5C40B706  *..........0......K.......C..... ..*
064520    918C6CCA 47CCB15A 5A4CB7CA 5D40B70E    18E58EE0 00204100 003C1DEC 50F0B716  *.......... ... ...V.............0..*
064540    50ECB71A 58ECB716 8EEC0C2C 41CC0C3C    10EC50F0 B71E50E0 B7165820 B71A4E20  *...................0...............*
064560    C02CF321 B722CC26 96FCB724 D2013C37    B7235820 B7164E2C CC20F321 E722C026  *..3.......C..K................3.....*
064580    96FCB724 D2C13C34 B7235820 B71F4E20    C020F321 B722C026 96F0B724 D2013C31  *..C..K..............3.........C..K...*
0645A0    B7235830 90C49504 30054770 B1E6D203    C04C87B2 47F0B1F0 411C0002 58FCC5F8  *...............WK.......C.0.......3E6*
0645C0    C5FF94EF 900847FC B3D25050 D0CC9C0C    D0140580 41F00202 1BBF50DC E6C241F0  *.........0.K..............C.........*
```

5
5D340
+1D0
5D510
A
B
6
a
b
c
a
b
C
ADDRESS OF FIRST DATA BYTE OF CURRENT TRACE RECORD
EID IN CURRENT TRACE ADDRESS
ADDRESS OF GTF OPTIONS
OUTPUT AREA ADDRESS
INPUT TRACE RECORD ADDRESS

Figure APNDX-6. Sample ABEND Dump Showing Fields Needed for Debugging User Exit Routine ABENDXIT (Part 3 of 3)

Debugging a User Program

Figure APNDX-6 shows a sample ABEND dump of the user exit routine ABENDXIT, shown in Figure APNDX-5. Figure APNDX-6 as shown and the following explanatory notes are valid for VS1 only; a similar figure and notes valid for VS2 would substitute AMD for HMD wherever that combination appears in a field name. Certain important fields are highlighted in the figure and marked with numbers; the numbers refer to the explanations below:

1. PSW for the abnormally terminating program. The address in the second half of the PSW is an address in the abnormally terminated program. To find the entry point and name of the program, compare this address to the entry point addresses in the contents directory entry list. The abnormally terminating program is the one whose entry point address is closest to and greater than the address in the PSW.

   NOTE: If the address in the PSW does not immediately indicate the entry point address of the failing program, you can locate the beginning address of the abnormally terminating program by tracing PRDMP's save area chain. See point 4, below.

2. Part of a contents directory entry (CDE). This shows the name of the abnormally terminating program, ABENDXIT, its entry point, X'05D080', and the pointer to the appropriate entry in the extent list.

3. An extent list entry. This shows the beginning address (not necessarily the entry point) of the abnormally terminating program. Subtract this address from the address in the PSW to find the address of the instruction following the instruction that failed.

   For example, in this case:

   address in PSW - beginning address = offset (hex)

   5D092 - 5D080 = 12

   The failing instruction in ABENDXIT can be found at offset X'12' in the program. (See part 2 of Figure APNDX-6, number 3.)

4. The first save area in the save area trace table (system save area) is chained to the following HMDPRDMP module save areas:

   HMDPRCTL - HMDPRDMP control routine

   HMDPRMSC - HMDPRDMP scan routine

   HMDPRFRM - EDIT control routine

   HMDPRFLT - EDUT trace record selection routine

   HMDPREXT (or HMDPRAPP) - EDIT user program selection routine.

5. The user program's registers are stored in HMDPREXT's or HMDPRAPP's save area. Add the contents of register 12 to X'6AC' to get the address of a fullword that points to an EDIT communication table. At offset X'1D0' into this table are the following:

APP

A. The 8-oyte EBCDIC name of the current user program (the failing program).

B. The entry point address of the current user program (the failing program).

Ihese fields are shown in part 3 of Figure APNDX-6.

6. Register 1 in HMDPREXT's or HMDPRAPP's save area points to the parameter list that EDIT passes to the user program. (See Figure APNDX-1.)

Job and Control Statement Examples

The following examples show how to test a user program.

Example 1: Link Editing a User Exit Routine into a Library

This example shows how to make a user exit routine available to PRDMP by link-editing it into a system library.

```
//LKUSRPGM     JOB        MSGLEVEL=(1,1)
//             EXEC       PGM=IEWL,PARM='XREF,LET,LIST,NCAL'
//SYSPRINT     DD         SYSOUT=A
//SYSLMOD      DD         DSNAME=SYS1.LINKLIB,DISP=OLD
//SYSLIN       DD         *
    object deck
         NAME       EXITNAME
/*
```

In this example:

EXEC Statement

invokes the linkage editor and requests maximum diagnostic listings.

SYSPRINT DD Statement

defines the message data set.

SYSLMOD DD Statement

defines the output data set, in this case the linkage library, SYS1.LINKLIB. The output data set can also be a permanent library to be invoked later by a JOBLIB or STEPLIB DD statement; in that case the SYSLMOD DD statement should be coded as follows:

```
//SYSLMOD DD    DSNAME=MYLIB,UNIT=2314,VOL=SER=231400,
//              DISP=(NEW,KEEP),SPACE=(1024,(20,2,1))
```

SYSLIN DD Statement

defines the input data set, in this case, the object deck for the user program.

NAME Control Statement

specifies the member name, and thus the program name, to be assigned to the user program. In this case, the member name is EXITNAME; to invoke this program in a later execution of PRDMP, you would have to speciy EXIT=EXITNAME on the EDIT control statement.

Example 2: Testing a User Exit Routine

This example shows how to link edit a user exit routine into a library for testing.

```
//TSEXTRTN    JOB       MSGLEVEL=(1,1)
//STEP1            EXEC      PGM=IEWL,PARM='XREF,LET,LIST,NCAL'
//SYSPRINT    DD        SYSOUT=A
//SYSLMOD     DD        DSNAME=MYLIB,UNIT=2314,VOL=SER=231400,
//       DISP=(NEW,KEEP),SPACE=(1024,(20,2,1))
//SYSLIB      DD        *
    object deck
         NAME      MYEXIT
/*
//STEP2       EXEC      PGM=HMDPRDMP,PARM='ER=1'
//STEPLIB     DD        DSNAME=MYLIB,UNIT=2314,VOL=SER=231400,
//       DISP=OLD
//SYSPRINT    DD        SYSOUT=A
//PRINTER    DD         SYSOUT=A
//TRACEDD     DD        DSNAME=TRACE2,UNIT=2400,VOL=SER=TRC2TP,
//       LABEL=(,NL),DISP=OLD
//SYSIN       DD        *
         EDIT      DDNAME=TRACEDD,SYS,EXIT=MYEXIT
/*
```

This example consists of two steps. In the first step:

EXEC Statement

invokes the linkage editor and requests diagnostic information.

SYSPRINT DD Statement

defines the message data set.

SYSLMOD DD Statement

defines the output data set, in this case a permanent job or step library named MYLIB.

SYSLIN DD Statement

defines the input data set, in this case an object deck containing the user program.

NAME Control Statement

specifies a member name (program name) to be assigned to the user program. Specify this program name on the EDIT control statement (EXIT=MYEXIT) when you need the exit routine for a particular PRDMP execution.

In the second step:

EXEC Statement

invokes PRDMP and specifies that, if an error occurs in the exit routine, EDIT should print the record associated with the error and delete the exit routine. (See the discussion of the EXEC statement in the section "Job Control Language Statements" earlier in this chapter.)

STEPLIB DD Statement

defines the data set that contains the exit routine, which, in this case, is MYLIB, a data set defined in STEP1 by the SYSLMOD DD statement.

SYSPRINT DD Statement

defines the message data set.

PRINTER DD Statement

defines the data set to which PRDMP output will be directed.

TRACEDD DD Statement

defines the data set containing trace records to be processed by the exit routine.

SYSIN DD Statement

defines the data set that contains the PRDMP control statement. The data set follows immediately.

EDIT Control Statement

invokes the EDIT function of PRDMP, specifies that the trace data exists as an external trace data set, and supplies the name of the exit routine. Note that this name is the same as the membername specified in the NAME control statement in STEP1.

APP

Example 3: Testing a User Format Appendage

This example shows how to add a user format appendage to a temporary data set for testing.

```
//TSTFMT        JOB         MSGLEVEL=(1,1)
//STEP1         EXEC        PGM=IEWL,PARM='XREF,LET,LIST,NCAL'
//SYSPRINT      DD          SYSOUT=A
//SYSLMOD       DD          DSNAME=&TEMPLIB,UNIT=SSYSDA,
//         SPACE=(1024,(20,2,1)),DISP=(NEW,PASS)
//SYSLIN        DD          *
    object deck
         NAME      HMDUSR01
/*
//STEP2         EXEC        PGM=HMDPRDMP,PARM='ER=3'
//STEPLIB       DD          DSNAME=&TEMPLIB,DISP=OLD
//SYSPRINT      DD          SYSOUT=A
//PRINTER       DD          SYSOUT=A
//TRACEDD       DD          DSNAME=TRACE,UNIT=2400,VOL=SER=TRCTPE,
//         LABEL=(,NL),DISP=OLD
//SYSIN         DD          *
         EDIT      DDNAME=TRACEDD,USR=ALL
/*
```

This example consists of two steps. In the first step:

EXEC Statement

invokes the linkage editor.

SYSPRINT DD Statement

defines the message data set.

SYSLMOD DD Statement

defines a temporary data set that contains the format appendage.

SYSLIN DD Statement

defines the input data set, in this case the object deck containing the format appendage.

NAME Control Statement

specifies a member name (program name) for the format appendage. Note that the name shown in this example conforms to the convention for naming format appendages; that is, it is formed from the prefix HMDUSR concatenated with the format identifier (FID) to be specified in the GTRACE macro when user records are created. (In VS2, the name must be formed from the prefix AMDUSR concatenated with the format identifier.)

In the second step:

EXEC Statement

invokes PRDMP and specifies that ABEND processing should not be suppressed if a program check occurs in the format appendage. (See the discussion of the EXEC statement in the section "Job Control Language Statements" earlier in this chapter.)

STEPLIB DD Statement

defines the data set where the format appendage resides.

SYSPRINT DD Statement

defines the message data set.

PRINTER DD Statement

defines the data set to which the format appendage will direct its output.

TRACEDD DD Statement

defines the trace data set containing the records that the format appendage will process. In this case, the trace data set is on tape.

SYSIN DD Statement

defines the data set containing PRDMP control statements. The data set follows immediately.

EDIT Control Statement

invokes the EDIT function of PRDMP, specifies that the trace data exists as an external trace data set, and specifies that EDIT is to process all user-created records.

Appendix B: SADMP Wait State Codes →

Explains wait state codes issued during execution of SADMP stand-alone dump program.

APP

Appendix B - SADMP Wait State Codes

The stand-alone dump program (SADMP) uses the following wait state codes to communicate with the operator during execution of the dump. These codes appear in the console lights as the last four bytes of the current PSW.

0000 Explanation: Normal termination of the dump program. The dump program has successfully executed; all tapes (if any) have been taped marked, rewound, and unloaded.

Response: None.

0004 Explanation: An I/O error occurred attempting to write an informational message on the console.

Response: Correct the console error before re-executing SADMP.

0008 Explanation: An I/O error occurred attempting to write an error message on the console.

Response: Correct the console error before re-executing SADMP.

000C Explanation: A short tape reel exists. Under normal circumstances, message HMD006I or AMD006I would be issued to signal this problem, but in this case the console is unavailable.

Response: Refer to message HMD006I for appropriate action.

0010 Explanation: An unknown error condition occurred during SADMP processing.

Response: To diagnose the problem in SADMP processing, restart SADMP.

1000 Explanation: An unknown external interrupt condition occurred.

Response: To diagnose the problem in SADMP processing, restart SADMP.

2000 Explanation: An unknown SVC interrupt condition occurred.

Response: To diagnose the problem in SADMP processing, restart SADMP.

3000 Explanation: An unknown program check interrupt condition occurred.

Response: To diagnose the problem in SADMP processing, restart SADMP.

4000 Explanation: An unknown machine check interrupt occurred.

Response: Execute the SEREP program to diagnose the machine check condition.

Index

Indexes to OS/VS publications are consolidated in the *OS/VS Master Index,* GC28-0602, and the *OS/VS Master Index of Logic,* GY28-0603. For additional information about any subject listed below, refer to other publications listed for the same subject in the Master Index.

INDE

GC28-0633-1

OS/VS Service Aids Printed in U.S.A. GC28-0633-1

IBM

International Business Machines Corporation
Data Processing Division
1133 Westchester Avenue, White Plains, New York 10604
(U.S.A. only)

IBM World Trade Corporation
821 United Nations Plaza, New York, New York 10017
(International)

OS/VS Service Aids

GC28-0633-1

READER'S COMMENT FORM

Your views about this publication may help improve its usefulness; this form will be sent to the author's department for appropriate action. Using this form to request system assistance or additional publications will delay response, however. *For more direct handling of such requests, please contact your IBM representative or the IBM Branch Office serving your locality.*

Possible topics for comment are:

Clarity Accuracy Completeness Organization Index Figures Examples Legibility

What is your occupation? ____
Number of latest Technical Newsletter (if any) concerning this publication: ______________
Please indicate in the space below if you wish a reply.

Thank you for your cooperation. No postage stamp necessary if mailed in the U.S.A. Elsewhere, an IBM office or representative will be happy to forward your comments.

Cut or Fold Along Line

GC28-0633-1

Your comments, please . . .

This manual is part of a library that serves as a reference source for system analysts, programmers, and operators of IBM systems. Your comments on the other side of this form will be carefully reviewed by the persons responsible for writing and publishing this material. All comments and suggestions become the property of IBM.

Cut or Fold Along Line

Fold Fold

First Class
Permit 81
Poughkeepsie
New York

Business Reply Mail
No postage stamp necessary if mailed in the U.S.A.

Postage will be paid by:

International Business Machines Corporation
Department D58, Building 706-2
PO Box 390
Poughkeepsie, New York 12602

Fold Fold

OS/VS Service Aids Printed in U.S.A. GC28-0633-1

IBM®

International Business Machines Corporation
Data Processing Division
1133 Westchester Avenue, White Plains, New York 10604
(U.S.A. only)

IBM World Trade Corporation
821 United Nations Plaza, New York, New York 10017
(International)

IBM Technical Newsletter

This Newsletter No. GN28-2540
Date December 15, 1972

Base Publication No. GC28-0633-1
File No. S360-37

Previous Newsletters None

OS/VS Service Aids

This Technical Newsletter, a part of release 2 of OS/VS1, provides replacement pages for the subject publication. These replacement pages remain in effect for subsequent releases unless specifically altered. Pages to be inserted and/or removed are:

Cover,2
6.1
17,18
47,48
127,128
135,136
175
183-190

A change to the text or to an illustration is indicated by a vertical line to the left of the change.

Summary of Amendments

Chapter 1 contains a single change which indicates a restriction on starting GTF using the START command.

Chapter 5 was changed to include a description of the new USR parameter, DMA1, which has been created for EDIT. It was also changed to include a FORMAT control statement restriction which occurs when input is an SVC dump.

Chapter 6 was changed to add a specification that the PCHF DD statement must describe Stage 1 output from the generation of the system to be updated. Two figures were deleted because LINKS and ASMS catalogued procedures are the same for VS1 and VS2.

Note: Please file this cover letter at the back of the manual to provide a record of changes.

IBM Corporation, Publications Development, Department D58, Building 706-2,
PO Box 390, Poughkeepsie, New York 12602