IMS

Diagnosis Guide and Reference

Version 7

LY37-3738-05

IMS

Diagnosis Guide and Reference

Version 7

LY37-3738-05

Note

Before using this information and the product it supports, be sure to read the general information under “Notices” on page 565.

Sixth Edition (April 2005) (Softcopy only)

This edition replaces or makes obsolete the previous edition, LY37-3738-04. This edition is available in softcopy format only. The technical changes for this version are summarized under “Summary of Changes” on page xvii.

Contents

Figures

Tables

About This Book

This book helps system programmers and other diagnostic technicians diagnose internal problems in IMS. It also provides instructions for reporting these problems to IBM®.

| This information is available in PDF and BookManager® formats. To get the most current versions of the PDF and BookManager formats, go to the IMS Library page at www.ibm.com/software/data/ims/library.html.

You must enter a customer license number in order to view the book on the Web.

Summary of Contents

This book has three sections and several appendixes. Basic concepts presented in each section are outlined below.

Part 1, "Identifying System Problems," on page 1, guides you in systematically setting up your system so that you can properly collect data about problems that might occur. You then use a set of keywords to search an IBM software support database to determine if the failure has been previously reported and corrected. If it has not, you can use the keyword string when communicating with IBM support representatives.

Part 2, "Data Areas and Record Formats," on page 51, contains diagrams that show the interrelationships of control blocks for some major IMS functions. This section also includes the layout of various types of records useful in diagnosis.

Part 3, "Diagnostic Aids," on page 107, describes service aids and other techniques used to detect, trace, and document failures in IMS functions. You will probably want to use this section when your keyword search has been unsuccessful and you need to gather additional information to resolve the problem.

Appendix A, "IMS Keyword Dictionary," on page 445, contains information that you might need while following the procedures in Chapter 4, "Selecting the Keywords," on page 19 or while analyzing program failures.

All information is valid for a Database Control (DBCTL) environment except where specifically noted. CICS® information is intended only for CICS local-DL/I users.

For a list of all non-IMS publications cited in this book, see the "Bibliography" on page 569.

Prerequisite Knowledge

You will be most successful in using this book if you have a basic understanding of:

- IMS concepts and externals
- How to access an IBM software support database
- Dump analysis
- MVS™ diagnostic practices
- Telecommunications
- System Network Architecture (SNA)

How to Send Your Comments

Your feedback is important in helping us provide the most accurate and highest quality information. If you have any comments about this book or any other IMS documentation, you can do one of the following:

- Go to the IMS Library page at www.ibm.com/software/data/ims/library.html and click the Library Feedback link, where you can enter and submit comments.
- Send your comments by e-mail to imspubs@us.ibm.com. Be sure to include the name of the book, the part number of the book, the version of IMS, and, if applicable, the specific location of the text you are commenting on (for example, a page number or table number).

Summary of Changes

| Changes to The Current Edition of This Book for IMS Version 7

| This edition, which is available in softcopy format only, includes technical and editorial changes.

Changes to This Book for IMS Version 7

This book contains new technical information for Version 7, as well as editorial changes.

This book contains new information for the following topics:
- DBRC Serviceability Enhancements
- HALDB (High Availability Large Database)
- IMS ESAF Trace Enhancement
- MADS I/O Timing
- TM Serviceability
- RECON Performance Enhancement

You will also find new information to explain how to set up your installation to collect data about system problems that might occur.

Library Changes for IMS Version 7

The major change to the IMS Version 7 library is that it is available not only in hardcopy and in softcopy on BookManager, but also in softcopy Portable Document Format (PDF).

Changes are indicated by a vertical bar (|) to the left of the changed text.

The library includes a new book: *IMS Version 7 IMS Java™ Guide and Reference* (IJUG). As a new book, the IJUG is available only in PDF and BookManager formats.

Other changes include changes to these following books:
- *IMS Version 7 Common Queue Server and Base Primitive Environment Guide and Reference*

  The book formerly titled *IMS/ESA® Common Queue Server Guide and Reference* in the Version 6 library is called *IMS Version 7 Common Queue Server and Base Primitive Environment Guide and Reference*.

  The *IMS Version 7 Common Queue Server and Base Primitive Environment Guide and Reference* is divided into two parts: ″Part 1: Common Queue Server,″ and ″Part 2: Base Primitive Environment.″

  The *IMS Version 7 Common Queue Server and Base Primitive Environment Guide and Reference* is now an unlicensed book.
- *IMS Version 7 Command Reference*

  The book formerly titled *IMS/ESA Operator's Reference* in the Version 6 library is called *IMS Version 7 Command Reference*.
- *IMS Version 7 Utilities Reference: Database and Transaction Manager*

  The books formerly titled *IMS/ESA Utilities Reference: Database Manager* and *IMS/ESA Utilities Reference: Transaction Manager* in the Version 6 library have been combined into one book called *IMS Version 7 Utilities Reference: Database and Transaction Manager*.
- *IMS Version 7 Application Programming: Database Manager* and *IMS Version 7 Customization Guide*

  The chapter titled ″IMS Adapter for REXX Exit Routine″ has been moved from the *IMS Version 7 Application Programming: Database Manager* to the *IMS Version 7 Customization Guide*.
- *IMS Version 7 Sample Operating Procedures*

For IMS Version 7, this book is available only in BookManager and PDF formats.

- The book formerly titled *IMS Version 7: IMS Java User's Guide* is now titled *IMS Version 7 IMS Java Guide and Reference*.

Part 1. Identifying System Problems

Chapter 1. Setting Up Your System

IMS™ can process large amounts of work efficiently; it is a very complex product. However, IMS can experience problems that need to be diagnosed and corrected. The following are examples of problems that you might encounter while running IMS:

- An abnormal end (known as an *abend*) occurs in processing.
- A job hangs in the system and does not process.
- A process repetitively loops through a series of instructions.
- Processing slows down.

For these types of problems, IMS displays symptoms that can help you with your diagnosis, but in order to obtain that information, you'll need to be sure your system is set up correctly. To ensure that you have effectively gathered the correct data to diagnose a problem, begin to set up your system, using the recommendations in the following sections.

Specify the IMS Control Region EXEC Parameter Value: FMTO=D

Specify the IMS control region `EXEC` parameter value, `FMTO=D`, to produce an SDUMP for terminating and non-terminating errors, specifically, DB2® and dynamic allocation abends. A SYSMDUMP, SYSABEND, or SYSUDUMP will be produced only if SDUMP fails.

Specify a SYSMDUMP Statement in JCL for CTL, DLI, and DBRC Regions

Place a `SYSMDUMP DD` statement in the JCL of the IMS control, DLI, and DBRC regions. In the event that SDUMP processing fails, IMS will use the SYSMDUMP you specified.

If a SYSMDUMP needs to be taken, specify the following dump options in the IEADMR00 member of SYS1.PARMLIB to ensure that adequate areas of MVS storage are dumped to diagnose the problem:

```
SDATA=(CSA,GRSQ,LSQA,RGN,SQA,SUM,SWA,TRT)
```

Specify a SYSUDUMP Statement in JCL for IMS Dependent Regions

Place a `SYSUDUMP DD` statement in the JCL of IMS dependent regions. The following dump options should be specified in SYS1.PARMLIB member IEADMP00 to ensure that adequate areas of MVS storage are dumped:

```
SDATA=(CB,ERR,SUM) PDATA=(JPA,LPA,PSW,REGS,SA,SPLS)
```

Set Up the Internal Trace Environment

IMS dispatcher, scheduler, DLI, and lock traces are the internal IMS traces that are most useful for general problem diagnosis. Set up these traces by one of the following methods:

- Specify these options in IMS.PROCLIB member DFSVSMxx:
  ```
  DISP=ON, SCHD=ON, DL/I=ON, LOCK=ON
  ```
- Enter this IMS command:
  ```
  /TRA SET ON TABLE nnnn
  ```

  where *nnnn* is either: `DISP`, `SCHD`, `DLI`, or `LOCK`. Only one trace table option can be entered per `/TRA` command.

Recommendation: Use the IMS LATCH trace for all test systems. Your system can run with the LATCH trace active in production without measurable performance degradation. Specify `LATC=ON` for the LATCH trace in the IMS PROCLIB member DFSVSMxx.

Install the IMS Dump Formatter

Install the IMS interactive dump formatter, if your installation is at IMS Version 4 or higher.

The IMS dump formatter can be used to format either the complete IMS dump, or only those sections needed to analyze the problem. The interactive dump formatter is IPCS-based and uses an ISPF dialogue to allow you to view a specific control block.

See “Interactive Dump Formatter” on page 151 for more information about using the interactive dump formatter.

Set Up the External Trace Environment

Request external tracing by starting traces with the `OUT` option, or if the MTO starts a trace with the LOG option.

You can start certain traces at initialization time with these methods:

- For online systems, specify the appropriate trace keywords on the `OPTIONS` statement in IMS.PROCLIB member DFSVSMxx.
- For a batch environment, specify the appropriate trace keywords on the `DFSVSAMP DD` statement.

You can also turn tracing off or on by using the `/TRACE` command.

Control the Volume of Traces

Control the volume of the traces using the trace volume. It can be set to *High, Medium*, or *Low*, where *High* generates the largest volume of trace entries, and *Low* generates the smallest volume of trace entries.

For details about the `/TRACE` command parameters, refer to *IMS Version 7 Command Reference*. For details about the `OPTIONS` statement in the DFSVSAMP or DFSVSMxx data set, see *IMS Version 7 Installation Volume 2: System Definition and Tailoring*.

Recommendation: Ensure that your IMS environment is running with the following traces on at all times:

- Dispatcher
- DL/I
- Lock
- Scheduler

None of these traces causes a noticeable performance impact, and each of these can be extremely helpful to you in diagnosing a variety of problems that might occur in your environment.

Activate Fast Path Traces

In a Database Control (DBCTL) environment, you can trace DL/I and Fast Path activity. You turn on the DL/I trace in the same way as in a DB/DC environment. The trace records for coordinator controller (CCTL) threads contain the recovery token that can help you correlate CCTL tasks with DBCTL threads.

Activate Fast Path tracing in one of the following ways:

- The DBCTL operator can enter the `/TRACE SET ON TABLE FAST` command. This is the same way you activate the trace in a DB/DC environment. In both DBCTL and DB/DC environments you must also specify the FPTRACE DD statement in the IMSFP procedure, which is described in *IMS Version 7 Installation Volume 2: System Definition and Tailoring*.

- The CCTL decides which transactions to trace and directs DBCTL to activate the trace for those transactions. After the transaction completes, the trace output file is closed and sent to the SYSOUT data set, class A. However, when certain transactions fail in Fast Path processing and the trace is not already active, the Database Resource Adapter (DRA) recommends to the CCTL that Fast Path tracing be activated. The failures for which tracing is recommended are based on the list that IMS uses for Fast Path Transaction Retry. The CCTL can then direct DBCTL (through the DRA) to activate Fast Path tracing the next time that transaction is scheduled.

Write Trace Tables Externally

You can write the incore trace tables to an external device, tape data set, or to the online log data set (OLDS).

When the IMS MTO starts IMS trace table traces with the `LOG` option, the following selection order determines where the external traces are written.

DASD JCL
DD statements are checked to verify that DFSTRA01 or DFSTRA02 are present. If either or both are present, the JCL specified DASD external trace data sets are used if possible.

DASD MDA
An attempt is made to dynamically allocate and open DFSTRA01 and DFSTRA02 using dynamic allocation members. If either or both dynamic allocations succeed, the DASD external trace data sets are used if possible.

TAPE MDA
An attempt is made to dynamically allocate and open member DFSTRA0T. If the dynamic allocation succeeds, the external trace tapes are used if possible.

IMS log data set
The IMS log data set is used for external trace. Because of the performance effects of logging trace data to the online log data set, the operator is asked to approve tracing to the online log data set when external trace data sets cannot be used.

To print the X'67FA' records, use the File Select and Formatting Print utility (DFSERA10), and specify exit DFSERA60 to format the trace entries.

DFSTRA01 and DFSTRA02 are the external trace data sets used by the IMS online systems. The trace data sets are used when the trace table `OUT` parameter is used in the DFSVSMXX options statement, or when the `/TRACE START ON TABLE` *nnn* option log command is used. The trace data sets are used in a wrap-around fashion. For example, when DFSTRA01 fills, DFSTRA02 is used; when DFSTRA02 fills, DFSTRA01 is used.

Recommendation: You must remember to offload the trace data set before it is reused. Use the IEBGENER utility to offload the data set.

Create Output Data Sets with Correct Attributes

Create the DFSTRA01 and DFSTRA02 trace data sets with the following attributes, in order for you to use them to hold your trace data:

DSORG
SEQUENTIAL

RECFM
VB

LRECL
4004

BLKSIZE
A formula of: (LRECL*N)+4. The block size must be a multiple of the LRECL (4004), with the additional 4 bytes for the block descriptor word. IBM recommends a BLKSIZE of 20024, which is 5 logical records in length (4004 bytes, multiplied by 5), plus the block descriptor word (4 bytes). The BLKSIZE of 20024 is recommended for current DASD because it is equal to one-half track.

Recommendation: These data sets must be allocated as a single extent, meaning contiguous tracks. Do not specify secondary allocation.

In order to use a tape to hold the external trace data set, you must use the DFSTRA0T data set. DFSTRA0T must be dynamically allocated with the following attributes:

DSORG
SEQUENTIAL

RECFM
VB

LRECL
4004

BLKSIZE
A formula of: (LRECL*N)+4. The block size must be a multiple of the LRECL (4004), with the additional 4 bytes for the block descriptor word.

In order to dynamically create these data sets, use the following JCL example.

```
/STEP    EXEC IMSDALOC
//SYSIN     DD  *
DFSMDA TYPE=INITIAL
DFSMDA TYPE=TRACE,DDNAME=DFSTRA01,DSNAME=IMS41.DFSTRA01
DFSMDA TYPE=TRACE,DDNAME=DFSTRA02,DSNAME=IMS41.DFSTRA02
DFSMDA TYPE=TRACE,DDNAME=DFSTRAT2,DSNAME=IMS41.DFSTRA0T
DFSMDA TYPE=FINAL
END
```

Set MVS System Trace Table Size

The MVS system trace is useful for many types of MVS problems. At times, it is the only means of reconstructing a problem. The larger you can specify the size of the trace table, the better the chance of diagnosing some of the more intricate problems encountered while running IMS. Specify the MVS command `TRACE ST,999K` in the MVS COMMNDxx SYS1.PARMLIB member so that the trace table size is in effect during IPL. If you do not specify a trace table size, at MVS versions lower than 5.2.0, the default size is 16K; at MVS version 5.2.0 and above, the default size is 64K. If your installation has a limited number of real page frames, remember that the system trace table is page fixed. If you specify the dump option `SDATA=(TRT)`, the dump size will increase.

Set MVS Master Trace Table Size

The MVS master trace table contains a buffer of messages from the MVS master console. These messages will be saved in the SDUMP data set and can be viewed using IPCS to aid in problem diagnosis. Specify the MVS command `TRACE MT,100K` in the MVS SYS1.PARMLIB member SCHEDxx so that the trace table size is in effect during IPL. If you do not specify a trace table size, at MVS versions lower than 5.2.0, the default size is 24K; at MVS version 5.2.0 and above, the default size is 64K.

Ensure the Size of SYS1.DUMPxx Data Sets are Correct

The SYS1.DUMPxx data sets should be large enough to contain up to five IMS regions in one dump data set. IMS attempts to dump the CTL, DLI, DBRC, IRLM, and possibly one dependent region, into the SYS1.DUMP data set. For some large installations, the required size can be over 500 cylinders of 3390 DASD. The mixture of IMS GEN specifications, MVS GEN specifications, and IMS processing will produce different storage utilizations, and therefore, different sizes of IMS dumps.

Follow these recommendations to find a safe SYS1.DUMPxx data set size:

- Allocate a SYS1.DUMP data set using the following MVS DUMP command to obtain an IMS dump for estimation purposes:

```
DUMP COMM=(dump title)
R id JOBNAME=(j1,j2,j3,j4,j5),
SDATA=(CSA,PSA,RGN,SQA,SUM,TRT),END
```

This will produce a very large dump. In the previous example,

j1 is the IMS CTL or DBCTL region jobname

j2 is the IMS DLI region jobname

j3 is the Large IMS dependent region jobname

j4 is the IRLM region jobname (If IRLM DB Locking used)

j5 is the DBRC region jobname

SYS1.DUMPxx dynamic allocation is allowed at version MVS 5.1.0 and higher.

- Take a dump of these regions as close as possible to a high utilization period.

After the dump completes, its size can be referenced as a *minimum* size and increased, with an acceptable buffer allowance, for peak utilization periods.

Set Up CQS Tracing

The PROCLIB member you specify using the `BPECFG=` parameter in the CQS (common queue server) execution parameters defines configuration parameters to BPE. The `TRCLEV=` parameter is used in the BPE configuration parameter to specify the trace level for a trace table, and optionally the number of pages of storage allocated for the trace table. You can specify one `TRCLEV=` parameter for each trace table type that BPE and CQS supports. These trace tables are internal incore tables only. Trace records are not written to any external data sets.

```
TRCLEV=(type,level,IMS component,[ PAGES=num_pages])
```

The value for *type* can be one of the following:

AWE - component = BPE
The asynchronous work element (AWE) services trace table traces AWE server creation and deletion and AWE processing requests.

CBS - component = BPE
The control block services trace table traces requests for control block storage.

DISP - component = BPE
The dispatcher trace table traces BPE dispatcher activity.

LATC - component = BPE
The latch trace table traces BPE latch management serialization.

STG - component = BPE
The storage service trace table traces storage service requests.

SSRV - component = BPE
The system services trace table traces general BPE system service calls.

USRX - component = BPE
The user exit routine trace table traces activity related to exit routines (for example, loads, calls, or abends).

CQS - component = CQS
The CQS trace table traces general activity that is not related to a specific structure.

INTF - component = CQS
The interface trace table traces activity in the interface between a CQS and its client.

STR - component = CQS
: The structure trace table traces activity related to a structure. CQS defines one STR trace table for each structure pair defined to CQS.

The `LEVEL` parameter controls how much tracing is done in the specified trace table. Each trace entry that is made in CQS or BPE has a level associated with the entry. Each trace table has a level setting, which is controlled by the value for `LEVEL` that you specify on the `TRCLEV` statement for the table. A trace entry is written only if the trace entry's level is less than or equal to the table's level setting. For example, if the trace entry level is *MEDIUM*, the trace entry would be added to the trace table only if the table's level is *MEDIUM* or *HIGH*. So, the level you specify controls the volume (or number) of trace entries that are written to a given table. The value for *level* can be one of the following:

NONE
: **Recommendation:** Do not specify `NONE` as the level parameter because no tracing, not even tracing for error conditions, will be done for the specified table.

ERROR
: Only trace entries for error conditions are made. `ERROR` is the default.

LOW
: Low-volume tracing (key component events) is the recommended trace level setting for normal CQS operation.

MEDIUM
: Medium-volume tracing (most component events).

HIGH
: High-volume tracing (all component events).

The `PAGES=` parameter can be added to the `TRCLEV` statement to specify the number of 4 KB pages that are to be allocated for the trace table type. If you do not specify this parameter, the default number of pages defined internally by BPE or CQS is obtained for the trace table.

Specify the following trace entries within the `BPECFG=nnnnnnnn` PROCLIB member:

```
DEFINITIONS FOR BPE SYSTEM TRACES
    TRCLEV=(AWE,MEDIUM,BPE)            /* AWE SERVER TRACE           */
    TRCLEV=(CBS,MEDIUM,BPE)            /* CONTROL BLK SRVCS TRACE    */
    TRCLEV=(DISP,MEDIUM,BPE)           /* DISP WITH 12 PAGES (48K)   */
    TRCLEV=(LATC,MEDIUM,BPE)           /* LATCH TRACE                */
    TRCLEV=(SSRV,MEDIUM,BPE)           /* GEN SYS SERVICES TRACE     */
    TRCLEV=(STG,MEDIUM,BPE)            /* STORAGE TRACE              */
    TRCLEV=(USRX,MEDIUM,BPE)           /* USER EXIT TRACE            */

--DEFINITIONS FOR CQS TRACES

    TRCLEV=(CQS,MEDIUM,CQS)            /* CQS GENERAL TRACE          */
    TRCLEV=(STR,MEDIUM,CQS)            /* CQS STRUCTURE TRACE        */
    TRCLEV=(INTF,MEDIUM,CQS)           /* CQS INTERFACE TRACE        */
```

Chapter 2. Collecting Data about Problems

When you pass a problem to the IBM Support Center, the information you collect when the problem occurs is very important to help diagnose what went wrong at your installation. Having this information available when you call IBM can save you time because you might not need to recreate the problem. When you decide you need to diagnose a system problem, follow these steps:

1. When the problem occurs, collect the symptom data and determine what type of problem it is.
2. Once you determine the type of problem, use the procedures recommended to diagnose the problem. This will help you determine if the problem is an IMS problem or a user problem.
3. If it is an IMS or system problem, build a search argument from the data that you collect as a result of following the procedure for that problem. For example, the data you gather from a control region wait can be helpful in building a search argument.
4. Perform the search. You might have to refine your search with more data from the problem.
5. If you cannot find a fix, report the problem to IBM.

Collecting Data about General Problems

Depending on the complexity of the problem, you may need to gather the following information:

- SYSLOG

  Save the SYSLOG from time of IMS start up. The SYSLOG is useful when the dumped MTRACE buffer is not large enough to find necessary error messages.
- LOGREC data set

  Save the LOGREC data set from IMS start up time. MVS failures are logged internally.
- IMS master console log

  Save the master console log from IMS start up time. The master console log provides a different message set than the SYSLOG.
- IMS log data sets

  Save the IMS online data sets active at the time of the error.
  - IMS system log data sets (SLDS)

    Save the SLDS from IMS start up time.

  The IMS log data sets enable you to track IMS transaction and database activity; the tracking is critical for proper diagnosis of many IMS problems.
- JES job log of jobs related to failure

  Save the JES job log from IMS start up time. The JES job log provides JCL start up parameters and isolated system messages.
- Any dumps produced

  Multiple SYS1.DUMP data sets are sometimes produced. Examine SYSMDUMPs if there is a primary SYS1.DUMP failure. Also, examine SYSUDUMPs for IMS dependent regions or ABENDU0002 SYSUDUMPs for wait or hang problems.
- MVS log data sets produced

  Save the current MVS log data sets for the failing CQS job stream. The MVS log data sets provide information for structure rebuild and checkpoint related problems.

Collecting Data about Specific Problems

Occasionally, there are problems in specific environments, or for certain problem types, that require special handling. Some types of problems of this nature include:

- Control Region Wait/Hang
- Control or DLI Region Loop
- Dependent Region Wait/Loops
- DB2 ESS Interface Problems
- DBRC Related Problems
- DBCTL Related Problems
- IMS/VTAM Related DC Problems
- APPC Related DC Problems
- CQS Related Problems

Diagnosing a Control Region Wait or Hang

When an IMS control region waits or hangs, IMS can take on various appearances from being completely frozen, to losing a partial function. The most critical piece of information will be the MVS SVC dump.

Recommendation: Do not use the MVS `MODIFY dump` (`F jobname,DUMP`) command as a source of IMS diagnostic information. This command adds unnecessary complexity to the dump while processing the modify abends.

Obtain an MVS SVC dump with this series of commands:

```
DUMP COMM=(dump title)
R id JOBNAME=(j1,j2,j3,j4,j5,j6),
SDATA=(CSA,PSA,RGN,SQA,SUM,TRT),END
```

In the previous example,

j1 is the IMS CTL or DBCTL region jobname

j2 is the IMS DLI region jobname

j3 is the suspicious IMS dependent region jobname, if any

j4 is the suspicious CCTL (CICS) region name, if any

j5 is the IRLM region jobname (if IRLM DB locking is used)

j6 is the DBRC region jobname

Most likely, a dump of the IMS CTL, DLI, and suspicious dependent region or CCTL is sufficient to solve wait or hang problems. Occasionally, the DBRC and IRLM (if used for DB locking) regions can become a factor. So, DBRC and IRLM should also be included.

If IMS is not completely stopped (for example, IMS commands can still be entered, BMPs are still processing, and some transactions still process), taking a second MVS SVC dump will help differentiate normal IMS processing from the problem.

Diagnosing a Control or DLI Region Loop

If IMS appears to be looping, follow these steps:

1. If IMS can accept commands, use the following IMS command to set up the internal trace environment:

   ```
   /TRA SET ON TABLE nnnn
   ```

where *nnnn=* can be `DISP`, `SCHD`, `DLI`, `LOCK` or `LATCH`. Each must be entered separately.

2. Set the MVS System Trace table size to 999K and turn on branch tracing with this command:
```
TRACE ST,999K,BR=ON
```
3. Obtain two MVS SVC dumps of the CTL, DLI, suspicious dependent region, or CCTL, DBRC, and IRLM regions. Taking a second MVS SVC dump will help differentiate normal IMS processing from the problem. Obtain an MVS SVC dump with this series of commands:
```
DUMP COMM=(dump title)
R id JOBNAME=(j1,j2,j3,j4,j5,j6),
SDATA=(CSA,PSA,RGN,SQA,SUM,TRT),END
```

In the previous example,

j1 is the IMS CTL or DBCTL region jobname

j2 is the IMS DLI region jobname

j3 is the suspicious IMS dependent region jobname, if any

j4 is the suspicious CCTL (CICS) region name, if any

j5 is the IRLM region jobname (if IRLM DB locking is used)

j6 is the DBRC region jobname

Diagnosing an IMS Dependent Region Wait or Loop

If the dependent region appears to be looping, follow these steps:

1. If IMS can accept commands, use the following IMS command to set up the internal trace environment:
```
/TRA SET ON TABLE nnnn
```

where *nnnn=* can be `DISP`, `SCHD`, `DLI`, `LOCK`, or `LATCH`. Each must be entered separately.

2. Set the MVS System Trace table size to 999K and turn on branch tracing with this command:
```
TRACE ST,999K,BR=ON
```
3. If the problem is a wait, obtain two MVS SVC dumps of the CTL, DLI, suspicious dependent region, or CCTL, DBRC, and IRLM regions. If the problem is a loop, obtain two MVS SVC dumps of the CTL, DLI, suspicious dependent region, or CCTL, DBRC, and IRLM regions. Obtaining a second MVS SVC dump will help differentiate normal IMS processing from the problem. Obtain an MVS SVC dump with this series of commands:
```
DUMP COMM=(dump title)
R id JOBNAME=(j1,j2,j3,j4,j5),
SDATA=(CSA,PSA,RGN,SQA,SUM,TRT),END
```

In the previous example,

j1 is the IMS CTL or DBCTL region jobname

j2 is the IMS DLI region jobname

j3 is the suspicious IMS dependent region jobname, if any

j4 is the IRLM region jobname (if IRLM DB locking is used)

j5 is the DBRC region jobname

Diagnosing a DB2 ESS Interface Problem

IMS DB2 ESS interface problems are fairly rare, and therefore, can be difficult to diagnose. The IMS ESS trace is costly (it impacts performance) so it is unwise to activate it on a regular basis. Turn on the trace when you notice a problem or if you need to recreate a problem. If you are diagnosing a problem involving the DB2 ESS interface, follow these steps:

1. Use this IMS command to turn on the IMS ESS trace and to direct its output to the external trace data set:

```
/TRA SET ON TABLE SUBS OPTION LOG
```

   The SUBS trace is more complete if a successful ESS call is performed before the failure, and activates tracing at a lower level.
2. Obtain dumps of the IMS CTL and involved dependent regions, before and after the failure, with this series of commands:

```
DUMP COMM=(dump title)
R id JOBNAME=(j1,j2,j3,j4,j5),
SDATA=(CSA,PSA,RGN,SQA,SUM,TRT),END
```

3. Obtain an MVS SVC dump of DB2 MSTR and DBM1 regions with this series of commands:

```
DUMP COMM=(dump title)
R id JOBNAME=(dbtmstr,dbwdbm1),
SDATA=(CSA,PSA,RGN,SQA,SUM,TRT),END
```

4. Save the IMS online log data set that was active during the failure because IMS TYPE5501, 08, 07, 56 and other log records can be critical to diagnosis. The IMS TYPE5501 records are updated by DB2 modules and their contents are explained in *DB2 for OS/390 Version 5 Diagnosis Guide and Reference*. The internal buffer for these records is stored at the location described by the CDE entry named WAL in the IMS regions.
5. If the IMS monitor is started, use the following command to monitor the IMS data set:

```
/TRACE SET ON MONITOR ALL
```

Diagnosing a DBRC Related Problem

DBRC related problems can manifest themselves in a variety of symptoms, including waits and loops. If you need to recreate the problem, copies of the RECON listing, before and after the problem occurred, are most useful. To diagnose a DBRC related problem you will need the following information:

- Obtain a listing of the DBRC RECONs for the time frame that is as close as possible to failure time. Use the Recover Control Utility (DSPURX00) `LIST.RECON` command to obtain the listing.
- Obtain a subsystem listing if you cannot obtain a RECON listing because of its size. Use the Recover Control Utility (DSPURX00) `LIST.SUBSYS ALL` command to obtain a subsystem listing.

Diagnosing a DBCTL Related Problem

DBCTL related problems can be centered in either the CCTL region or in one of the IMS regions (CTL, DLI, DBRC, or IRLM). So, it is important to obtain dumps relating to all these regions.

1. Use the following IMS commands to aid in problem diagnosis because they include region ID numbers and recovery tokens in their various display output:

```
/DISPLAY ACTIVE
```

   and

```
/DISPLAY CCTL
```

   The information from these commands will greatly increase the accuracy and speed required to diagnose the problem. The `DISPLAY ACTIVE` command provides the reasons for waits and region numbers. The `DISPLAY CCTL` command provides recovery tokens and region numbers. Save the IMS console output.
2. Set the ERM portion of the CICS trace to level 1-2. Save this output.
3. Set the FILE CONTROL portion of the CICS trace to level 1-2. Save this output.
4. Obtain the necessary MVS SVC DUMP of the IMS regions with this series of commands:

```
DUMP COMM=(dump title)
R id JOBNAME=(j1,j2,j3,j4,j5,j6),
SDATA=(CSA,PSA,RGN,SQA,SUM,TRT),END
```

In the previous example,

j1 is the IMS CTL or DBCTL region jobname

j2 is the IMS DLI region jobname

j3 is the suspicious IMS dependent region jobname, if any

j4 is the suspicious CCTL (CICS) region name, if any

j5 is the IRLM region jobname (if IRLM DB locking is used)

j6 is the DBRC region jobname

5. Save the IMS online log data set that was active during the failure.

Diagnosing a DC Related Problem

IMS DC related problems are mainly associated with VTAM®. VTAM dumps are often required to help diagnose problems, but are infrequently obtained by operations personnel. IMS NODE traces, VTAM BUFFER traces, and VTAM INTERNAL traces are often required in conjunction with the IMS region dumps and VTAM dumps to solve DC problems. It is important to obtain this information while you are experiencing the problem.

The IMS log tapes contain much of the transaction data that flows through IMS. This transaction data includes the following IMS records:

- TYPE01
- TYPE03 (MSG queue entries)
- TYPE11 through TYPE16 (SPAs, DIALs, SIGN)

Start the recreate attempt after issuing an IMS `/SWITCH OLDS` command to have the related data placed on a new OLDS.

1. Issue the IMS `DISPLAY NODE x` command and save the IMS console output. Here is the syntax:

   ```
    /DIS NODE nodename
   ```

2. Turn on the IMS NODE trace with the following command. Data will be captured in the IMS TYPE6701 log record. Save the IMS online log data set for use with the IMS utility programs DFSERA10 and DFSERA30.

   ```
   /TRA SET ON NODE nodename
   ```

3. Consider turning on the VTAM Buffer Trace and VTAM Internal Trace to complement the IMS NODE trace with this series of commands:

   ```
   F NET,TRACE,TYPE=BUF,ID=nodename
   F NET,TRACE,TYPE=VTAM,MODE=EXT,
    OPT=(API,PIU,MSG)
   ```

   GTF must be active with the `USR` option to capture these trace entries.

4. Obtain an MVS dump of the IMS regions with this series of commands:

   ```
   DUMP COMM=(dump title)
   R id JOBNAME=(j1,j2,j3,j4,j5,j6),
   SDATA=(CSA,PSA,RGN,SQA,SUM,TRT),END
   ```

   In the previous example,

   j1 is the IMS CTL or DBCTL region jobname

   j2 is the IMS DLI region jobname

   j3 is the suspicious IMS dependent region jobname, if any

   j4 is the suspicious CCTL (CICS) region name, if any

   j5 is the IRLM region jobname (if IRLM DB locking is used) *j6* is the DBRC region jobname

5. Obtain a dump of the VTAM address space with this series of commands:

```
DUMP COMM=(dump title)
R id JOBNAME=(vtam jobname),
SDATA=(CSA,PSA,RGN,SQA,SUM,TRT),END
```

6. Save the IMS log tapes created during the error period.

Diagnosing an APPC Related DC Problem

APPC problems originating from IMS dependent regions that make calls explicitly, rely heavily on the dependent region dumps. Follow these steps to diagnose an APPC-related DC problem.

1. Turn on the IMS LUMI trace, for the external trace data set, using the following IMS /TRACE commands:

```
/TRACE SET ON TABLE LUMI OPTION LOG
```

The LOG option can be set up to cause the output to be sent to the external trace data set with this /TRACE command:

```
/TRACE SET ON LUNAME XXXXXXX INPUT
```

```
TRACE SET ON LUNAME XXXXXXX OUTPUT
```

where *XXXXXXX* is the partner LU

2. Turn on the VTAM Buffer Trace and VTAM internal trace to complement the IMS LUMI trace with these commands:

```
F NET,TRACE,TYPE=BUF,ID=luname
```

```
F NET,TRACE,TYPE=VTAM,MODE=EXT,
  OPT=(API,PIU,MSG)F
```

GTF must be active with the USR option specified to capture these trace entries.

3. Turn on the program trace to trace TPPCB DL/I calls, so that the APPC component trace can send its trace buffers to a SYS1.DUMP data set when it stops. Turn on the program trace with this command:

```
/TRACE SET ON PROGRAM pppppppp
```

where *pppppppp* is the program name of the application.

4. Turn on the MVS APPC component trace with this command:

```
TRACE CT,ON,200K,COMP=SYSAPPC
```

5. Reply to the MVS outstanding reply with the following response:

```
nn,OPTIONS=(GLOBAL),END
```

6. When the problem has been recreated stop the CTRACE with this command:

```
TRACE CT,OFF,COMP SYSAPPC
```

You can use the following IPCS commands to format the trace:

- For 1-line entries:

```
CTRACE COMP SYSAPPC SHORT
```

- Summary of each entry:

```
CTRACE COMP SYSAPPC FULL
```

7. Obtain an MVS SVC dump of the IMS regions with this series of commands:

```
DUMP COMM=(dump title)
R id JOBNAME=(j1,j2,j3,j4,j5,j6),
SDATA=(CSA,PSA,RGN,SQA,SUM,TRT),END
```

In the previous example,

j1 is the IMS CTL or DBCTL region jobname *j2* is the IMS DLI region jobname

j3 is the suspicious IMS dependent region jobname, if any

j4 is the suspicious CCTL (CICS) region name, if any

j5 is the IRLM region jobname (if IRLM DB locking is used)

j6 is the DBRC region jobname

8. Obtain a dump of the APPC, APPC Scheduler, and VTAM address spaces with this series of commands:

```
DUMP COMM=(dump title)
R id JOBNAME=(j1,j2,j3),SDATA=(CSA,PSA,RGN,SQA,SUM,TRT),END
```

In the previous example,

j1 is the APPC jobname

j2 is the APPC scheduler jobname

j3 is the VTAM jobname

9. Start the recreate attempt after issuing an IMS `/SWITCH OLDS` command to have related data placed in a new OLDS. Save the IMS log tapes that are created during the error period. IMS log records are not as useful for explicit APPC applications as they are for implicit APPC applications because very little information is logged about explicit APPC applications.

Diagnosing a CQS Related Problem

CQS produces SDUMPs for internal errors. The CQS dumps can be found in the SYS1.DUMP data sets. CQS can also produce LOGREC data set entries for errors.

If you encounter a CQS WAIT problem, obtain one dump using the command in step 1. If you encounter a CQS loop problem, obtain two dumps.

1. Obtain an MVS SVC DUMP of the CQS, CTL, DLI, suspicious dependent region, DBRC, and IRLM regions with the following series of commands:

```
DUMP COMM=(dump title)
R id JOBNAME=(j1,j2,j3,j4,j5,j6),
SDATA=(CSA,PSA,RGN,SQA,SUM,TRT),END
```

In this command:

j1 is the IMS CTL or DBCTL region jobname

j2 is the IMS DLI region jobname

j3 is the suspicious IMS dependent region jobname, if any

j4 is the suspicious CCTL (CICS) region name, if any

j5 is the IRLM region jobname (if IRLM DB locking is used)

j6 is the DBRC region jobname

2. Save the IMS log data sets that are created during the error period.
3. Save the current MVS log data sets that are created. The current MVS log data sets for the CQS log stream can be copied using the IEBGENER utility. There are no archived MVS log data sets (unlike the IMS logger that does have log archive capability through SLDS).

If an isolated event type within CQS encounters an error, the IBM Support Center might request additional trace level settings for the various trace types. See “Set Up CQS Tracing” on page 7 for information about trace descriptions. If a structure rebuild or structure checkpoint related problem occurs, you will also need to dump the CQS address spaces for any CQS associated with the given structure, and save the associated SRDS (structure recovery data set) for the CQS structure checkpoints and CQS system

| checkpoints.

Chapter 3. Searching Problem Reporting Databases

After you have obtained information about the problem you are diagnosing, you can use that information to create search arguments to search problem reporting databases for known problems that describe an aspect of a program failure.

You use keyword strings to search an IBM software support database, such as the Software Support Facility (SSF). SSF is an online database containing information about the resolution of reported problems called Authorized Program Analysis Reports (APARs). If the search is successful, you will find a similar problem description, and usually a correction, or fix. If the failure is one that is known, you will use the keywords to describe the failure when contacting the IBM Support Center for assistance, or when documenting a possible APAR.

Some optional search tools might require keywords in a structured database (SDB) format. Follow the procedures described here to build your keyword string. Then, if necessary, translate these keywords into the SDB format by using Appendix A, "IMS Keyword Dictionary," on page 445. Each search argument example in the procedures shows a free-form example followed by an SDB example.

Developing Search Arguments

A keyword describes one aspect of a program failure. A set of keywords, called a *keyword string*, describes a specific problem in detail. Because you use a keyword string to search a database, a keyword string is also called a *search argument*.

The keywords you use to search for problems in IMS are:

- The component identification

  This is the first keyword in the string. A search of the database with this keyword alone detects all reported problems for that version of IMS.
- The type of failure

  The second keyword specifies the type of failure that occurred. Its values can be:

  ABENDxxx

  ABENDUxxxx

  DOC

  PERFM

  MSGx

  INCORROUT

  WAIT/LOOP
- Symptom keywords

  These can follow the keywords above and supply additional details about the failure. You select these keywords as you proceed through the type-of-failure keyword procedure that applies to your problem.

  Add symptom keywords to the search argument gradually so that you receive all data matches or *hits*, which are problem descriptions that might match your problem. If you receive too many problem descriptions to examine, you can add **AND** or **OR** operators to additional keywords in various combinations to the keyword string to reduce the number of hits.
- Dependency keywords

  These are program or device dependent keywords that define the specific environment that the problem occurred in. When added to your set of keywords, they can help reduce the number of problem descriptions you need to examine. See Appendix E, "Dependency Keywords," on page 545 for a list.

Creating a Search Argument

To build the keyword string and search the IBM software support database for a problem similar to the one you are experiencing, follow these steps:

1. Begin with “Component Identification Keyword Procedure” on page 19 to determine the failing IMS component.
2. Follow the sequential steps in one of the “Type-of-Failure Keyword” procedures until you build a keyword string.
3. Then go to “Searching the Database” on page 47, to learn how to search the IBM software support database with your completed string.
4. If your search is unsuccessful, go to “Preparing an APAR” on page 49.

You might also want to refer to these appendixes:

- Appendix A, “IMS Keyword Dictionary,” on page 445 provides guidance on translating free-form keywords into structured database (SDB) format.
- Appendix C, “Module-to-Function-to-Subfunction List,” on page 465 lists alphabetically all IMS modules and the function and subfunction in which they appear.
- Appendix D, “Save-Area-ID-to-Module Cross-Reference Table,” on page 521 lists all IMS save area IDs and identifies which module contains each of them.
- Appendix E, “Dependency Keywords,” on page 545 lists words used as search techniques to narrow search arguments.
- Appendix F, “Module-to-Waiting-Resource List,” on page 547 lists the waiting conditions or resources that can be associated with an IMS task.

Chapter 4. Selecting the Keywords

This chapter shows you how to select the proper keywords to search the IBM Software Support database for a problem similar to the one you are experiencing. The keywords you select depend on the component that is experiencing the problem and the type of failure that occurs.

Component Identification Keyword Procedure

Use a component identification number with at least one other keyword to search the IBM software support database.

The component identification numbers for IMS appear in Table 1.

Table 1. IMS Component Identification Numbers

5655B0100	IMS Services Database Manager Transaction Manager Extended Terminal Option (ETO) Recovery-level Tracking Database-level Tracking
569516401	Internal Resource Lock Manager (IRLM) 2.1

To determine the type of IMS program failure that is occurring, go to “Type-of-Failure Keyword.”

Some of the procedures on the following pages contain offsets in control blocks. Be aware that maintenance might change the offsets in these control blocks. For a current version of the layout of the control blocks for your system, assemble DFSADSCT from IMS.ADFSSRC.

Type-of-Failure Keyword

From the following seven types, select the one that best describes the program failure. Then go to the procedure for that type of failure.

ABENDxxx
Use this procedure when the system terminates abnormally with a system abend completion code. An abend produces an SVCDUMP, SYSABEND dump, or SYSUDUMP.

ABENDUxxxx
Use this procedure when an IMS application program terminates abnormally with an abend completion code. An abend produces a SYSABEND dump, SVCDUMP, or SYSUDUMP.

DOC
Use this procedure if a deficiency is found in documentation through omission or inaccuracy.

PERFM
Use this procedure if performance is other than what is expected.

MSGx
Use this procedure if a problem involves an IMS message.

INCORROUT
Use this procedure when output is missing or incorrect.

WAIT/LOOP
Use the WAIT/LOOP procedure when there is no response from IMS functions.

ABENDxxx Procedure

Use this procedure when the system terminates abnormally with a system abend completion code. For user abends, go to “ABENDUxxxx Procedure” on page 21.

After you have developed a search argument, refer to Chapter 5, “Procedures and Techniques,” on page 47 for detailed information on how to use the search argument.

Keyword: ABENDxxx

Compare the completion code and PSW address in both the MVS-formatted section of the dump and the IMS-formatted section of the dump. If they do not match, use only the data from the IMS-formatted section because the system dump data might be produced if an abend occurs during ABEND processing.

Replace the xxx part of the ABENDxxx keyword with the abend code from either the termination message or the abend dump.

Keyword: RCxx

This keyword applies only if the abend has an associated return code as described in *MVS/ESA System Codes*.

Replace the xx part of the RCxx keyword with the return code.

Keyword: Module Name

You can determine the name of the module that received the abend in one of the following ways:

- Check both the dump title and message DFS629I, which might contain the name of the abending module.
- Check the summary section, called “Diagnostic Area”, in the offline formatted dump.
- Find the PSW address at the time of abend. Locate this address in the storage section of the dump, and scan backward through the eye-catchers until you find a module identifier. To relate a module identifier to a module name, see Appendix D, “Save-Area-ID-to-Module Cross-Reference Table,” on page 521.

Module-Specific Keywords

Failing Instruction, Register: You can use these module-specific keywords to further narrow the field of hits.

- **Failing Instruction:** The PSW address at the time of abend usually points to the next instruction to be executed. If ABEND0C4 or ABEND0C5 occurs and the INTC (interrupt code) field on the PSW AT ENTRY TO ABEND line contains X'0011' (segment exception) or X'0010' (page translation exception), the PSW points directly to the instruction that failed.

  Use *System/390® Reference Summary* to determine the instruction mnemonic.
- **Register in Error:** Examine the code near the failure to determine the register that is invalid or in error, if possible.

  Example: If the failing instruction is BALR (05EF), look at registers 14 (E) and 15 (F). If register 15(F) contains zeros, the program cannot branch to that location. Therefore, register 15 is in error.

  In performing system-abend analysis, another module might have passed the register in error. You might be able to determine this by looking at the registers on entry to the failing module. If the incorrect value is in one of the registers, that value might have been passed.

Search Argument Example

If, for example, ABEND0C4 occurred in IMS module DFSFXC30 on a BALR (05EF) instruction because register 15 (F) contained zeros, the search argument to use is:

```
5655B0100 ABEND0C4 DFSFXC30
```

For a structured database search, use this search argument:

```
PIDS/5655B0100 AB/S00C4 RIDS/DFSFXC30
```

With this search argument, you might receive numerous hits, which would most likely include the APAR describing your problem. You can add keywords from “Module-Specific Keywords” on page 20 to narrow the field of hits received. It is a good idea to use the **OR** operator with these additional keywords at first.

The additional keywords for this example are:

```
BALR | R15  ZEROS
```

For a structured database search, use this search argument:

```
OPCS/BALR  |  REGS/GR15 VALU/H00000000
```

ABENDUxxxx Procedure

Use this procedure when an IMS user abnormal termination occurs. For user abends, you must gather more information before calling the IBM support center.

A message usually precedes a user abend. First look up the message and then the abend code in *IMS Version 7 Messages and Codes, Volume 1*. If *IMS Version 7 Messages and Codes, Volume 1* indicates that more information is available in *IMS Version 7 Failure Analysis Structure Tables (FAST) for Dump Analysis*, refer to that book for diagnostic information (such as return codes) that you can use to build the search argument. The FAST also explains why the abend was issued, and often provides useful information for problem analysis.

If you cannot solve the problem by using the FAST, develop a search argument.

After you have developed a search argument, refer to Chapter 5, “Procedures and Techniques,” on page 47 for detailed information on how to use the search argument.

Keyword: ABENDUxxxx

Replace the xxxx part of the ABENDUxxxx keyword with the user abend code from either the termination message or the abend dump. User abends are always represented in decimal.

Keyword: Module Name

You can determine the name of the module that received the abend in either of the following ways:

- Check both the dump title and message DFS629I, which might contain the name of the abending module.
- Use the PSW address at the time of abend. You can find this address in the IMS-formatted section of the dump under the diagnostic area or in the MVS-formatted section. From the PSW address, scan backward through the eye-catchers until you find a module identifier. To relate a module identifier to a module name, see Appendix D, “Save-Area-ID-to-Module Cross-Reference Table,” on page 521.

Use the module name in the search argument for standard user abends only. For pseudoabends, do not include the module name as part of the argument. *IMS Version 7 Failure Analysis Structure Tables (FAST) for Dump Analysis* indicates whether the abend is a pseudoabend or a standard abend.

Abend-Specific Keywords

By examining the information in *IMS Version 7 Failure Analysis Structure Tables (FAST) for Dump Analysis*, you might gather additional keywords that can be pertinent to the problem, such as:

- User call function
- Internal call function
- Database organization
- Messages

  Replace the *xxxxxxx* part of keyword MSG*xxxxxxx* with the actual message identifier (for example, the keyword for message `DFS053I` is `MSGDFS053I`).
- Return codes

Replace the *xx* part of keyword RC*xx* with the associated hexadecimal return code (for example, the keyword for return code C is RC0C).

- Function codes

  Replace the xxxx part of keyword FCxxxx with the associated hexadecimal function code (for example, the keyword for function code 13 is FC0013).

Search Argument Example

If, for example, ABENDU3046 occurred in IMS module DFSPCC20 with message DFS3624I indicating function code 291 and return code 4, the search argument to use is:

```
5655B0100 ABENDU3046.
```

For a structured database search, use this search argument:

```
PIDS/5655B0100 AB/U3046
```

With this search argument, you might receive numerous hits, which would most likely include the APAR describing your problem. You can add keywords from the section "Abend-Specific Keywords" on page 21 to narrow the field of hits received. It is a good idea to use the **OR** operator on these additional keywords at first. Module name DFSPCC20 is not included as part of the search argument because ABENDU3046 is a pseudoabend.

The additional keywords for the above scenario are:

```
MSGDFS3624I | RC04 | FC0291
```

For a structured database search, use this search argument:

```
MS/DFS3624I PRCS/00000004 OPCS/0291
```

Additional Documentation

The IBM support center might ask you to obtain certain information to determine and resolve the problem. At times you might need to re-create the problem in order to gather this documentation.

For database problems, ensure that you have access to the following documentation before calling the IBM support center:

- A dump of the problem
- DBDGENs
- PSBGENs
- A copy of the databases involved in the error
- Logs and archive tapes that might have activity against the databases
- Output from both the DL/I and LOCK traces
- When tracing to the log, a printout of the traces
- A current CDS list or a current SMP/E target zone
- A current assembly listing of DFSADSCT from IMS.ADFSSRC (control block DSECTs)

Problems can be resolved more quickly if the documentation listed above is available.

IRLM Procedure

Use this procedure when the IRLM terminates abnormally.

1. Locate the PSW and register contents at entry to the abend either from the software LOGREC entry or from the RTM2WA summary in the formatted section of the SDUMP.
   a. If the PSW is not within an IRLM module (prefixed with DXR), determine the system component in which the abend occurred and use the diagnostic procedure for that component to resolve the problem.

b. If the RTM2WA summary entry shows that the IRLM was terminated by an abend completion code of U2017, U2018, U2019, U2020, U2022, U2023, U2024, U2025, U2027, U2031 (X'7E1', X'7E2', X'7E3', X'7E4', X'7E6', X'7E7', X'7E8', X'7E9', X'7EB', or X'7EF'), the IRLM task was terminated because of an error either in a subtask or in an SRB related to the IRLM. To diagnose the problem, use the software LOGREC entry or the RTM2WA summary entry for the original error in the subtask or related SRB.

2. Register 12 normally contains the base register contents for the module that was in control at the time of the error.
3. Register 9 normally contains the address of the RLMCB if the error occurred during IRLM processing.
4. Using the module name, find the function keyword and refer to Appendix C, "Module-to-Function-to-Subfunction List," on page 465 to locate the function and subfunction keywords.

Example: An example of a search argument for an IRLM problem is:

```
569516401 ABEND0C4 DXRRL200
```

For a structured database search, an example is:

```
PIDS/569516401 AB/S00C4 RIDS/DXRRL200
```

DOC Procedure

You can report publication problems to IBM by using one of the methods described on the form for readers' comments located in the back of each book. Corrections resulting from readers' comments are included in future editions of the manual, but are not included in the software support database.

If a problem can have severe results or cause lost time for many other users, contact the IBM Support Center to initiate a documentation change. Have the following information available:

- The order number of the manual
- The page number of the error in the manual and a description of the problem it caused

APARs are not generally accepted for publication errors. However, APARs that correct a programming error can result in documentation changes. You can search for changes to manuals using this procedure.

Keyword: Order-number

Use this keyword to search for all changes to a specific manual. The format for the order-number is *ppnnnnnnee*, where *pp* is the alphabetic prefix, *nnnnnn* is the six-digit base publication number, and *ee* is the edition number. For example, the order number for *IMS Version 7 Messages and Codes, Volume 1* is GC26-9433-00. Replace *ppnnnnnnee* with GC26943300. The edition number is optional. To broaden the search to include all editions of a manual, either omit the edition number or replace it with two asterisks (**).

Search Argument Example

Use this search argument to search for all changes to any edition of *IMS Version 7 Messages and Codes, Volume 1*:

```
5655B0100 GC269433**
```

For a structured database search, use this search argument:

```
PIDS/5655B0100 PUBS/GC269433**
```

You can add more keywords to narrow the search. For example, if you cannot find message DFS3007 in *IMS Version 7 Messages and Codes, Volume 1*, add this keyword to the above search argument:

```
MSGDFS3007
```

For a structured database search, use this search argument:

```
MS/DFS3007
```

If you do not find an APAR that adds message DFS3007, use one of the methods listed on the form for readers' comments in *IMS Version 7 Messages and Codes, Volume 1* to report the omission to IBM.

PERFM Procedure

Most performance problems are related to system tuning and should be handled by system programmers.

After you have developed a search argument, refer to Chapter 5, "Procedures and Techniques," on page 47 for detailed information on how to use the search argument.

Keyword: PERFM or PERFORMANCE

Always use the keywords PERFM and PERFORMANCE for performance problems. You should use the **OR** operator to link them together in the search argument.

Search Argument Example

You can use the following search argument to check for all performance APARs in IMS Fast Path:

```
5655B0100 PERFM | PERFORMANCE FAST | PATH | FASTPATH
```

For a structured database search, you can use this search argument:

```
PIDS/5655B0100 PERFM  | PERFORMANCE RIDS/FASTPATH
```

You can add the **OR** operator to the general component identifier together with the Fast Path component identifier as described in "Component Identification Keyword Procedure" on page 19. With this search argument, the resulting number of hits could be very large, but would include APARs describing performance problems in Fast Path.

You can add more keywords to narrow the number of hits. For example, if the performance problem occurs because of an excessive number of file opens and closes, you can add the **OR** operator with the following keywords to the above search argument:

```
OPEN | CLOSE
```

For a structured database search, use this search argument:

```
PCSS/OPEN  |  PCSS/CLOSE
```

If you cannot find an appropriate APAR with these search arguments, contact the IBM support center.

Appropriate documentation for performance problems might include:

- Traces, such as DL/I, lock, dispatcher, scheduler, external subsystem, and others, depending on the area of the performance problem
- Dumps of the problem during the period of performance degradation
- Dumps of the problem during normal periods, for comparison
- DB or IMS Monitor reports during the performance problem period
- DB or IMS Monitor reports during normal operations, for comparison
- Copy of the IMS log during the performance problem period
- Copy of the IMS log during the normal period, for comparison

If a coordinator controller (CCTL) application program experiences a performance problem in a Database Control (DBCTL) environment, you might need the following documentation in addition to that listed above:

- Any CCTL traces or monitor reports
- A dump of the CCTL subsystem during the period of performance degradation

MSG Procedure

IMS Version 7 Messages and Codes, Volume 1 describes IMS messages. If, after analyzing the message, you feel the message should not have been issued or describes an error condition, use the MSGxxxxxxxx keyword.

After you have developed a search argument, refer to Chapter 5, "Procedures and Techniques," on page 47 for detailed information on how to use the search argument.

Keyword: MSGxxxxxxxx

Replace the *xxxxxxxx* part of keyword MSG*xxxxxxxx* with the actual message identifier (for example, the keyword for message `DFS0861` is `MSGDFS0861`).

Search Argument Example

If, for example, you receive message `DFS3401I RACF NOT AVAILABLE`, and you determine that RACF® is indeed available in your system, the search argument to use is:

```
5655B0100 MSGDFS3401I
```

For a structured database search, use this search argument:

```
PIDS/5655B0100 MS/DFS3401I
```

INCORROUT Procedure

`INCORROUT` is defined as a condition when either of the following occurs:

- Output was expected, but not received (missing).
- Output was different from expected (incorrect).

Use the following procedure to determine the appropriate search argument. After you have developed a search argument, refer to Chapter 5, "Procedures and Techniques," on page 47 for detailed information on how to use the search argument.

Keyword: INCORROUT

Always use the keyword `INCORROUT` for problems related to incorrect or missing output.

Keyword: Utility Module Name

If the incorrect or missing output is associated with a utility, use the utility module name as a keyword. For example, if output from the File Select and Formatting Print utility (DFSERA10) is incorrect, use DFSERA10 as a keyword.

Keyword: Command

If the output from a command is missing or incorrect, use the first three letters of the command as a keyword. Also, you should use the **OR** operator in the search argument with `CMDxxx`, where *xxx* is replaced by the first three letters of the command.

If, for example, the `DISPLAY` command provides incorrect output, use the following search argument:

```
5655B0100 INCORROUT DIS | CMDDIS
```

For a structured database search, use this search argument:

```
PIDS/5655B0100 INCORROUT PCSS/DIS
```

If applicable, you can add the output column or heading as a keyword in the search argument. (See "Keywords: Columns, Headings, Fields" on page 26.)

Keywords: Columns, Headings, Fields

Whenever possible, you can add additional keywords to narrow the field of hits. If a particular heading, field name, or column is incorrect, use it as a keyword. For example, if the deadlock event summary section of the IMS Monitor report (DFSUTR20) is incorrect for the DMB NAME column, use the following search argument:

```
5655B0100 INCORROUT DFSUTR20 DEADLOCK | DMB
```

For a structured database search, use this search argument:

```
PIDS/5655B0100 INCORROUT RIDS/DFSURT20
PCSS/DEADLOCK PCSS/DMB
```

If you receive too many hits, remove the **OR** operator (|) to focus the selection.

Keyword: Database Type or Call

If the incorrect output is a database record, use the database type (such as VSAM, HDAM, or HIDAM) and possibly the call (such as GU, ISRT, or DELETE).

Additional Diagnostics

This section does not apply to a Database Control (DBCTL) environment.

If the output is a transaction message produced as output from an application program, perform the steps below. (The message can be directed either to a terminal or to another application program. This is called a program switch.)

1. If the output is missing, continue with this step; otherwise, go to step 2 on page 27.
   a. When the output is missing, determine if the transaction is being scheduled.
      - Issue the `/DIS ACTIVE` command to make sure the transaction is not stopped.
      - Then issue the `/DIS TRAN` command to find out if the transaction is scheduled.

      QCT should decrease by at least one each time the transaction is scheduled and terminates normally.

      If the transaction is not being scheduled, go to step 1f on page 27.
   b. Determine if the message is being enqueued to the proper output destination by issuing one of the following commands:
      - Issue the `/DIS TRAN` command (for program switch). ENQCT should increase.
      - Issue the `/DIS LTERM` command (for output to terminal). ENQCT should increase.

      If the message is not being enqueued to the proper output destination, go to step 1e.
   c. If the output destination is another application program, it should be scheduled as a result of the message enqueue.

      If the transaction is scheduled but there is no input, the problem is probably within the SYS function.

      If the application program is not scheduled, go to step 1f on page 27.
   d. If the output destination is a terminal, verify that I/O errors did not prevent the message from being sent. Take both of the following actions.
      - Review the console log for I/O error messages.
      - Issue the `/DIS LTERM` command for operational status.

      If you detected valid I/O errors, stop here and correct the hardware problem. Otherwise, the problem is probably within the TM function. Stop here and build your search argument.
   e. Determine if the application program is using the proper PCB for the `ISRT` call.
      - Force a dump in the application program at the time of the `ISRT` call.

      If the proper PCB is being used, the problem is probably within the SYS function. Stop here and build your search argument. Otherwise, stop here and correct the application program.

f. Determine if the resources necessary to schedule the application program are available.
   - Issue the /DIS ACTIVE command for the active region.
   - Issue the /DIS SUBSYS ALL command for all external subsystems connected to or in the process of being connected to IMS.
   - Issue the /DIS TRAN command to make sure the transaction is not stopped.
   - Issue the /DIS DATABASE command to determine if the necessary databases are available.

   If a resource is not available, stop here and make it available. Otherwise, force a console dump. Use the PST ANALYSIS step in procedure “WAIT/LOOP Procedure” on page 28 to determine the reason the transaction is not being scheduled. Stop here and build your search argument using that information.

2. If the incorrect data is input to an application, perform this step, otherwise go to step 3.
   a. Verify the text data in the X'01' log record to determine if the data reached IMS properly.

      If the data did not reach IMS properly, go to step 2c.
   b. Force a dump in the application program immediately after the application program GU call, in order to determine if the data reached the I/O area correctly.

      If the data did not reach the I/O area correctly, the problem is probably within the SYS function. Stop here and report the problem. Otherwise, the application program received the data correctly. Stop here.
   c. Start the line or node trace and verify the data in the X'6701' log record to determine if the data reached the input TP buffer correctly.

      If the data reached the input TP buffer correctly, the problem is probably within the DC function. Stop here and report the problem. Otherwise, if the data did not reach the input TP buffer correctly, the problem is probably a hardware or an operating system failure. Stop here and correct the hardware or operating system problem.
3. Determine if the message data is actually incorrect rather than merely formatted incorrectly.
   - Compare received data with expected data.
   - Check MFS blocks for correct format definition.

   a. Force a dump in the application program just before the ISRT call to determine whether the data is correct in the I/O area at the time of the ISRT.

      If the data in the I/O area is incorrect, the problem is probably in the application program. Stop here and correct the application program. Otherwise, continue. Verify the text in the X'03' log record to determine whether the data reached the message queue correctly.

      If the message did not reach the message queue correctly, the problem is probably within the SYS function. Stop here and build your search argument. Otherwise, continue.
   b. Start the line or node trace and verify the data in the X'6701' log records, in order to determine if the data reached the output TP buffer correctly.

      If the data did not reach the output TP buffer correctly, the problem is probably within the DC function. Stop here and build your search argument. Otherwise, if the data is correct in the output TP buffer, but not at the terminal, the problem is probably a hardware or operating system failure. Stop here and correct the hardware or operating system problem.

IRLM Problems

Incorrect output from the IRLM can be divided into the following three areas:
- Incorrect information on a display status command
- Locks granted when locks should not be granted
- Locks not granted when locks should be granted

For help in diagnosing these problems, call the IBM Support Center. A support representative will tell you what type of documentation to gather.

WAIT/LOOP Procedure

The procedures for the WAIT and LOOP keywords are combined because the WAIT and LOOP symptoms might not be distinguishable at first. Use the following procedure to determine the type of WAIT or LOOP occurring, and to find the appropriate keywords for the problem.

Be aware that maintenance might change the offsets in these control blocks. For a current version of the control blocks assemble DFSADSCT.

1. Is IMS being shut down?
   - If the operator issued a `CHECKPOINT DUMPQ`, `PURGE`, or `FREEZE` command before the manifestation of the wait/loop, go to “Shutdown Processing” on page 41.
   - If IMS is not being shutdown, continue with the next step.
2. Determine whether IMS was in selective dispatching mode.

   Find the dispatch work areas in the formatted dump. The dispatch work areas are created using the `DISPATCH` or `All` IMS dump formatting options. The dispatch work area eye catcher is `**DSP`.

   The selective dispatch bits are in the SFLAGS field in the `DYNAMIC SAP EXT.` section, where the X'xxxxxx8x' bit represents selective dispatching. To determine whether selective dispatching was entered for save area prefixes (SAPs), search the `DISPATCH AREA` section for the following message:

   ```
   *** NOTE: THIS TCB IS IN SELECTIVE DISPATCHING FOR SAPS
   ```

   If you find this message, IMS wrote a X'450F' log record to the OLDS. This log record contains information about dynamic SAPs, such as the highest number of dynamic SAPs used and the number of times IMS was in selective dispatch for dynamic SAPs.

   | Examine this X'450F' log record to help determine what might have led to the shortage of dynamic SAPs. Then go to the “SAP Analysis Procedure” on page 31. While performing SAP analysis, keep in mind that the dynamic SAPs are labeled `DYNAMIC SAP`, and that the CURRENT TCB= indicates the associated task control block (TCB).

   If IMS is not in selective dispatching mode, continue with the next step.
3. Can the operator communicate with IMS through the MVS system console by using the IMS outstanding reply to enter an IMS command, such as `/DISPLAY`?
   - If no, or if you are not sure, go to step 5 on page 29 now.
   - If yes, the problem might be caused by:
     - A data communications failure.
     - The inability of a task to acquire a resource.
     - Non-completion of an event, such as I/O.

     Continue with the next step.
4. Can the IMS master terminal operator (MTO) communicate with IMS by issuing various IMS commands, such as `/DISPLAY`?
   - If yes, go to “SAP Analysis Procedure” on page 31.
   - If no, the problem might be data communication related. If IMS is still running, do the following:
     - Issue the IMS `/DIS NODE` *nodename* command. Save the IMS console output.
     - Turn on the IMS node trace with the `/TRA SET ON NODE` *nodename* command.

       Data is captured in the IMS X'6701' log record. Save the IMS OLDS for execution with IMS utility programs DFSERA10/DFSERA30.
     - Consider turning the VTAM buffer trace and VTAM internal trace on to complement the IMS node trace, as follows:

       ```
       F NET,TRACE,TYPE=BUF,ID=nodename
       F NET,TRACE,TYPE=VTAM,MODE=EXT,OPT=(API,PIU,MSG)
       ```

       GTF must be active for this option.
     - Obtain a dump of the IMS and VTAM regions using this series of commands:

```
DUMP COMM=(dump title)
R id JOBNAME=(j1,j2,j3,j4,j5,j6,j7),SDATA=(CSA,PSA,RGN,SQA,SUM,TRT),END
```

The variables have the following meanings:

j1 IMS CTL region job name.

j2 VTAM region job name.

j3 IMS DLI region job name.

j4 Suspicious IMS dependent region job name, if any.

j5 Suspicious CCTL (CICS) region name, if any.

j6 DBRC region job name.

j7 IRLM region job name (if IRLM database locking was used).

The jobs are listed in order of importance.

Recommendations: A dump of the IMS CTL, VTAM, DLI, and suspicious dependent region or CCTL is usually sufficient to solve wait/hang problems. Occasionally, the DBRC and IRLM (if they are used for database locking) can be a factor. Therefore, you should also include them.

SYS1.DUMP data sets are often not large enough to hold all regions requested in the `DUMP` command. Make them large enough to hold the regions. If the MVS SVC `DUMP` command fails due to lack of space, take separate dumps in smaller combinations to accommodate the smaller SYS1.DUMP data set size.

– Go to the “SAP Analysis Procedure” on page 31. If SAP analysis does not yield any unusual flows, go to “Receive-Any Buffer Analysis” on page 290.

5. Query the IMS Dispatch Work Areas.
   a. Find the Dispatch Work Areas in the formatted dump. The Dispatch Work Areas are created using the `DISPATCH` or `ALL` IMS dump formatting options. The Dispatch Work Area eye catcher is `**DSP`.
   b. Scan **each** Dispatch Work Area (STM, CTL, RST RDS, and so on) except for the DRC and dependent region entries (labeled DEP, MPP, BMP, DBT, DRA, or IFP). Examine the QPOST field at offset X'1C'.

      If the high-order bit of the QPOST field is off, note the address and type of Dispatch Work Area.
   c. If, after scanning **all** Dispatch Work Areas, **except** for the DBRC (DRC) task and dependent regions, you find that the QPOST high-order bit is always set, one of the following is true:
      - IMS is in an IMS WAIT (IWAIT) state. Go to “SAP Analysis Procedure” on page 31 now.
      - If at least one Dispatch Work Area has the high-order bit off, this is a LOOP or operating system WAIT. Continue with the next step.
6. Query the TCB/RB chain.
   a. Find the current ECB, ASID, and TCB address for each Dispatch Work Area noted previously in step 5b.
      - In IDSPWRK SECTION 1, find field CECB at offset X'28'. The field CECB at offset X'28' contains the ECB of the current dispatched ECB.
      - In IDSPWRK SECTION 1, find the field ASIDS at offset X'30'. The first halfword of the field ASIDS at offset X'30' contains the ASID number for the task; the second halfword contains the CTL region ASID.
      - In IDSPWRK SECTION 1, find the field TCB at offset X'40'. The field TCB at offset X'40' contains the TCB address for the task.
   b. Find the formatted TCB/RB chain in the MVS formatted dump. Use the IPCS `SUMMARY FORMAT ASID(X'__')` command for the ASID/TCB found in step 6a. Use the following `FIND` command to locate the TCB:

      ```
      F 'TCB: xxxxxxxx' 1 16
      ```

where *xxxxxxxx* is the 8-character TCB address, including leading zeros.

c. Examine the request block (RB) structure (PRBs, SVRBs, or IRBs), focusing on the last RB in the chain for that TCB. The TCBRBP field at offset X'00' contains the address of the last RB. Use the following `FIND` command to locate the RB:

```
F 'RB: xxxxxxxx' 1 16
```

where *xxxxxxx* is the 8-character RB address, including leading zeros.

Exception: Using the last RB in the TCB's RB chain is usually accurate. However, there are occasions when additional RBs might be appended to the end of the chain to facilitate dump processing, but they have nothing to do with the problem. X'00020033' in the WLIC field in any RB in the RB chain normally indicates dump processing. In such a case, examine the RBs prior to the RB with WLIC=X'000020033'. If the RB prior to the RB containing WLIC=X'00020033' contains WLIC=X'0002000C, it might be necessary to examine the RB prior to the RB containing WLIC=X'00002000C'.

Example:

```
PRB  WLIC = X'00020006'
PRB  WLIC = X'00020078'
SVRB WLIC = X'0002000C'  Examine prior RB.
SVRB WLIC = X'00020033'  <== Indicates dump processing
SVRB WLIC = X'00020078'
```

d. Examine the LINK field in the RB found in step 6c. The high-order byte of the LINK field is the wait count field.

- **If the wait count = X'00'**, this usually indicates that the task is looping. Do the following:
  - Perform system loop diagnostics. Obtain the OPSW and registers from the looping RB, (located in the following RB or in the TCB, if this is the last RB (TCBRBP)) for a snapshot of the loop.
  - Obtain the PSW address from the MVS SYSTEM TRACE TABLE. Use the IPCS `VERBX TRACE ASID(`*xx*`)` command to obtain the entries for the ASID in question. Focus on the entries for the TCB found in step 6a on page 29. You can ignore entries between any SVC and associated SVCR because they reflect necessary MVS operating system activity indirectly involved in the loop. (The IMS TYPE2 SVC is an exception to this since it results in execution of IMS code.) Sorting the pertinent addresses by OPSW address greatly aids in laying out the loop.
  - Resolve the PSW address found by using either IPCS BROWSE mode, the IPCS `WHERE` command, or by using an LPA or NUCLEUS MAP to obtain the name of the modules involved in the loop. The IPCS commands used to obtain the maps are `LPAMAP`, and `VERBX NUCMAP`, respectively. Calculate the offset at which the instruction appears in the modules to outline the path of the loop.
  - Another source of information for the looping task can sometimes be found at the top of the IMS SAPS AND SAVEAREA section (`**SSA`) of the IMS formatted dump. Look for the `**** A C T I V E ****` save area set nearest the top of the `**SSA` with the SAPECB filed matching the CECB field obtained in step 6a on page 29. The save area flow can indicate IMS modules involved in the loop or those passing control to the looping function.
- **If the wait count is not = X'00'** (that is, = X'01', X'02', and so on), this usually indicates that a system WAIT occurred. Do the following:
  - Obtain the address portion of the OPSW. It points to the waiting module.
  - Resolve the PSW address found by using either IPCS BROWSE mode, the IPCS `WHERE` command, or by using an LPA or NUCLEUS MAP to obtain the name of the waiting module. The IPCS commands used to obtain the maps are `LPAMAP`, and `VERBX NUCMAP`, respectively. Calculate the offset at which the wait occurred in the module. This information can be used for APAR searches and/or for contact with the owning component's IBM Support Center representatives.
  - Use the CECB field obtained in step 6a on page 29 to find the related SAP save area by scanning for the SAPECB match in the IMS formatted dump **SSA section.

SAP Analysis Procedure

1. Find the formatted SAP AND SAVE AREA section in the IMS formatted dump.

   Choose the SAVEAREA, SYSTEM, ALL or SAVEAREA,SUM options of the IMS Offline Dump Formatter. The eye catcher of the SAP AND SAVE AREA section is `**SSA`.

   Table 2 defines the key fields in SAP analysis.

Table 2. Key Fields in SAP Analysis

Offset	Field Name	Length	Field Description
SAP+X'00'	SAPFLAG1	1	X'80' = Active SAP X'40' = Waiting SAP
SAP+X'01'	SAPDSPCD	1	IMS TCB number. This number matches the associated TCB number at offset X'3B' in the dispatch work area.
SAP+X'14'	SAPIWAIT	4	In waiting SAPs, this is the address of the last active save area. Those below this address are residual. In SAPs that are active but not waiting, this field is residual and should not be used. **Exception:** SAPIWAIT might not be valid for Fast Path save area sets (DBF-prefixed modules). The active save area set usually ends with DBFXSL30, the Fast Path wait module, unless DFSIWAIT or DFSISERW appears previously in a save area set.
SAP+X'18'	SAPECB	4	Address of the ECB associated with this ITASK. If the PST is used, this field points to the beginning of the PST.
SAP+X'24'	SAPCDSP	4	Address of the current dispatch work area.
SAP+X'30'	SAPSDPNO	4	Dispatch number for the ITASK.

2. Begin SAP analysis at the end of the sorted SAPs.

   Find the end of the sorted SAPS. Eye-catcher `***END OF SORTED SAP FORMATTING` marks the end of the list. SAPs are sorted by the SAPSDPNO (system dispatch number). The most recently dispatched ITASKs are at the end of the sorted SAPs. These are the ITASKS that have been waiting the longest and possibly causing the other ITASKS to wait behind them by holding a resource, such as a lock or a latch.

3. Scan backwards from the end, examining only active or waiting SAPs. Focus **only** on the active save area sets (that is, SAPFLAG1 has the X'00' bit turned on (X'08', X'Cx', X'Dx', X'Fx')). Active save area sets are marked with the eye-catcher `**** W A I T I N G ****` or `**** A C T I V E ****`. To find waiting or active SAPs, use the following find command: `F '   **** ' PREV`.

   Remember that the SAVEAREA,SUM option of the Offline Dump Formatter produces only active save area sets. Active running SAPs are marked with eye-catcher `RUN`. The end of this formatting is marked by eye-catcher `****** END SAP SUMMARY`.

4. Skip over all normal save area sets.

   This step describes all normal save area sets. After you have identified all types of normal save area sets, you can disregard them as they are unrelated to the problem.

   a. `WAITING` save area sets in which module name DFSIWAIT appears after label EP at the second-level save area are considered normal save area sets.

      The following example shows a normal save area set at the second level:

```
***SAVE AREA SET***
   EP DFSQMRT0-11/13/94
   SA 00133BC4        WD1 8091E430   HSA 80000000   LSA 00133C0C ...

   EP DFSIWAIT
```

```
SA 00133C0C         WD1 00000000   HSA 00133BC4   LSA 00133C54 ...

EP DFSFLLG0-220-PL46803
SA 00133C54         WD1 00000000   HSA 00133C0C   LSA 00133C9C ...
......
```

b. The only normal save area sets in which the save area set contains DFSIWAIT at the third level are shown in the example below. Be sure that register 08 contains a value of X'00000003' for any of the first four save area sets, as shown below. Otherwise, it is abnormal and indicates an intent conflict as described in the “Intent Conflict” on page 38. Use the SAPSECB field to obtain the PST address for use in the intent conflict procedure.

```
EP DFSSMIC0 --> EP SMSC2    --> EP DFSIWAIT with REG08 = x'00000003'
  EP DFSSMIC0 --> EP DFSSMSC2 --> EP DFSIWAIT with
  REG08 = x'00000003'
  EP DFSSMIC0 --> EP DFSSMSC1 --> EP DFSIWAIT with
  REG08 = x'00000003'
  EP DFSSMIC0 --> EP MPPENQ00 --> EP DFSIWAIT with REG08 = x'00000003'

  EP DFSFXC30 --> EP DFSFXC30-WFITEST  --> EP DFSIWAIT
  EP DFSVTP00 --> EP VTPOWORK --> EP DFSIWAIT
  EP DBFHCL00 --> EP DBFHGU10 --> DBFXSL30
```

c. The only normal save area sets in which the save area contains DFSIWAIT at the fourth level are those listed below. Be sure that register 08 in the DFSIWAIT save area set contains X'00000003'. Otherwise, it is abnormal and indicates an intent conflict as described in “Intent Conflict” on page 38. Use the SAPSECB field to obtain the PST address for use in the intent conflict procedure.

The following examples show normal save area sets at the fourth level:

```
DFSSMIC0 --> DFSSMSC0 --> SMSC1000 --> DFSIWAIT  REG08 = x'00000003'
DFSFXC30 --> DFSDLA30 --> DLA32000 --> DFSIWAIT
```

d. The following active save area sets are probably normal, so you can ignore them.

- Save area sets marked `ACTIVE` or `RUN` with SAPDSPCD=X'07'. This is a DRC task SAP. This condition is usually normal for the DBRC task.
- Save area sets marked `ACTIVE` or `RUN` with SAPDSPCD=X'0F'. This is the ESI task SAP if SAPCDSP=X'00000000'.
- Dependent region save area sets marked `ACTIVE` with SAPDSPCD=X'03'(MPP), X'04'(BMP), X'0D'(DRA), X'12' (IFP), X'13'(DBT), X'0C' (ESS), or X'00' (RESIDUAL), in which the top save area indicates it was returned. (The last bit of the address in the field labeled RET, which is register 14, is odd or has X'FF' in the high-order byte.)
- If the SAPDSPCD=X'13'(DBT), and the first save area EPA is marked `UNKNOWN` with the second-level save area RET field marked returned (the last bit of the address in RET is odd), this is a normal save area set if the first save area EPA is within module DFSDASC0 or DFSDAST0.

5. Obtain abnormal save area set information.

The remaining save area sets (those that are `ACTIVE` or `WAITING`, but abnormal, as described in step 4 are involved in the wait in some way.

Recommendation: Concentrate on one save area set at a time, beginning with the first abnormal save area set. Remember to start from the end of the sorted SAPs.

If you find an abnormal save area set marked `**** A C T I V E ****` (SAPFLAG1=X'80'), the problem is associated with the TCB/RB save area set. Use the address of the current dispatch area in SAPCDSP to find the dispatch work area associated with this save area set. Go to step 6b in the “WAIT/LOOP Procedure” on page 28. Continue from there, using the ASID/TCB obtained from the dispatch work area. If the high-order bit in QPOST is on (QPOST=X'8x'), this SAP is suspended. Record this save area set and continue to the next abnormal save area set. Discontinue step 6b because this save area set should probably be ignored. Otherwise, continue.

Record the following key fields from the abnormal save area sets flagged as `**** W A I T I N G ****`:

a. The address of the SAP.

b. For each save area in the save area set, from the first save area down to the save area pointed to by the SAPIWAIT field, obtain the following information. (See exception for SAPIWAIT in Table 2 on page 31 before proceeding.)
   1) EP module name
   2) APAR level (the APAR number and last few letters of the changeid string)
   3) RET address (this is register 14)
   4) EPA address

   If the module name is `UNKNOWN` and the module save area set begins with DFSDLA00, the EPA address can probably be resolved in the DLI region dump by using IPCS BROWSE mode for the DLI ASID.
c. The offset from which DFSIWAIT, DFSISERW, or DBFXSL was invoked from the calling module.

   You can calculate the offset by subtracting the EPA address in the save area **before** the save area pointed to by SAPIWAIT from the RET address of the save area pointed to by SAPIWAIT.

   Table 3 shows key data from an abnormal save area set.

Table 3. Key Data from an Abnormal Save Area Set

EP Module Name	APAR Number	Last Few Changeids	RET	EPA	Wait Call Offset
DFSCST00	PL45938	abcde	80A7BA14	00A8E110	
DFSDBDR0	PL49770	..mnopr	60A8E6D6	00A07A58	
DFSBML00	none		50A07AC2	00B5DAE0	X'10E'
DFSIWAIT	none		40B5DBEE	70A7C7F6	

6. Identify the reason for the WAIT.

   To identify the reason for the WAIT, do the following:
   a. Use the Appendix F, “Module-to-Waiting-Resource List,” on page 547 for a brief description for some of the IMS waits that are issued.
   b. Assemble the module that issued the wait. Use the offset obtained in step 5c as an approximate displacement into the module where an IWAIT or ISERWAIT was issued. Examine the code and comments at that point. Most modules give the reason for the IWAIT in the comments above the IWAIT issue point.

      The EP name might not be the actual module name, but rather a CSECT within a module. To find the actual module name, do one of the following:
      - Use Appendix D, “Save-Area-ID-to-Module Cross-Reference Table,” on page 521 to obtain the actual module name.
      - Using IPCS BROWSE mode, scan backwards from the EPA address for the actual module name.
7. Repeat steps 5 and 6 for the first three abnormal save area sets you found.

   You should be able to gather enough information from the first three abnormal save area sets to perform a search or determine the cause of the problem.

Keyword: WAIT

At this point, you can be sure that you are in an IMS WAIT. Therefore, WAIT is an appropriate keyword for the search argument.

Keyword: Module Name Issuing IWAIT or ISERWAIT

The Module Name column in your worksheet indicates the modules that issued the IWAITs. These modules can provide useful search arguments. Use the eight-character module name for this keyword.

Keyword: WAIT Reason

The IWAIT REASON column in your worksheet indicates the reason and/or resource that is causing the IMS WAIT.

For example, if the reason was a WAIT for the DPST latch, the IWAIT REASON keyword is DPST LATCH.

Keyword: Additional Related Keywords

External events might trigger WAITs. These events might be indicated by console messages, or they might be related to a procedure that was being performed at the time the WAIT began.

You can use each of these additional keywords in the search argument when applicable.

Search Argument Example

Consider this scenario:

- IMS went into a IWAIT after a WADS write error occurred.
- Multiple unusual save area sets were found from module DFSFLLG0.
- The reason for the IWAIT was found to be the LOG LATCH.

The broad search argument to use is:

```
5655B0100 WAIT LOG | LATCH | W ADS | DFSFLLG0
```

For a structured database search, use this search argument:

```
PIDS/5655B0100 WAIT PCSS/LOG | PCSS/LATCH | PCSS/WADS | RIDS/DFSFLLG0
```

With this search argument, you might receive numerous hits, which will probably contain the APAR describing your problem. You can then take various combinations of the additional keywords that were compared with the **OR** operator in the above example and use the **AND** operator on the keywords instead. You can use this technique to narrow your field of search until you find the appropriate APAR.

PST Analysis

This section deals with analyzing regions for possible problems in scheduling, intent conflicts, and so forth.

1. Determine the number of active regions.

   SCDREGCT at SCD+X'BCE' is a 2-byte field that contains the number of active regions, if any.

   If SCDREGCT = X'0000', no regions are active. Go back to "SAP Analysis Procedure" on page 31.

   If SCDREGCT is not equal to X'0000', go to step 2.

2. Determine if the scheduler sequence queues (SSQs) have any entries.

   Obtain the address of the transaction anchor block (TAB) from the SCDTAB field in the DSECT (label TABEP in the formatted dump). The TAB, which is mapped by DSECT DFSTAB, consists of:

   TAB header

   Headers for each of the six subqueues (SSQ1 - SSQ6)

   Class vector table (CVT)

   Transaction class tables (TCTs)

   If the count of partition specification tables (PSTs) waiting on any subqueue (field TABSCHQC) equals 0, no region should be waiting on any subqueue. However, you should also check each subqueue header. Calculate the address of the subqueue header for a specific subqueue (SSQ#) as follows:

   ```
   SSQ# × X'18' − X'8' = offset of header for SSQ#
   Offset of header for SSQ# + SCDTAB address = address of header for SSQ#
   ```

   Perform this calculation for each subqueue number. If field TABSSQ*n*F, where *n* is the subqueue number, is not zero, this field contains the address of an entry on the SSQ for the specified subqueue.

   a. The SSQ consists of six subqueues. All subqueues are formatted in a dump, but subqueues 1 and 2 are unused.

b. Each subqueue represents a resource. A PST enqueued on a subqueue is waiting for that resource.

c. The TAB and SSQs are formatted after the `SCD LATCH EXTENSION` in an IMS formatted dump, as follows:

```
**TAB - TRANSACTION ANCHOR BLOCK**

0D1873B0                    005800FF 00000000    *       ........*
0D1873C0  0000000E 00000000 00000000 00000000    *................*
0D1873D0  00000000 00000000 00000000 00000000    *................*
    LINES   0D1873E0-0D1873EF   SAME AS THE ABOVE
0D1873F0  00000000 00000000 0CF18544 0CF00C40    *.........1...0. *
0D187400  00000000 00000000 00003614 00000000    *................*
0D187410  0CF18C40 0CF18C40 00000000 00000000    *.1. .1. ........*
0D187420  00003AEB 00000000 00000000 00000000    *................*
0D187430  00000000 00000000 0000396E 00000000    *................*
0D187440  00000000 00000000 00000000 00000000    *................*
0D187450  000010B4 00000000 0D187858 0D1878B0    *................*
0D187460  0D187908 0D187960 0D1879B8 0D187A10    *................*
0D187470  0D187A68 0D187AC0 0D187B18 0D187B70    *................*
........
........
........
........

    ***SCHEDULER SEQUENCE QUEUES***

    DFSPSTQE  00000000     SUBQ  1            NOT ACTIVE
                           SUBQ  2            NOT ACTIVE
                           SUBQ  3            NOT ACTIVE
                           SUBQ  4            NOT ACTIVE
                           SUBQ  5            NOT ACTIVE
                           SUBQ  6            NOT ACTIVE
```

d. If the words `NOT ACTIVE` follow the subqueue entry, no PSTs are enqueued on that entry.

e. If entries are listed for subqueue 3, go to “No Work to Do” on page 36.

f. If no entries are listed for subqueue 3, go to step 3.

3. Are there subqueue 4 or 5 entries?

   Subqueue 4 does not apply to a DBCTL environment.

   Entries on subqueue 4 or 5 are waiting for intent conflicts to be resolved.

   a. If entries are listed for subqueue 4 or 5, go to “Intent Conflict” on page 38.

   b. If not, go to step 4.

4. Are there subqueue 6 entries?

   This step does not apply to a DBCTL environment. Continue with the next step.

   Entries on subqueue 6 are waiting for input.

   a. If there are entries listed for subqueue 6, go to “WAIT for Input” on page 39.

   b. If there are no entries, go to step 5.

5. Are all regions accounted for?

   Compare the number of regions in the SCDREGCT (SCD+X'BCE') with the number of regions enqueued on the subqueues. (The SCDREGCT is 2 bytes.)

   a. If the numbers of regions are equal, go to step 6.

   b. If the numbers of regions are not equal, all regions are unaccounted for. Go to the analysis for “PST Active” on page 36.

6. Report the problem.

   This problem occurs when there are entries queued on the subqueues and no reason can be found to prevent their scheduling, but nothing schedules. Report the problem to the IBM Support Center.

PST Active

You reach this point in the analysis either when:

- The SCDREGCT field is not equal to zero, and there are no entries on the Scheduler Sequence Queues, or
- No problem was found in analyzing the PSTs on the subqueues, and the number of PSTs on the subqueues is less than that in the SCDREGCT field.

1. Locate the PSTs.

   Find the stack of dependent region PSTs in the dump. (Two stacks of PSTs exist in the dump. System PSTs are printed separately from the dependent region PSTs.)
2. Is the PST scheduled?
   a. Find all the PSTs with PSTTERM (X'1BC') = X'02' (ACTIVE) and PSTCODE1 (X'B7A') = X'10' (SCHEDULED).
   b. Ignore the PSTs without the SCHEDULED bit on.
3. For the scheduled PSTs, do SAP analysis.
   a. PST at offset minus X'04' (field name PTR) is usually the SAP address. (The PTR field is the last entry on the line above the X'0000' line in the dump.) If not, PST + X'5B8' (PSTSAV1) is the address of the first Save Area in a set, and WD1 in that Save Area is the address of the SAP.
   b. Go to “SAP Analysis Procedure” on page 31. Return here after doing SAP analysis for the scheduled PSTs only.
4. Are there any ACTIVE non WAITING SAPs?
   a. If any of the SAPs are marked ACTIVE go to step 5.
   b. If SAPs are found WAITING, use normal SAP analysis to report the problem. Use the search argument format on page 33.
5. Is the dependent region active within an IMS save area set?
   a. If SAP +X'08' (SAPCNTRL) = X'10', this region is in a DL/I call within IMS. Go to step 6.
   b. Otherwise go to step 7.
6. Analyze the region dump.

   You must analyze the region dump using the PSW address to identify the problem. Refer to WAIT/LOOP Procedure, steps 6c on page 30 and 6d on page 30.
7. Determine what the application program is doing.

   You must analyze the region dump using the PSW address to identify what the application program is doing.

   In a DBCTL environment, you must analyze the CCTL region dump using the PSW address to find out what the DRA, CCTL, or application program is doing. Refer to WAIT/LOOP Procedure, steps 6c on page 30 and 6d on page 30.
8. Determine the reason the latch is not freed.

   If a latch is being waited for, and the owner is not waiting for I/O, use SAP analysis to identify the reason for the WAIT.

No Work to Do

This section does not apply to a DBCTL environment.

You came to this point because there are PSTs on subqueue 3.

1. Locate the PSTs on subqueue 3.

   The addresses under the field name SQPSTADD are the PST addresses. In the formatted dump, the PSTs start with the eye-catcher `*** DB PST AREA ***`. Locate the PSTs that are on subqueue 3.
2. Find the classes the PSTs can execute.

   PST + X'118' (PSTCLASS) is a 4-byte field. Each byte indicates a class transaction that the PST is allowed to process.

If, for example:

PSTCLASS = 01030506

the PST can process classes 01, 03, 05, and 06.

3. For each PST on subqueue 3, locate the transaction class table (TCT) for each class that the PST can process. There is one TCT for each class.
   a. Obtain the TAB address from the SCDTAB.
   b. Take the first PSTCLASS value and subtract 1.
   c. Multiply this result by 4.
   d. Add this value to the TABCLASS offset value + X'70'.
   e. TCT = 4(first PSTCLASS value - 1) + X'70'.

      When the high-order byte contains a X'80' this indicates the TCT class is not active ***
4. Can any SMBs be scheduled?

   TCT+X'04' = *zero* or the address of an SMB that can be scheduled.
   a. If zero, no SMBs can be scheduled. Go to step 7.
   b. If SMBs can be scheduled, locate the SMBs and then go to step 5.
5. Is SMB locked or stopped?
   a. If SMB+X'24' (SMBSTATS) = X'10' (STOPPED) or X'08' (LOCKED), go to step 6.
   b. Otherwise, go to step 9.
6. Are there any more SMBs on this class?
   a. If SMB+X'04' (SMBQEFP) is not equal to zero, it is the address of the next SMB. Move on to the next SMB and repeat step 5.
   b. If SMB+X'04' (SMBQEFP) = zero, there are no more SMBs. Go to step 7.
7. Are all classes accounted for?
   a. If all classes found in PST + X'108' (PSTCLASS) are not accounted for, repeat step 4 for each remaining class.
   b. Otherwise, go to step 8.
8. Are all regions accounted for?

   To determine whether all regions are accounted for, use SCDREGCT (SCD + X'BCE'). The SCDREGCT is 2 bytes. There is one PST for each region.
   a. If the number of PSTs on subqueue 3 is equal to the SCDREGCT and they have been examined and accounted for, there are no transactions scheduled for the regions. This is a normal WAIT, and there is no work for IMS to perform. This is not a problem.
   b. Otherwise, go back to step 3 on page 35 to continue the scheduler queue analysis.
9. Locate the PSB directory (PDIR).

   If the SMB is not locked or stopped, locate the PDIR.

   SMB+X'3C' (SMBPDIR) = address of the PDIR.
10. Can PDIR schedule?

    Locate the PDIR entry. When any of the following bits are ON, the PDIR is unable to schedule.

    PDIR+X'20' (PDIRCODE) = X'40'X'10'X'08'X'02'
    a. If the PDIR cannot schedule, go back to step 6.
    b. Otherwise, go to step 11.
11. Is PDIR marked parallel?
    a. If the PDIR is marked scheduled but not parallel:
```
PDIR+X'20' (PDIRCODE) = X'04' (Scheduled)
and:
PDIR+X'21' (PDIROPTC) is not equal to X'04' (Not parallel)
```

If there are entries listed for subqueue 6, go to "WAIT for Input" on page 39 to determine if any of the waiters on subqueue 6 are pseudo WFIs scheduled against the same PDIR. If there is a pseudo WFI scheduled against the same PDIR, report the problem to the IBM Support Center.

If there are no entries listed for subqueue 6 or none of the waiters on subqueue 6 point to the same PDIR, go back to step 6 on page 37.

b. If marked parallel (PDIR+X'21' = X'04'), go to step 12.

12. Are enough messages enqueued for another PST?

If the PDIR is marked parallel, check if enough messages are enqueued on the SMB to schedule another PST.

a. You do this by finding:
   1) SMB+X'46' (SMBPARLM) = number of messages per region (2 bytes).
   2) SMB+X'44' (SMBRGNS) = number of message regions scheduled for the SMB (2 bytes).
   3) SMB+X'1A'(SMBENQCT) minus SMB +X'18' (SMBDEQCT) = number of messages currently enqueued. (To find the number currently enqueued, subtract the messages dequeued from those enqueued.)
b. If the number of messages currently enqueued (step 12a3) is greater than the number of messages per region (step 12a1) multiplied by the number of message regions scheduled (step 12a2), there are enough messages enqueued on the SMB to schedule another PST. Go back to step 6 on page 37.
c. Otherwise, go to step 13.

13. Report the problem.

At this point, regions are waiting, enqueued on subqueue 3 with transactions that can be scheduled. Report the problem to the IBM Support Center.

Intent Conflict

You reach this point by having entries on subqueue 4 or 5.

An intent problem is indicated when the PST is on the intent queue.

1. Locate the PSTs that are on subqueue 4 and/or subqueue 5.

   The addresses under the field name SQPSTADD are the PST addresses. To analyze the INTENT CONFLICT fields in a PST, you must locate the PST in the unformatted section of the dump.
2. Is the PSB work pool too small?
   a. If PST + X'B7A' (PSTCODE1) = X'06', the PST is on the PSB WAIT queue for pool space. The PSB work pool is too small. You must increase the size of the PSBW parameter in the DFSPBxxx member.
   b. Otherwise, go to step 3.
3. Is the Data Management Block (DMB) pool too small?
   a. If PST + X'B7A' (PSTCODE1) = X'20', the DMB pool is too small. You must increase the size of the DMB parameter in the DFSPBxxx member.
   b. Otherwise, go to step 4.
4. Can intent be satisfied?
   a. If PST + X'B7A' (PSTCODE1) = X'40', the intent cannot be satisfied. Go to step 6 on page 39.
   b. Otherwise, go to step 5.
5. Is the region scheduled?
   a. If any PST has the following:

      PST +X'B7A' (PSTCODE1) = X'10'(SCHEDULED)

      and:

      PST +X'1BC' (PSTTERM) = X'02'(ACTIVE)

the region is scheduled, and this a normal WAIT for subqueue 4 and subqueue 5. Usually this is not a problem. Go back to the subqueue 6 entry of PST Analysis, step 4 on page 35 and continue.

b. Otherwise, go to step 7.

6. There is an intent conflict.

   If you reach this point, there is an intent conflict. Usually, the intent conflict is caused by a PSB having the exclusive option. This option is defined during the PSBGEN. See the PSBGEN section of *IMS Version 7 Utilities Reference: Database and Transaction Manager*. If the exclusive option did not cause the intent conflict, report the problem to the IBM Support Center.

7. Report the problem.

   If you reach this point, the problem is that the last region to terminate should have posted the PST on subqueue 4 and subqueue 5 and did not. In a DBCTL environment, the last thread to unschedule a PSB did not post subqueue 4 or 5. Thus, there is a WAIT with a PST on subqueue 4 or subqueue 5 with no scheduled regions. Use subqueue 4 or subqueue 5 in your search argument, or report the problem to the IBM Support Center.

WAIT for Input

You can reach this point only by having entries on subqueue 6.

1. Find the PSTs on subqueue 6.

   The addresses under the field name SQPSTADD are the PST addresses. The PSTs are found in the stack of PSTs.

2. Find Scheduler Message Blocks (SMBs) for the PSTs.

   For each PST enqueued on subqueue 6, find the related SMB.

   PST +X'C0' (PSTSMB) = address of the SMB

3. Are any of the regions on subqueue 6 pseudo WFIs?

   - If SMB+X'27' (SMBFLAG3) = X'08' (WFI transaction), the region is not a pseudo WFI.
   - If the region is a pseudo WFI, check if the region is holding any resources needed by transactions waiting to be processed.

4. Are any messages enqueued on SMB?

   There should be no messages enqueued on the SMB.

   SMB+X'1A' (SMBENQCT) minus SMB+X'18' (SMBDEQCT) = number of messages enqueued

   – If there are messages enqueued on the SMB, go to step 6.

   – If no messages are enqueued, go to step 5.

5. Are all regions accounted for?

   Compare the count of regions enqueued on the subqueues with the count in SCDREGCT (SCD + X'BCE') (2 bytes).

   - If the counts are equal, all regions are accounted for, and the IMS regions are in a normal scheduling environment. The problem is not with scheduling.
   - If not equal, other regions are active in IMS. Go to “PST Active” on page 36.

6. Report the problem.

   The problem is that IMS messages are enqueued on the SMB and wait-for-input (subqueue 6) is not posted. Report the problem to the IBM Support Center.

Loop

Use standard MVS system diagnostic procedures for loops.

Using the RB found in step 6c on page 30, determine the PSW address. The PSW address is labeled OPSW. The PSW address is always the second word following the label. This PSW address belongs to one of the modules involved in the loop.

You can use the MVS system trace to examine entries for the ASID and TCB indicated in the Dispatch Work Area at step 6 on page 29. The PSW address in the system trace entries indicates the modules involved in the loop.

Locate the above PSW addresses in the storage section of the dump and scan backward through the eye-catchers on the right side of the dump until you find a module identifier. To relate a module ID to a module name, see Appendix D, "Save-Area-ID-to-Module Cross-Reference Table," on page 521 for a list of cross-references between Save Area IDs and module names.

The looping module might not be an IMS module. Sometimes, the addresses are in the Link Pack Area (LPA) or the nucleus and might require an LPA or nucleus map.

Create the Search Argument

Keyword: LOOP

At this point, you can be sure that you are in a loop situation. Therefore, LOOP is an appropriate keyword for the search argument.

Keyword: Module Names Involved in the Loop

The module names derived in the loop procedure above are also valid keywords.

Keyword: Label in Module

If it is a tight loop, labels from the assembly listing of the modules involved might be useful keywords.

Keyword: Additional Related Keywords

External events can trigger loops. These events might be indicated by console messages or be related to a procedure that was being performed at the time the LOOP began.

You can use these additional keywords in the search argument to narrow the search, but they might not be necessary.

Search Argument Example

Consider the scenario:

- IMS went into a loop.
- The active modules indicated in the RB chain and the MVS System Trace Table were DFSCFEI0 and DFSCFE00.
- The loop began after the operator issued a /DISPLAY NODE command.

The broad search argument to use is:

```
5655B0100 LOOP DFSCFE00 | DFSCFEI0 | DISPLAY | NODE
```

For a structured database search, use this search argument:

```
PIDS/5655B0100 LOOP RIDS/DFSCFE00 | RIDS/DFSCFEI0 | PCSS/DIS | PCSS/NODE
```

With this search argument you might receive numerous hits, which will probably contain the APAR describing your problem. You can then take various combinations of the additional keywords that were compared with the **OR** operator in the above example and use the **AND** operator on them instead. You can use this technique to narrow the field of search until you find the appropriate APAR.

If the loop was not in an IMS module, do not use the IMS component ID, 5655B0100.

System Wait

Use standard MVS systems diagnostic procedures.

If the PSW address is for a system module, include that information when reporting the problem. You can use the module name in your search along with the WAIT keyword.

Shutdown Processing

Use this analysis if the operator issued a /CHECKPOINT FREEZE, DUMPQ, or PURGE to IMS and IMS failed to come down normally. Before taking IMS out of the system, be sure to use a /DISPLAY SHUTDOWN STATUS command. Obtain the listing of the /DISPLAY command and any subsequent activity to find any unusual conditions that might have prevented an orderly termination of IMS.

You should also use this analysis if IMS shut itself down and failed to terminate normally. For example, when IMS runs low on message queue space, it shuts itself down.

Before starting this procedure, you need to obtain an IMS dump in order to examine bit settings. Be aware that if you received only the first part of the DFS994I message during shutdown processing, VTAM might be involved in the failure. (For a DBCTL environment, ignore any further instructions that refer to VTAM in this section and in the next section, "Shutdown Analysis (CHE FREEZE, DUMPQ, or PURGE).") If you received the DFS994I xxx (FREEZE, DUMPQ, PURGE), but not DFS994I IMS SHUTDOWN COMPLETED, be sure to obtain a dump of VTAM and IMS. Here are two ways to get a dump:

- Enter the MVS DUMP command to dump the VTAM address space and then modify IMS down with a dump.
- Enter the MVS DUMP command to dump the VTAM, IMS control, DL/I, and CCTL address spaces, and then modify IMS down without a dump.

  Be sure to include the RGN option along with the other standard SDATA defaults in the DUMP command.

In the "Shutdown Analysis" that follows, note the following:

- Displacements and test conditions can change when maintenance is applied to a system.
- The bit settings shown are cumulative. This means that they usually combine with any bits already set in the byte. Check the bit settings as described. If a bit was not set or reset as shown, include both the module name and the cumulative bit settings in each byte in your search argument.
- SET turns the bit ON. RESET turns the bit OFF. Other bits in the byte might already be ON.
- It is essential in using the following analysis to find out if the indicated bits were SET or RESET and to use only the DUMPQ/FREEZE or PURGE sections where applicable.
- The Save Areas (SAs) might not always identify the last module to have control. In some cases, control is passed back to the initiating module (such as DFSCST00), and you can find no trace of any lower modules in the SAs.
- The main control block in shutdown problem analysis is the system contents directory (SCD). This flow of control lists most of the modules involved. When you find a field that does not have the bits SET or RESET as indicated, stop the analysis and report the problem.
- Be aware that defective code can produce results that appear to contradict this book.
- The following analysis does not list every action that is taking place in IMS shutdown processing, but only activity that causes bit setting to be changed in key SCD fields.
- Comments scattered throughout the analysis are for information only. For example, the statement, "If input or output is pending, return to DFSICIO0 with RC=C to complete", is for information. Do not look at return codes, but examine only the bit settings.

Shutdown Analysis (CHE FREEZE, DUMPQ, or PURGE)

Remember that in this analysis you'll be looking at bit settings, not hexadecimal values.

These sections do not apply to DBCTL shutdown:

PURGE
DFSICL20
DFSICLX0

DFSICIO0
DFSIPCP0
DFSCPCP0

DFSICL20

If PURGE, then
Set SCDCKCTL(X'B5C') = X'34' and then Set SCDSTOP1(X'B5E') = X'80'

If not PURGE, then

If DUMPQ,
Set SCDCKCTL(X'B5C') = X'1C'

If FREEZE,
Set SCDCKCTL(X'B5C') = X'14'
Reset POLL the lines and then (not applicable to DBCTL)
Set SCDSTOP1(X'B5E') = X'C0' (for DBCTL, set AWE to TRM1)

DFSICLX0

DFSICIO0

DFSIPCP0
If SCDCFLG1(X'A17') = X'08', then
Set SCDCQFLG(X'A18') = X'04' and
Set SCDCNXW4(X'A1F') = X'40'
If input or output is pending, return to DFSICIO0 with RC=C to complete.
When there is no input or output pending, or when the input or output is finished, then:
Set SCDCPCTL(X'A14') = X'80'
Set AWE to TRM1

DFSCST00

DFSTRM00

For PURGE
AWE = TRM1, First phase of termination
If SCDIDCNT+1(X'B0C') is not equal to X'000000'
and SCDCKCTL(X'B5C') = X'20' (PURGE)
Set SCDSTOP1(X'B5E') = X'10'
Set SCDSTOP1(X'B5E') = X'02'
If SCDFTFLG(X'25C') = X'20' (Fast Path active)
DBFTERM0 posts the Fast Path regions for SHUTDOWN

DFSTRM00

For DUMPQ or FREEZE
If SCDIDCNT+1(X'B0C') is not equal to X'000000'
and SCDCKCTL(X'B5C') is not equal to X'20' (Not PURGE)
Set SCDSTOP1(X'B5E') = X'04'
Set SCDSTOP1(X'B5E') = X'02'
If SCDFTFLG(X'25C') = X'20' (Fast Path Active)
DBFTERM0 posts the Fast Path regions for SHUTDOWN

For DUMPQ, PURGE, or FREEZE

If Fast Path was active on return from DBFTERM0, *or* if Fast Path was not active, then

If SCDREGCT(X'BCE') is not equal to X'0000' (ACTIVE REGIONS)

then

Post the PSTs waiting in the scheduler.

If SCDSHFL1(X'354') = X'80' (IRLM in system) and/or SCDIDCNT+1(X'B0C') is not equal to X'000000' then return to DFSCST00 to wait for regions to end, If DBCTL, notify DRA before returning to DFSCST00.

When OR If SCDIDCNT+1(X'B0C') = X'000000' (REGIONS ENDED)

Set SCDSTOP1(X'B5E') = X'01'

For PURGE only

If SCDCKCTL(X'B5C') = X'20' (PURGE)

Set SCDSTOP1(X'B5E') = X'20'

IWAIT for all output to go.

For DUMPQ, PURGE, or FREEZE

When all output is done for PURGE or FREEZE or DUMPQ, then

If SCDFTFLG(X'25C') = X'20' (Fast Path active)

DBFTERM1 closes the areas.

If SCDFTFLG(X'25C') is not equal to X'20' or when Fast Path areas are closed then

If SCDSMMS1(X'033') = X'02' (DL/I SAS)

Tell the DL/I region to close the databases (DFSSDL40).

IWAIT for the databases to close.

If not DLI/SAS, then let DFSDLOC0 close the databases.

Then when all databases and areas are closed

Set SCDSTOP1+1(X'B5E') = X'04'

<u>DFSCPCP0</u>

Set return code (RC) = 8 to ask DFSIPCP0 if communication is still going on.

<u>DFSIPCP0 (DFSIPCP2)</u>

If no output or no messages on Q3,

Set return code (RC) = 0 to inform DFSCPCP0

If output or messages on Q3,

Set return code (RC) = 4 to inform DFSCPCP0, which causes DFSCPCP0 to IWAIT

<u>DFSCPCP0</u>

If output is pending (RC = 4)

Set SCDCPCTL(X'A14') = X'08'

Set SCDSTOP1(X'B5E') = X'40'

IWAIT for DC to finish.

If no output or when output finishes

Set off SCDCPCTL(X'A14') = X'08' (reset the bit)

Set SCDSTOP1+1(X'B5E') = X'08'

Reset Poll all lines that are candidates for the SHUTDOWN message

Set CTBFLAG3(0D) = X'10' (for all terminals that are to receive the shutdown message)

DFSICLX0

DFSICIO0

DFSIPCP0

If any CTBFLAG3(0D) = X'10'

Set CTBACTL(10) = X'20'

Set CTBACTL(10) = X'10'

RC = 8 to DFSICIO0 (send SHUTDOWN message)

If NO CTBFLAG3(0D) = X'10'

Set SCDDFLGS(X'698') = X'80'

Set SCDCPCTL(X'A14') = X'20'

RC = 4 to DFSICIO0 (quiesce lines)

DFSICIO0

If RC = 4, idle the lines

If RC = 8, send DFS991 - IMS SHUTDOWN message

The WRITE interrupt from the SHUTDOWN message results in the following:

Set off CTBFLAG5(0F) = X'80' (reset)

Set off CTBFLAG3(0D) = X'10' (the)

Set off CTBACTL (10) = X'30' (bits)

DFSIPCP0

When all line activity is stopped

DFSCPCP0

DFSTRM00

If DBCTL set SCDSTOP =SCDSTSNT

| Set SCDSTOP1+1(X'B5E') = X'01'

DFSRCRT0

DFSRCP00

Send “DFS994I *CHKPT yyddd/hhmmss*ctype” (first part of DFS994I message)

Set AWE = “TRM2”

| Set off SCDCKCTL(X'B5C') = X'04' (reset the bit)

DFSTRM00

Set SCDTRMFL(X'3B8') = X'40'

DFSCST00

DFSTRM00

If DLI/SAS SCDSMMS1(X'033') = X'02'

Pass AWE to DFSSDL40 to begin Normal Termination

If not DLI/SAS or when DFSSDL40 returns

If SCDRFPIN(X'B76') = X'80' (Fast Path errors)

Print error message
Set off SCDRFPIN(X'B76') = X'80' (reset the bit)
Close queue data sets (not applicable to DBCTL)
IWAIT for closing
| Set off SCDSTOP1(X'B5E') = X'08' (reset the bit)

DFSTERM0
Terminate DASD log
Set off SCDRECTL(X'136') = X'80' (reset the bit)
Terminate RDS
Terminate IMS system type tasks
Signoff DBRC
Quit IRLM
Close VTAM ACB (not applicable to DBCTL)

If DLI/SAS, SCDSMMS1(X'033') = X'02'
and the ECB at SCDRSETF(X'C50') is not equal to X'40' (posted)
IWAIT for the DL/I region to end
Then set AWE = "TRM3"
Set SCDTRMFL(X'3B8') = X'20'
Send "DFS994I IMS SHUTDOWN COMPLETED" (second part of DFS994I message)

DFSTRM00

DFSCST00
Back to the SCP (all done)

IRLM Procedure

WAIT states can be encountered during IRLM processing in four areas.

Deadlock Involving Non-IRLM Resources:

Failure Description: Application programs waiting for non-IRLM resources and holding IRLM resources are waiting for other applications also holding IRLM resources. The IRLM cannot detect deadlocks involving non-IRLM resources.

Detection: Use the IMS WAIT diagnostic procedures to discover the non-IRLM resources being waited for. Follow the RLB chains representing resources held or requested for each requesting work unit (WHB) to discover the IRLM resources being waited for. If the wait state occurred as a result of an IRLM error, the function/subfunction is IRLM/DEADLK.

An example of a search argument is:

```
569516401 AR101 WAIT IRLM IRLM/DEADLK
```

| For a structured database search, use this search argument:

```
|   PIDS/569516401 LVLS/101 WAIT RIDS/IRLM RIDS/DEADLK
```

Deadlock Involving Only IRLM Resources:

Failure Description: Application programs are deadlocked for IRLM resources. If all the application programs are waiting for IRLM resources (there are no application programs running which could release the locks that the other application programs are waiting for), this is a deadlock. The IRLM should detect this condition and post one of the waiters as unable to obtain the lock because of a deadlock.

Detection: Follow the RLB chains representing resources held or requested for each requesting work unit (WHB) to discover the IRLM resources being waited for. If the wait state occurred as a result of an IRLM error, the function/subfunction is IRLM/DEADLK.

An example of a search argument is:

```
569516401 AR101 WAIT IRLM IRLM/DEADLK
```

| For structured database search, use this search argument:

```
PIDS/569516401 LVLS/101 WAIT RIDS/IRLM RIDS/DEADLK
```

Lock Request Not Granted Because Holder Did Not Release Lock:

Failure Description: An application program requested a lock, but the request was not granted because the holder of the resource did not release it. This does not result in a deadlock. However, If the requester is not timed out, its task and any others waiting after it might enter a wait state.

An example of a search argument is:

```
569516401 AR101 WAIT IRLM
```

| For structured database search, use this search argument:

```
PIDS/569516401 LVLS/101 WAIT RIDS/IRLM
```

IRLM Latch Unavailable:

Failure Description: An error in IRLM processing can result in an IRLM latch being permanently unavailable. If this condition exists, no new IRLM requests can be processed.

If this error occurs, call the IBM Support Center for help in diagnosing the problem. The support representative will tell you what type of documentation to gather.

Chapter 5. Procedures and Techniques

This chapter details procedures and techniques for the following:

- Searching the IBM Software Support Facility (SSF) to find out whether a problem like yours is already known to IBM.
- Preparing an APAR
- Searching RETAIN® for APARs closed within a specific time period

Searching the Database

You have completed your search argument. You now want to know whether a problem like yours has already been reported to IBM. To find out, you can use your newly developed keyword string in searching an IBM software support database, such as SSF (Software Support Facility), provided you have the necessary access. Or you can use it when talking to your Level 1 support representative.

1. Determine the maintenance level of the IMS system by identifying the APARs and/or PTFs that have been applied.
   - Run the SMP PTF list program or have access to online SMP/E dialogs.
2. Search SSF, using the keyword string developed by following procedures from Chapter 4, “Selecting the Keywords.” Your search will be most successful if you follow these guidelines:
   - Start with a broad search argument so you receive all problem descriptions that might match your problem.
   - If you find too many APARs to examine, add the logical operators **AND** or **OR** to the keyword string in various combinations gradually to reduce the number of database matches (hits). If the keywords are connected by the logical operator **AND** (a blank), a record is selected if it contains both words separated by the blank. If the keywords are connected by the logical operator **OR** (**|**), a record is selected if it contains either of the words separated by the character, **|**.
   - You can use dependency keywords with the keyword string to select only those APARs that apply to a certain environment. These can be particularly useful when a search yields a large number of database matches and you are almost certain that the program failure occurred in a specific environment. For the list of dependency keywords, see Appendix E, “Dependency Keywords,” on page 545.

     Recommendation: Use dependency keywords only if you are sure the problem is limited to that dependency. If you do not get any database matches, eliminate the dependency keyword.
   - If you want to narrow the search to a specific release level, you can add the logical operators **AND** or **OR** for the release level keywords to the search argument. For IMS Version 7 these are:

     AR700 for IMS Services

     AR701 for Database Manager

     AR702 for Transaction Manager

     AR703 for ETO

     AR704 for Remote Level Tracker

     AR705 for Database Level Tracker

     AR706 for database recovery service

     AR707 for IMS Connect

     AR101 for Internal Resource Lock Manager 2.1

     For a structured database search, the release level keywords are:

     LVLS/700 for IMS Services

LVLS/701 for Database Manager

LVLS/702 for Transaction Manager

LVLS/703 for ETO

LVLS/704 for Recovery-level Tracking

LVLS/705 for Database-level Tracking

LVLS/706 for database recovery service

LVLS/707 for IMS Connect

LVLS/101 for Internal Resource Lock Manager 2.1

An example is:

5655B0100 AR701 for the Database Manager

For a structured database search, an example is:

PIDS/5655B0100 LVLS/701

Recommendation: If you do not get any database matches, remove the release level from your search argument.

3. Eliminate the APARs that also appear in the SMP PTF list from the list of database matches. These will have already been applied.
4. Compare each remaining APAR with the current failure symptoms. Analyze trace output for your problem situation, looking for similarities in the situations described by APARs you're reviewing. Frequently APAR descriptions include some information about the traces that were run for those problems.
5. If you find an appropriate APAR, see if it has been closed. If it has been closed, you can correct the problem by applying the fix associated with the APAR. If it has not been closed, contact your IBM Support Center for instructions on what you can do until it is closed.
6. If you do not find an appropriate APAR, verify that the problem is not caused by a user specification error.
7. If you find no user specification error, contact the IBM Support Center for assistance.

Searching for APARs Closed within a Specific Time Period

You can search RETAIN for high-impact pervasive (HIPER) or performance APARs that were closed within a specific time period. For example, to search for HIPER APARs closed between 10/97 and 04/99, use this search argument:

```
P;CL97/10-99/4. HIPER
```

If you want to search only for HIPER APARs for a specific release, add the component ID to the search argument. For example, to search only for IMS Version 6 APARs, use this search argument:

```
P;CL97/10-99/4. HIPER  5655B0100
```

For a structured database search, use this search argument:

```
P;CL97/10-99/4. HIPER  PIDS/5655B0100
```

If you want to search only for HIPER APARs for a specific release, add the component ID to the search argument. For example, to search only for IMS Version 7 APARs, use this search argument:

```
P;CL97/10-99/4. HIPER  5655B0100
```

For a structured database search, use this search argument:

```
P;CL97/10-99/4. HIPER  PIDS/5655B0100
```

Preparing an APAR

An APAR (Authorized Program Analysis Report) might be necessary if the keyword search proves unsuccessful. Call the IBM Support Center for help in determining if an APAR is necessary. Only authorized IBM personnel can generate APARs.

Table 4. Preparing an APAR

Procedure	What to Do
Reporting a problem	To report a problem, contact your IBM Support Center. Be prepared to supply such information as: • Customer number • Release level • Current maintenance level (from PTF list) • The keyword string or strings used to search the IBM software support database
Gathering APAR documentation	You might be asked to supply various types of information that describe the IMS nucleus, database, environment, or activities. Include applicable items from the following list with the APAR. • JCL listings • Address space storage dumps at time of failure—the entire machine-readable dump data set (normally copied to tape) and the JCL used to copy the dump to tape • Link-edit map • MVS console printout. A partial console is generally in the offline formatted dump. • Master terminal printout • Local/remote terminal printout • IMS log data sets • IMSGEN listing • DBD listing • PSB listing • ACB generation output • Log trace • Consolidated trace output • Transmittal notes explaining any unusual events leading up to the problem symptoms • SNAPs produced before and after the failing call by DFSDDLT0 • Type X'67FF' SNAP log records • Type X'6705' SNAP log records • DBRC—RECON data set • LPA map • LOGREC (especially software diagnostic records)

Submitting APAR documentation	When submitting material for an APAR to IBM, carefully pack and clearly label all materials sent to IBM with the following information:
	1. The APAR number assigned by IBM
	2. A list of data sets on the tape, including JCL, if any
	3. A description of how the tape was made, including:
	  • The exact JCL listing or the list of commands used
	  • The recording mode and density
	  • Tape labeling
	  • The record format and block size used for each data set

Part 2. Data Areas and Record Formats

Chapter 6. Data Areas and Record Formats

This chapter describes the major IMS control blocks and their interrelationships. It also describes the formats of records that you need to analyze when diagnosing problems. This chapter includes:

- Introduction to the data areas and record formats
- A table of control block definitions
- Diagrams of control block interrelationship
- The format of an edited command
- DL/I record formats

Getting More Information on Modules, Control Blocks, and Record Formats

You can find the module directory, IMS control block DSECTs, and the log record formats on Service Link. Contact your systems engineer for further information on accessing Service Link.

The IMS.ACBLIB is a partitioned data set whose members are pre-system-generated, expanded PSB and DMB control blocks. You can view the formats of these control blocks by assembling the database DSECT and CSECT control blocks macro IDLI. You can also find the layout of IMS.ACBLIB members in the ACBGEN module, DFSUACBO, and the Write-PSBs-and-DMBs-to-ACBLIB module, DFSUAMBO.

Figure 1 on page 54 gives an overview of the linkage of the major control blocks used for diagnosis.

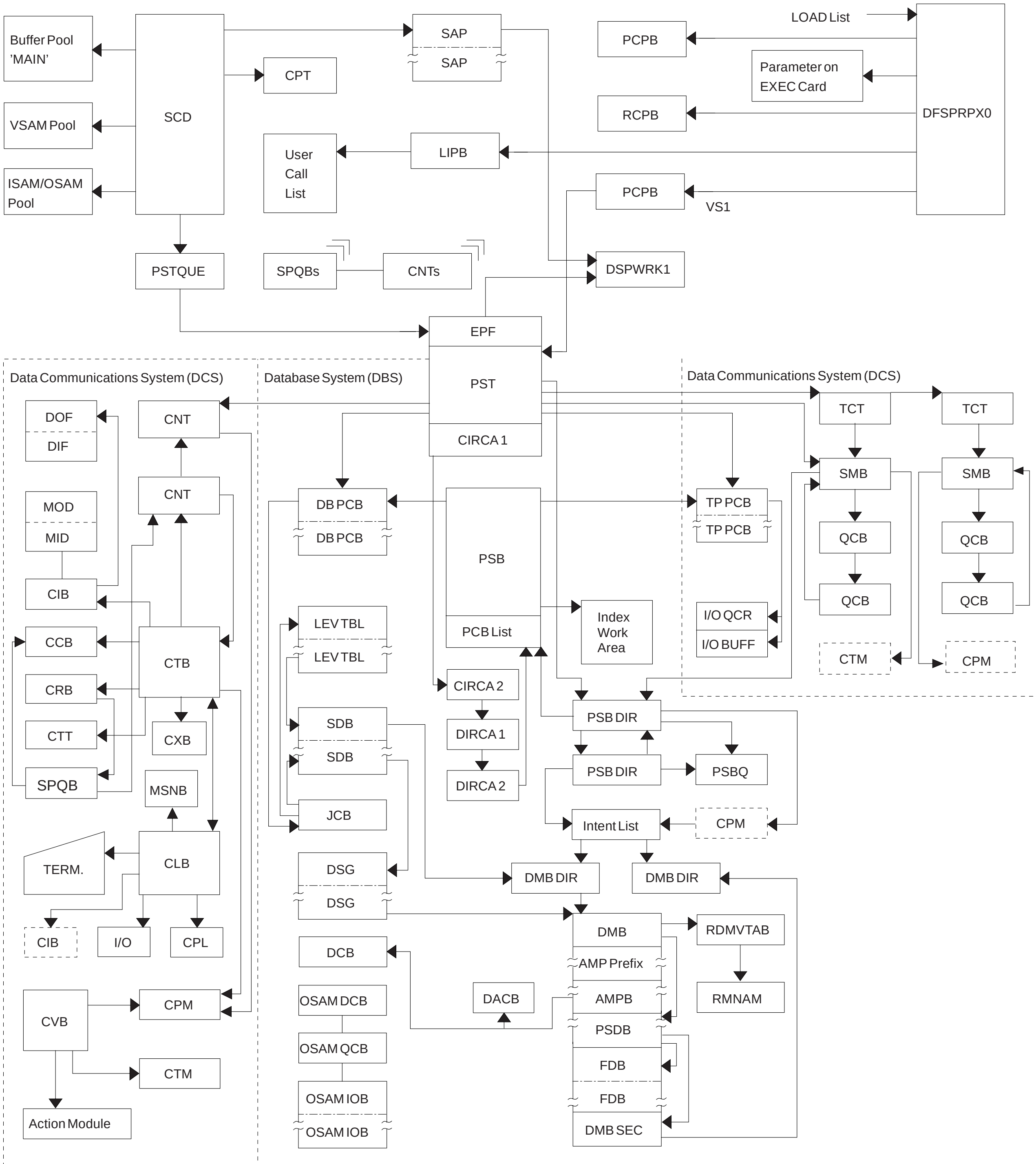


Figure 1. IMS Control Block Linkage for a Static DB/DC Environment

Table of Control Block Definitions

Table 5 lists:
- The acronyms of the control blocks described in this manual
- The macro that generates the block
- A brief description of the block

Table 5. Table of Control Block Definitions

Control Block Acronym	Mapping Macro	Description
ADSC	DBFADSC	Area data set control block.
ALDS	DBFAREA	Area list data set.
AMPB	IDLI DMBBASE=0	Access method prefix block. Contains information relative to a data set belonging to a database.
BALG	DBFBALG	Balancing group control block.
BFSP	IDLIVSAM BFSP	DL/I VSAM buffer handler pool prefix.
BFUS	IDLIVSAM BFUS	Subpool statistics block.
BHDR	BHDR	MSDB header.
BLOCKHDR	DFSSPBLK	Block header used by DFSPOOL Storage Manager.
BSPH	IDLIVSAM BSPH	Buffer subpool header block. Contains the number of buffers in this subpool.
BUFC	IDLIVSAM BUFC	Buffer control block. Contains pointers to actual buffers.
BUFENTRY	DFSSPBLK	Used by DFSPOOL Storage Manager to map the buffer size entries within the pool header.
CADSECT	ICADSECT	Communication area block. Contains the main dump formatter control block.
CBT	DFSCBTS	Control block. Represents storage pools (IPAGES) defined in DFSCBT00.
CCB	ICLI CCBBASE=0	Conversational control block. Controls resources for conversational tasks.
CIB	ICLI CIBBASE=0	Communication interface block. Contains information the DDM needs to determine Message Format Service (MFS) operation.
CIRCA	IPST	IMS control region interregion communication area.
CLB	ICLI CLBBASE=0	Communication line block. One exists for each communication line and for each node.
CLLE	DFSCLLE	Common Latch List Element. There is one block for each IMS ITASK, which is maintained in Key 7 storage.
CNT	ICLI CNTBASE=0	Communication name table. One exists for each named logical terminal and component.
CPM	(generated)	Communication password matrix. Length varies based upon the number of passwords in the CPT.
CPT	(generated)	Communication password table. Defined by user.
CRB	ICLI CRBBASE=0	Communication restart block.
CSAB	OCO	Callable Service Anchor Block. Used by IMS callable services modules.
CSVT	DFSCSVT	Callable Services Vector Table. Used by IMS callable services modules.
CTB	ICLI CTBBASE=0	Communication terminal block. One exists for each terminal and for each subpool in the system.
CTM	(generated)	Communication terminal matrix. Length varies based upon the number of logical terminals (CNTs).

Table 5. Table of Control Block Definitions (continued)

Control Block Acronym	Mapping Macro	Description
CTT	ICLI CTTBASE=0	Communication terminal table. There is one for each different type of terminal, as well as different features.
CULE	DFSCULE	Common Use List Element. Used in latching by the IMS Use Manager.
CVB	ICLI CVBBASE=0	Communication verb block. Reflects the relationship between the command message verbs and the passwords. It also reflects logical terminals associated with those commands.
CXB	(generated)	Communication extension block. Contains information that is required for control of a particular terminal. It is a logical extension of the CTB.
DBPCB	IDLI DPCBASE=0	DL/I DB PCB.
DCB	IDCBOS	Data communication block. Contains data pertinent to the current use of a data set.
DCB-EXT	DFSDCBEX	OSAM extension to the DCB.
DCBOSAM	IDCBOSD	OSAM DCB.
DDIR	IDLI DDRBASE=0	DMB directory entry. Contains an entry for each DMB known to IMS.
DFSAVEC	DFSAVECT	Dump formatter vector table.
DFSDOPTE	DFSDOPTB	Dump option entry block. Is the dump formatter CBTE request definition block.
DFSDPBFH	DFSDBPFH	Dump buffer pool blocks. Used for buffering offline dump storage.
DFSSBWO	DFSSBWA	Work area used by sequential buffering.
| DMAC	DBFDMAC	DEDB area control block.
DMB	IDLI DMBBASE=0	Data management block. There is one for each database descriptor entry described in the DDIR.
DMBSEC	IDLI DMBBASE=0	Secondary list. There is one or more entry for each logically related segment and each index relationship.
DMCB	DBFDMCB	DEDB master control block.
DMHR	DBFDMHR	The buffer header for Fast Path. Describes the status of a particular buffer. The buffer headers (and buffers) are allocated in DBFCONT0. ESCDDMHR points to the first buffer and ESCDMBFN contains the number of headers. The relationship between buffer headers and buffers is fixed during IMS control region initialization.
DSEB	DFSDSPDS	Dynamic SAP Extension Block. Used to manage dynamic SAPs.
DSG	IDLI JCBBASE=0	Data set group control block. There is typically one for each data set group referenced by the DBPCB.
DSPWRK1	IDSPWRK	Dispatcher work area. There is one for each VS task (TCB) in an IMS environment.
ECB	MVS macro	Event control block. Describes the status of an event in an IMS environment.
ECNT	DBFECNT	Extended communications name table. (Fast Path)
EDSG	DFSSBDSG	Sequential buffering extension to the DSG.
EMHB	DBFEMHB	Expedited message handler block. (Fast Path)
EIB	DFSPCA	Partition Exit Interface Block Prefix.
EPCB	DBFEPCB	Extended PCB. (Fast Path)
EPF	IEPF	ECB prefix. Used to indicate the current status of the ECB and to connect the ECB to the appropriate SAP.

Table 5. Table of Control Block Definitions (continued)

Control Block Acronym	Mapping Macro	Description
EPST	DBFEPST	Extended partition specification table. (Fast Path)
EQEL	DFSEQEL	Recoverable in-doubt structure queue elements. Identifies inaccessible data due to in-doubt status.
ESCD	DBFESCD	Extended system contents directory. (Fast Path)
ESRB	DBFESRB	Extended service request block. (Fast Path)
ESRT	DBFESRT	Expedited message handling region insert buffer. This buffer is a temporary save area for a message input. ESRTs are allocated in module DBFCONT0 by IMS control region initialization with a length equal to the largest terminal buffer defined. ESCDESRT points to the first ESRT. EPSTESRT points to a related ESRT. (Fast Path)
FAQE	DFSSPBLK	Free allocated queue element. Used by the DFSISMN0 Storage Manager to manage storage within a pool.
FDB	IDLI FDBBASE=0	Field descriptor block.
FDT	DBFMFDB	Field description table.
FEDB	ICLI FEDBBASE=0	Front end directory block. Stores global information about the front end switching facility.
FEIB	ICLI FEIBBASE=0	Front end interface block. Contains data to allow the front end switching user exit to communicate with the transaction manager.
FRB	DFSFRB	Fast restart block.
GB	IGLI	GSAM data set control block. Contains information concerning the data set operation and pointers to other control blocks used for accessing records.
GBCB	IGLI	GSAM buffer control block. Contains the address of a unique buffer.
GLT	IGLI	GSAM load table. Provides all addresses of the GSAM load modules necessary for initialization.
GPT	IGLI	GSAM pointer table. Provides information required by resident and nonresident GSAM routines.
GQCB	IGLI	GSAM queues control block. Contains first and last pointers for the four queues of GSAM GBCBs used by GSAM BUFFIO.
HSSR	DBFHSSR	Holds area range information from SETR statements. HSSR is formatted in the offline dump.
HSSO	DBFHSSO	Holds image copy (IC) information from SETO statements.
HSSD	DBFHSSD	Holds information for the /DISPLAY HSSP command. HSSD is formatted in the offline dump.
HSSP	DBFHSSPS	Skeleton block. Temporarily holds HSSO/HSSR/HSSD information before scheduling.
IBFPRF	IBFPRF	Buffer prefix. There is one for each buffer described in each subpool used by the OSAM buffer manager.
IBPOOL	IBPOOL	OSAM buffer handler main buffer pool. Contains statistics and vectors to OSAM buffer subpools.

Table 5. Table of Control Block Definitions (continued)

Control Block Acronym	Mapping Macro	Description
IDSC	DBFIDSC	IDSC is the image copy data set control block. It represents the Image Copy data set (IDS) the same way the area data set control block (ADSC) represents the area data set (ADS). IDSC also uses the same control block structure as the ADSC. An IDSC contains a description of the Image Copy data set. There are up to two IDSCs for each DEDB area with the Image Copy option. An IDSC is built dynamically at the first call to the area that is running as HSSP with the Image Copy option requested. The ISDC is released during Image Copy termination. The IDSC control block is formatted in the offline dump.
IEEQE	DFSIEQE	In-doubt error queue element. Contains buffers of changed data (data in the in-doubt state).
ISPL	ISUBPL	OSAM buffer subpool. Provides a base for fixed length buffers and statistics about the buffers.
ISL	DXRRLISL	IRLM identified subsystem list. Contains the name of each subsystem and its status.
JCB	IDLI JCBBASE=0	Job control block. There is one for each PCB. It contains level tables and segment blocks and a trace table of the previous calls.
LCB	LCB	Link control block. Represents the link for channel to channel, memory to memory, VTAM, and binary synchronous connections in MSC.
LCD	LCDSECT	Log contents directory. Controls the interface between the logical and physical loggers in a DB/DC environment.
LCRE	DFSLCRE	Local current recovery element. Contains the sync point, checkpoint recovery information relative to each PST.
LEV	IDLI JCBBASE=0	Level table. Consists of two parts: previous call and current call that is filled in by the call analyzer.
LIPB	IDLI PSTBASE=0	Language interface parameter block.
LLB	ICLI CLBBASE=0	Link line block.
LTB	ICLI CTBBASE=0	Link terminal block.
LXB	LXB	Link extension block.
MRMB	DBFMRMB	DEDB randomizing module block.
MSNB	MSNB	Message Control/Error exit interface block. Contains the block content before and after calling Message Control/Error exit DFSCMUX0 or during the interface processing.
PAC	DFSPAC	Database Resource Adapter (DRA) control block.
PAPL	DFSPAPL	DRA architected parameter list.
PARMLIST	ICADSECT	Dump formatter bulk print interface block.
PAT	DFSPAT	DRA thread control block.
PATE	DFSPAT	DRA thread entry control block.
PCA	DFSPCA	Partition Communication Area.
PCB	IDLI	Program communication block. There is one for each logical database being referenced by the application program.
PCIB	ICLI PCIBASE=0	Partition communication interface block.
PCPB	IDLI PSTBASE=0	Program control parameter block.
PCT	DFSPCT	Partition chaining table.

Table 5. Table of Control Block Definitions (continued)

Control Block Acronym	Mapping Macro	Description
PDA	DFSPSEIB	Partition Definition Area Prefix. Partition Definition Area Entry.
PDIR	IDLI PDRBASE=0	Program specification block directory. Contains entries for every program known to IMS.
PDL	DFSPDL	DRA dump parameter list.
PEC	DFSPSEIB	Partition Exit Communication Area.
PNT	DFSPNT	Partition Name Table.
POOLHDR	DFSSPBLK	Storage pool header used by the DFSPOOL storage manager to keep track of pool information.
PPRE	DFSPPRE	Standard IPAGE prefix mapping macro. Used for all IPAGEs created in IMS.
PQE	DFSPQE	DRA queuing element.
PSB	IDLI PSBBASE=0	Program specification block. Relates to the application program and contains the PCBs associated with this PSB.
PSDB	IDLI DMBBASE=0	Physical segment descriptor block. Describes each segment in the database.
PST	IPST	Partition specification table. There is one for each message or batch region; it contains a DECB for this partition, I/O terminal PCB, and parameters required for this region.
PTBWA	DXRPTBWA	IRLM pass-the-buck work area.
PTE	DFSPNT	Partition Table Entry.
PTK	DFSPTK	Partition Key Index Table.
PTX	DFSPTX	Partition Entry Index Table.
PXPARMS	PARMS	Region descriptor block.
QCB	IAPS SMBBASE=0	Queue control block.
QEL	IAPS SMBBASE=0	Queue Element.
QMBA	DFSQMGR	Queue Manager Buffer Area.
RCPB	IDLI PSTBASE=0	Region control parameter block.
RCTE	DBFRCTE	Routing code table entry.
RDLWA	DXRRDLWA	IRLM deadlock process work area. Contains information that must be communicated between the deadlock process modules.
RHB	DXRRHB	IRLM resource header block. Represents a resource.
RHT	DXRRHT	IRLM resource hash table. Provides a series of anchors for resource chains.
RLB	DXRRLB	IRLM resource lock block. Represents a request for a lock or a lock held on a resource.
RLCBT	DXRRLCBT	IRLM private area control block and table. Contains addresses of IRLM entry points.
RLMCB	DXRRLMCB	IRLM master control block. Contains branch entry addresses for all RLMREQ as well as queue anchors.
RLPL	DXRRLPL	IRLM request parameter list. This is the parameter list for all functional requests for the resource lock manager.
RLQD	DXRRLQD	IRLM query mapping macro. Maps IRLM control blocks/structures returned to the IMS invoker of QUERY.

Table 5. Table of Control Block Definitions (continued)

Control Block Acronym	Mapping Macro	Description
RPLI	IDLIVSAM	Request parameter list. Contains parameters passed to VSAM from IMS and the status returned to IMS from VSAM.
RPST	DFSRPST	Restart PST. Contains identifying information and characteristics of units of recovery.
RRE	DFSRRE	Residual recovery element. Contains sync point actions, such as Commit and Abort, relative to each Database 2™ (DB2) connection out of a dependent region and is used for BMP restart processing, in-doubt processing, and restartable backout processing.
SAP	ISAP	Save area prefix. Relates to a save area set.
SBHE	DFSSBHE	Sequential buffering hash entry. Used to hash or anchor SBCB control blocks and to serialize the sequential buffer SDCB and SDSG control block subsystem chains. The SBHEs are part of the SBSCD.
SBPARMS	DFSSBPAR	Sequential buffering extension to PXPARMS.
SBPSS	DFSSBPSS	Sequential buffering extension to the PST, which is located in CSA.
SBPST	DFSSBPST	Sequential buffering extension to the PST.
SBSCD	DFSSBSCD	Sequential buffering extension to the SCD. This extension contains the SBHE control blocks.
SBUF	IBFPRF SBEXT=YES	Sequential buffering buffer. One SBUF control block is used by sequential buffering to control each SB buffer. The SBUF control blocks of one SB buffer pool are contiguous in storage and are formatted as one entity.
SCAR	DFSSBCAR	Control block containing the interpreted data of one SBPARM control statement in the //DFSCTL file.
SCA1	DFSSBCAR	Control block containing the uninterpreted data of one SBPARM control statement in the //DFSCTL file.
SCD	ISCD	System contents directory. Produced at system generation time, it contains major entry points for all facilities and system control information.
SDB	IDLI SDBBASE=0	Segment descriptor block. Contains a logical description of the segment.
SDCB	DFSSBDCB	Sequential buffering extension to the DCB. Is for those DB data sets that are buffered by sequential buffering.
SDSG	DFSSBDSG	Sequential buffering extension to the DSG. Describes one I/O process. There is typically one SDSG control block for each data set group control block (DSG) that might potentially be buffered by sequential buffering.
SDWA	IHASDWA	System diagnostic work area.
SGT	DFSPRSGT	Segment table. Describes the segments used by the partial reorganization process. It is built during the DBD analysis phase. Its address is held in the common area field (COMASGT). The segment extension table (SGX) holds additional information about the segments.
SIDB	DXRSIDB	IRLM subsystem identification block. Used to identify each subsystem that relates to IRLM.
SIDX	DFSSSIE	Subsystem index entry.
SMB	IAPS	Scheduler message block. Related to a transaction.
SPQB	ICLI SPQBASE=0	Subpool queue block. The SPQB represents the dynamic user for an ETO terminal and represents a set of static queues (CNTs) for a static ISC parallel session terminal.
SQPST	ISQPST	PST queue. Associated with the scheduler sequence queue.

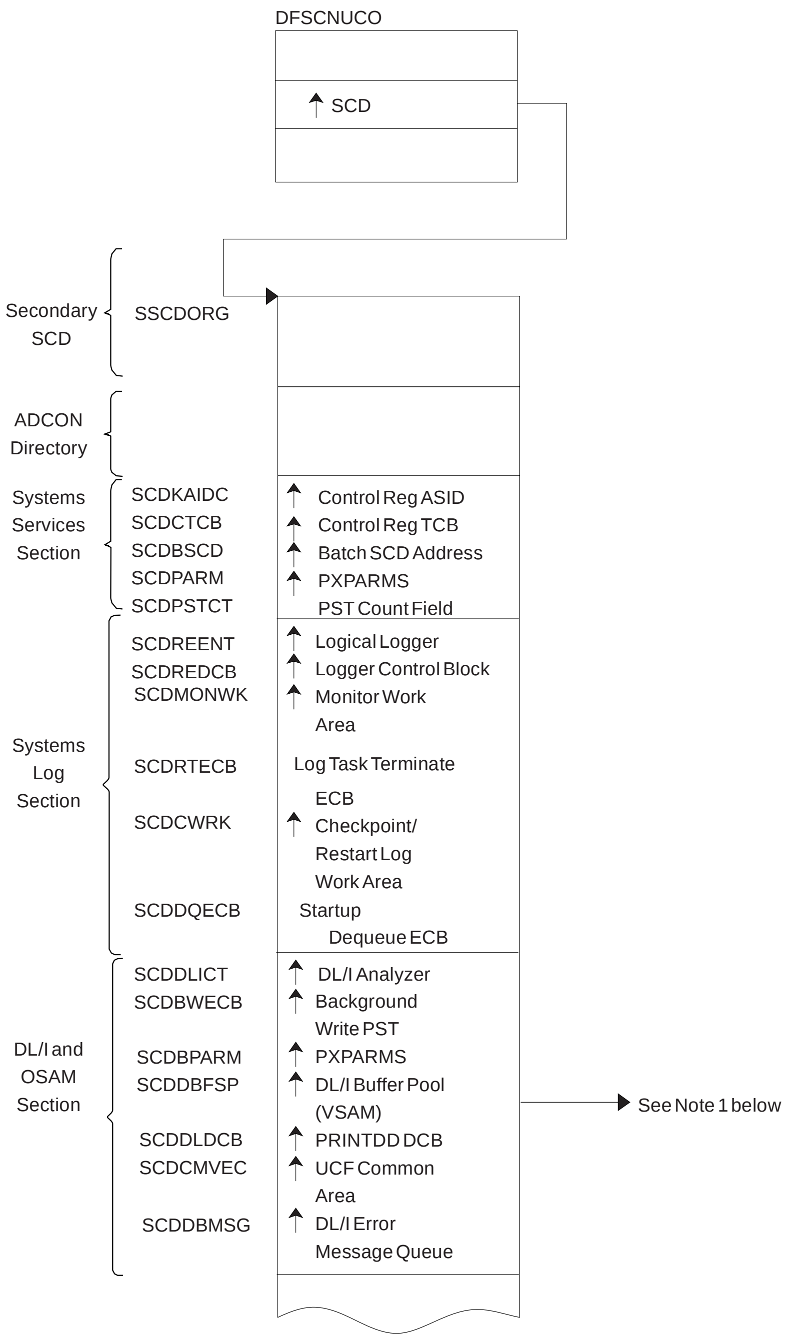
Table 5. Table of Control Block Definitions (continued)

Control Block Acronym	Mapping Macro	Description
SRAN	DFSSBRAN	Sequential range. Used in sequential buffering to describe a recently referenced set of consecutive DB blocks. Sequential buffering allocates one Sequential SRAN control block for each buffer set of each buffer pool. SB also allocates Random SRAN control blocks to each buffer pool. The Sequential SRANs and Random SRANs of one SB buffer pool are contiguous in storage and are formatted as one entity.
SSIB	IEFJSSIB	Subsystem identification block. Identifies the subsystem that requested services.
SSOB	IEFJSSOB	Subsystem options block. Used to request a particular function from the MVS subsystem.
SSVP	DFSSSVPL	System Services Parameter List. Used by IMS System Macros for parameter lists for mailing out of line calls. There is one SSVP per ITASK, anchored off of the SAP.
TAB	DFSTAB	Transaction anchor block.
TCT	DFSTAB	Transaction class table. Used for queuing of messages in a priority sequence within a specified class.
UEHB	UEHB	User exit header block. Used for automated operator exit interface processing.
UXDT	DFSUSRX	User Exit Definition Table. Contains control information and user exit addresses for user exits managed by IMS standard user exit service.
UXRB	DBRUXRB	A unit of work (UOW) is represented by a UOW exclusive resource control block (UXRB), similar to the XCRB representing the CI. The UXRB contains information about the UOW (for example, Area, RBA) and is used for resolving potential UOW resource contention among dependent regions. Other UXRB fields include the lock token, number of associated XCRBs, the owning EPST, the update intent flag, and the PCB. The UXRB control block is formatted in the offline dump.
VSI	IDLIVSAM VSI	VSAM sharing information control block. Controls VSAM sharing between subsystems.
WHB	DXRWHB	IRLM work unit block. Contains the anchor for all requests associated with that owner.
XCRB	DBFXCRB	Exclusive control resource block.
XMCA	DFSXMC	Cross-Memory Control-Address Spaces. There is one block for each IMS subsystem, which is maintained in Key 0 storage.
XMCI	DFSXMC	Cross Memory Control-ITASKs. There is one block for each IMS ITASK, which is maintained in Key 7 storage.
ZIB	IZIB	Zone initialization block. Used by the DFSISMN0 Storage Manager to keep track of a buffer obtained via ICREATE.

Control Block Interrelationship Diagrams

This section contains diagrams that show the interrelationships of major control blocks in an IMS environment. Descriptions of the figures in this section are listed below.

Figure	**Description**
2	Online system contents directory (SCD)
3	DFSPRPX0 parameter blocks



- Note 1: See Figure 4 on page 70.
- Note 2: See Figure 6 on page 72.

Figure 2. Online System Contents Directory (SCD) (Part 1 of 6)

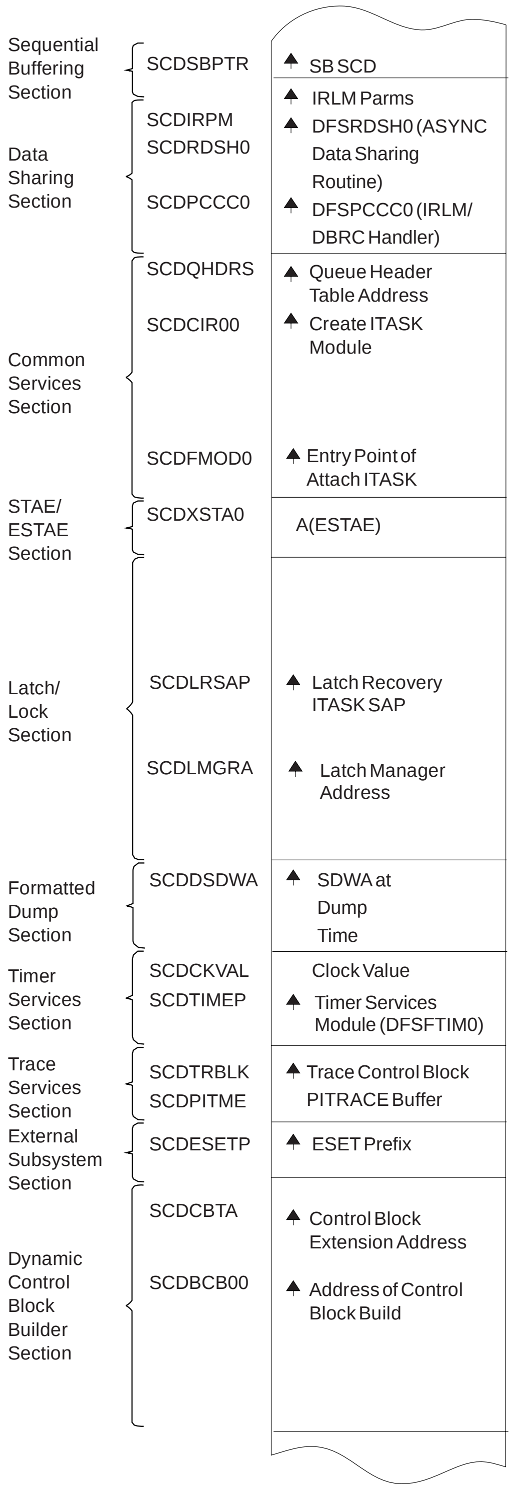
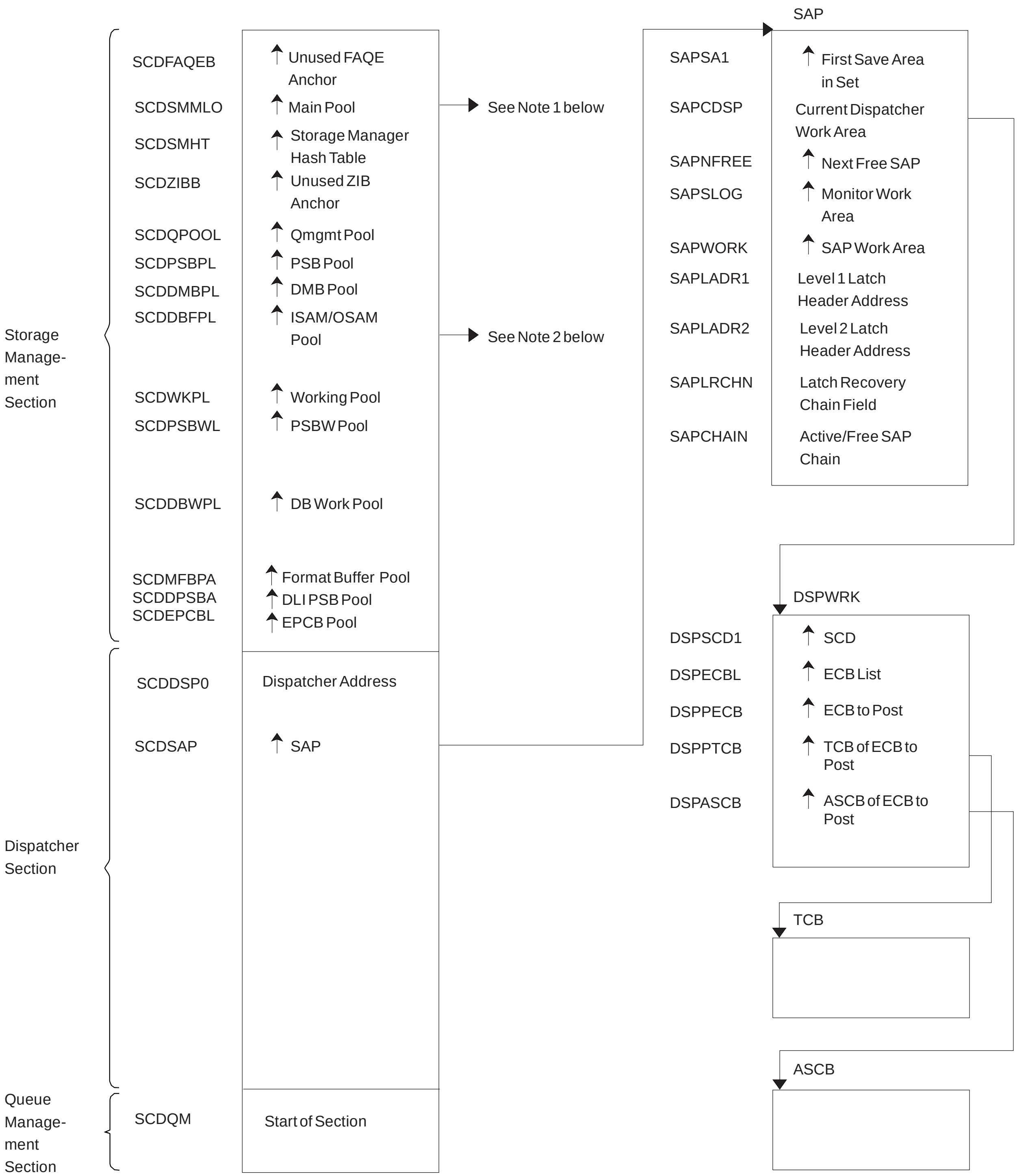


Figure 2. Online System Contents Directory (SCD) (Part 2 of 6)



Note 1: See Figure 9 on page 75.

Note 2: See Figure 4 on page 70.

Figure 2. Online System Contents Directory (SCD) (Part 3 of 6)

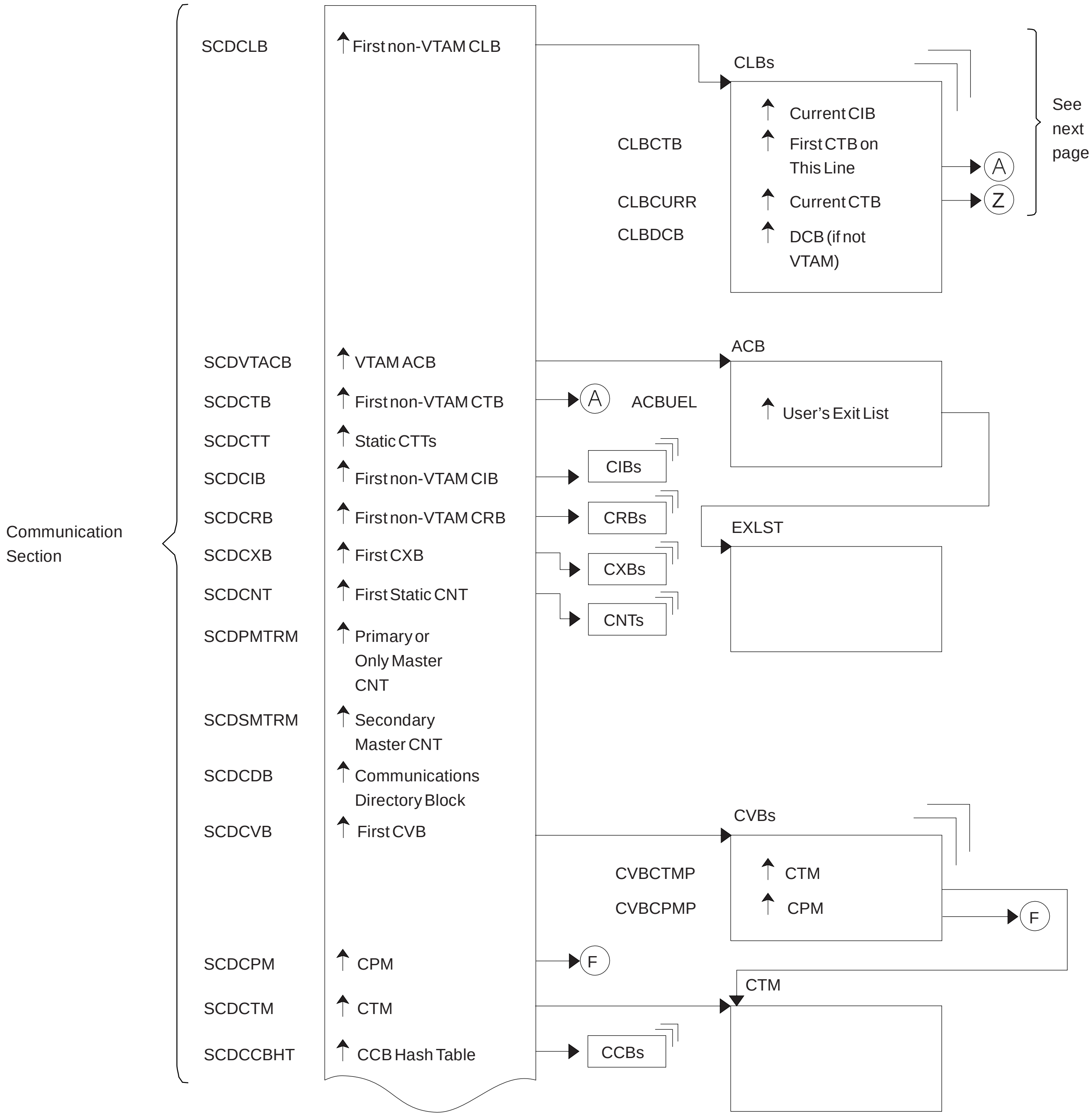


Figure 2. Online System Contents Directory (SCD) (Part 4 of 6)

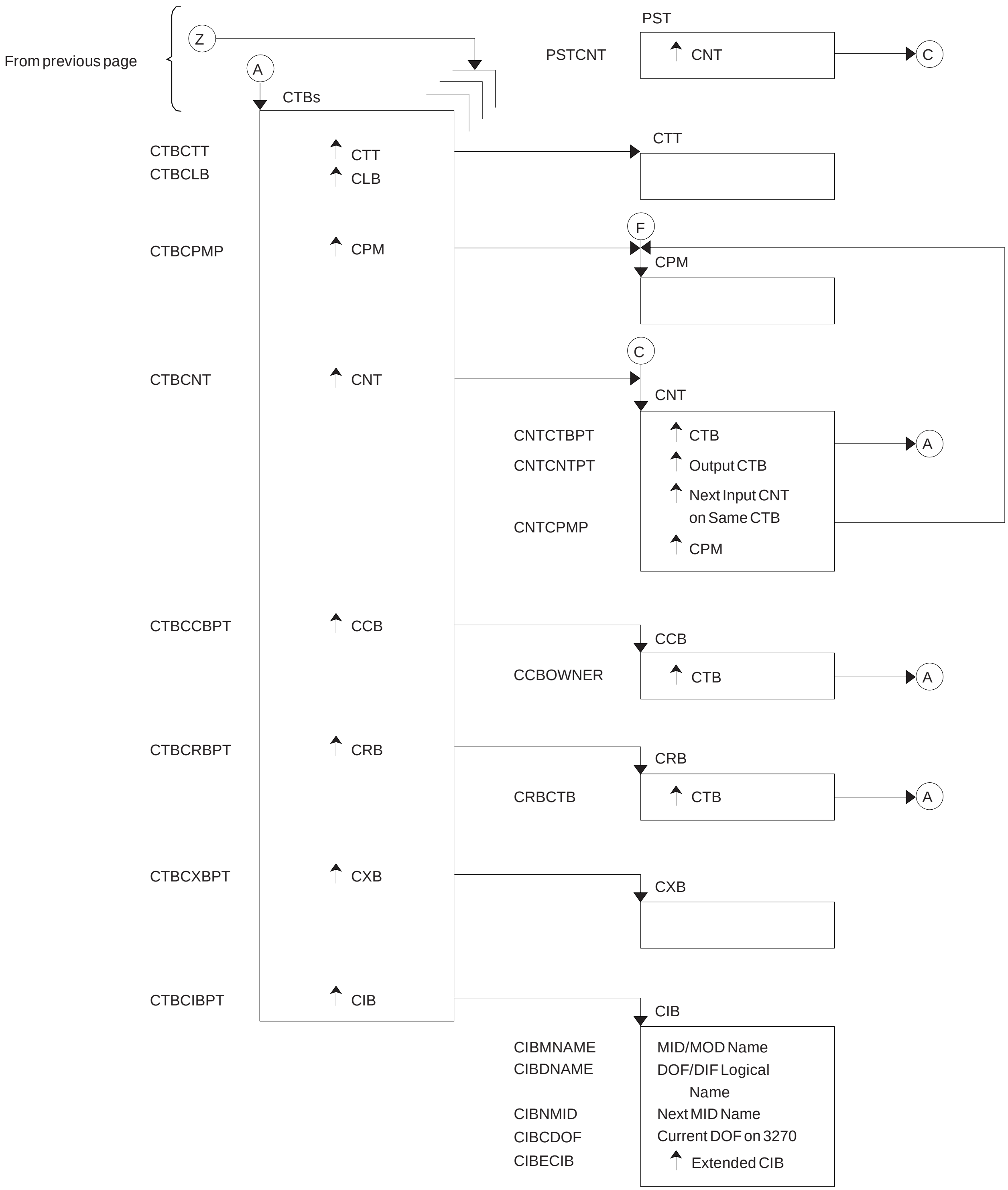


Figure 2. Online System Contents Directory (SCD) (Part 5 of 6)

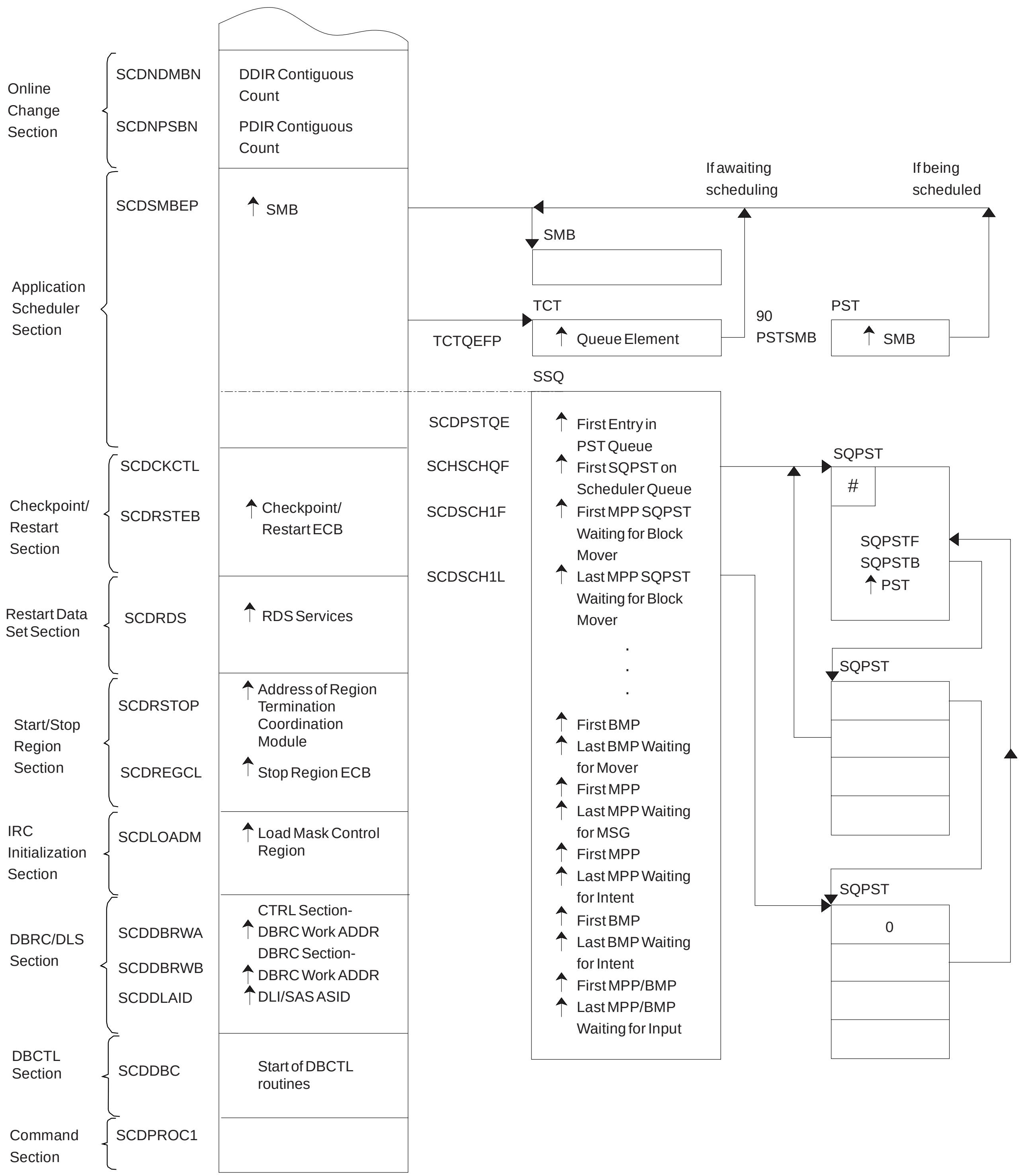
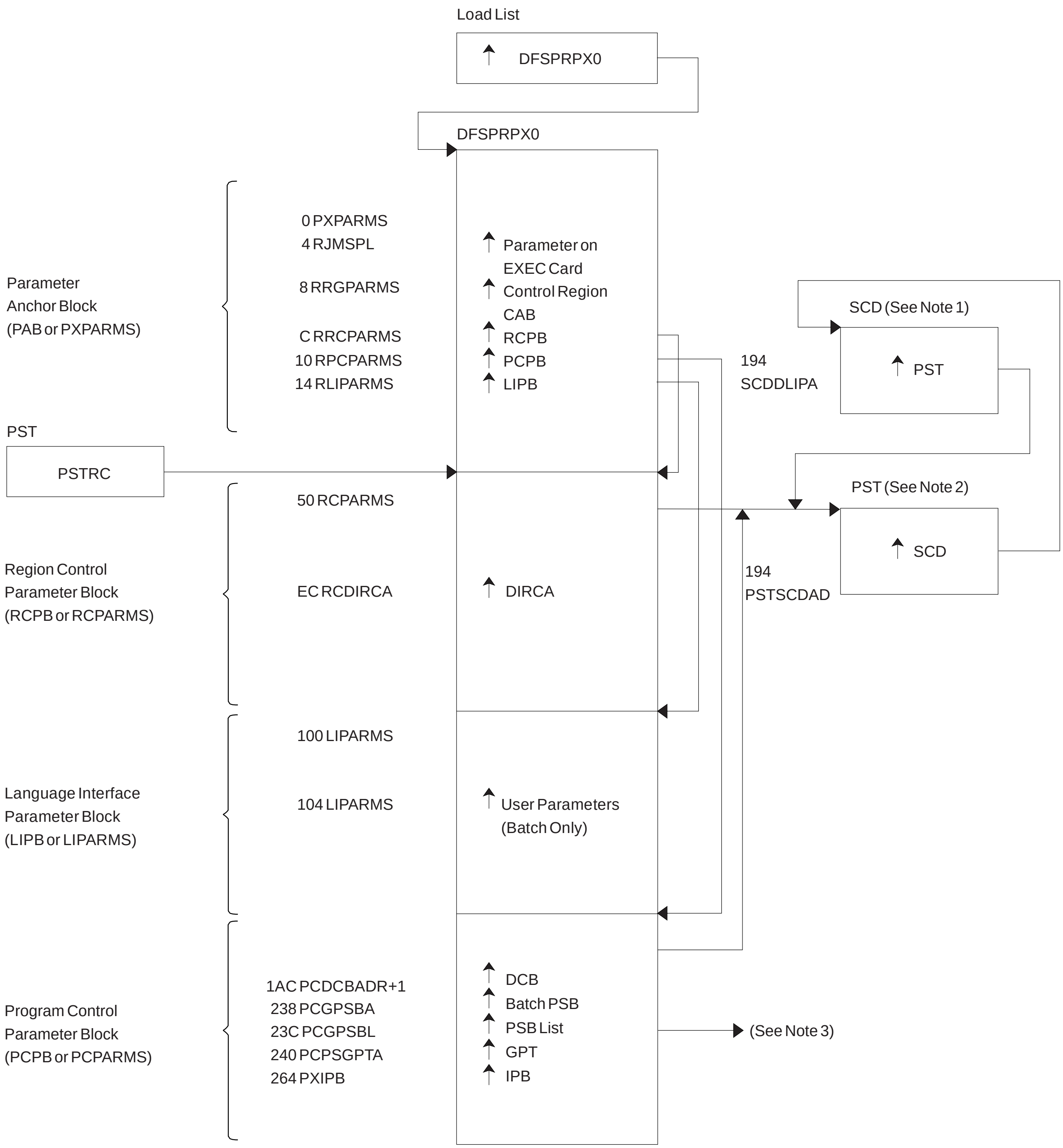


Figure 2. Online System Contents Directory (SCD) (Part 6 of 6)



- Note 1: See Figure 15 on page 82.
- Note 2: See Figure 14 on page 80.
- Note 3: See Figure 18 on page 85.

Figure 3. DFSPRPX0—Parameter Blocks

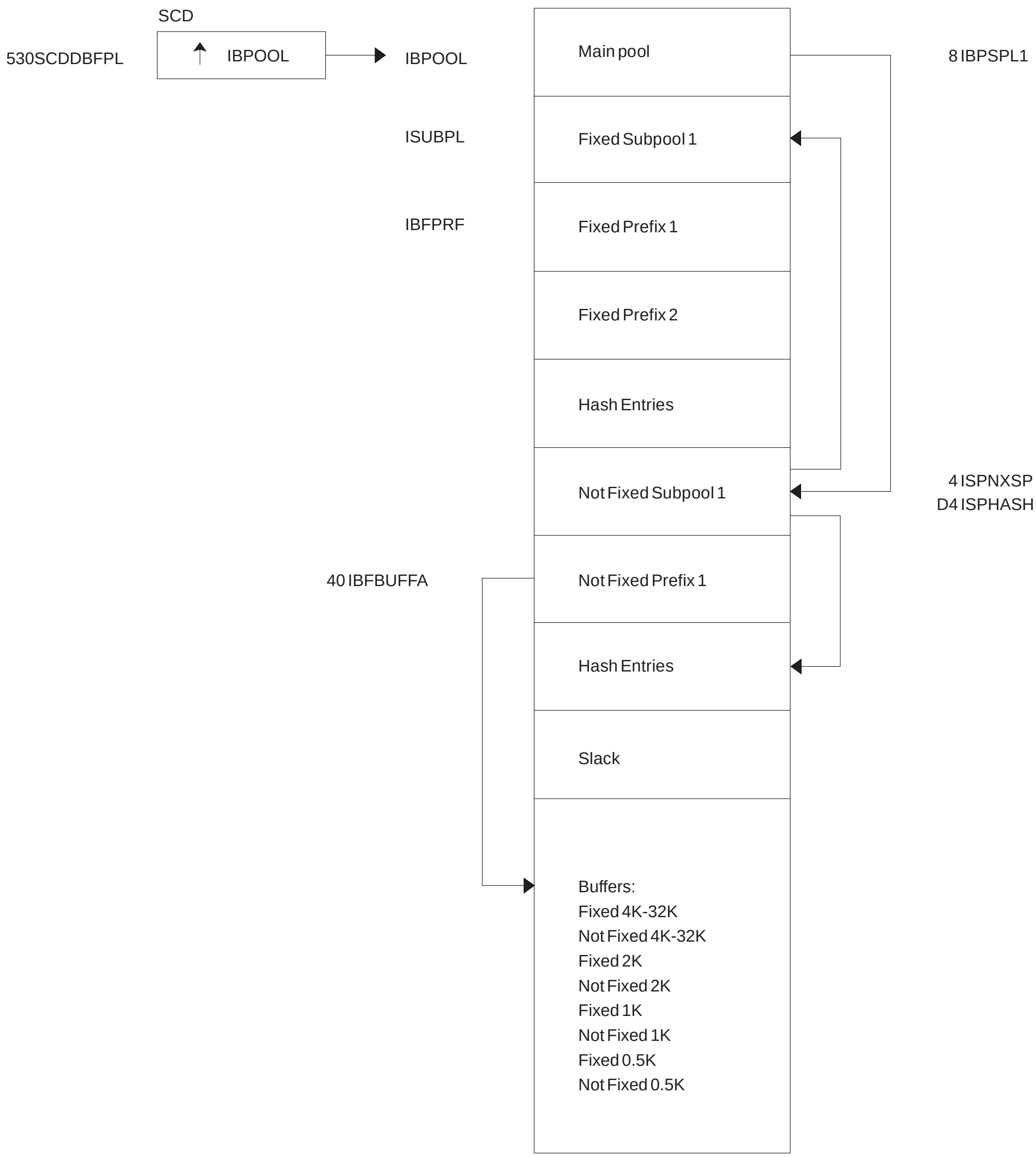


Figure 4. OSAM Buffer Pool

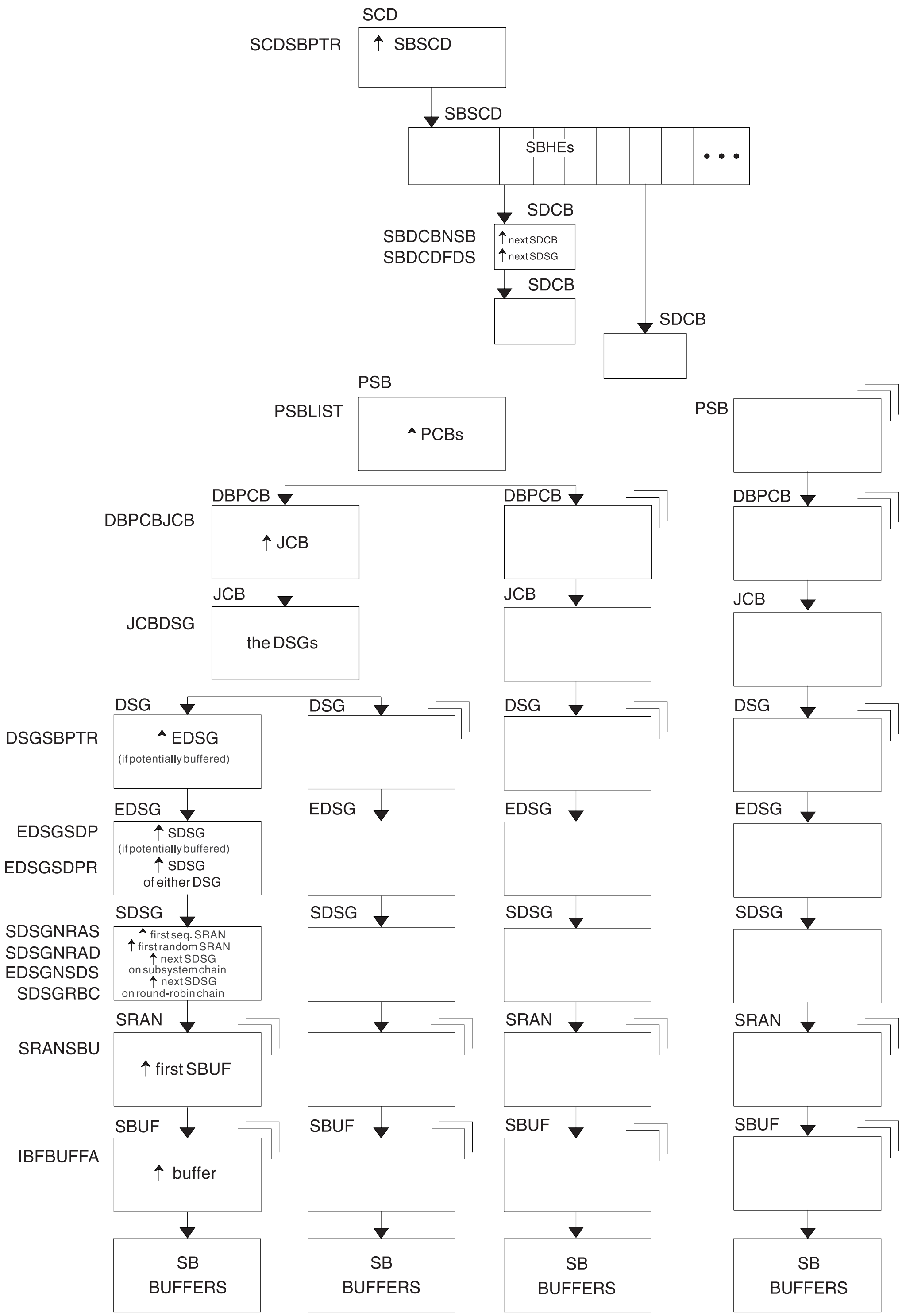


Figure 5. Sequential Buffering Control Blocks

Notes to Figure 5:

1. SCD is the IMS systems content directory.
2. SBSCD is a sequential buffering extension to the SCD.
3. SBHEs are sequential buffering control blocks located within the SBSCD (sequential buffering extension to the systems content directory). IMS uses SBHEs to:

- Anchor the sequential buffering extension to the DCB (DSCB)
- Serialize the SDCB and SDSG subsystem chains (defined in notes 4 and 8).

4. SDCB is a sequential buffering extension to the data communication block. There is one SDCB for each data set that is actively being sequentially buffered. IMS uses each SDCB to anchor any sequential buffering SDSGs that have buffer pools allocated to them.
5. The chains of SDCBs and SDSGs anchored in the SBHEs are called the SDCB and SDSG subsystem chains.
6. The program specification blocks, DBPCBs, job control blocks, and the data set group control blocks in the figure are DL/I control blocks.
7. EDSG is a sequential buffering extension to the DSG. The field EDSGSDP points to the SDSG if the data set group control block is potentially buffered by SB. If the DSG is not potentially buffered (but another DSG for the same data set and same application is), then the field EDSGSDPR points to one of the SDSGs of these "other" DSGs.
8. SDSG is a sequential buffering extension to the data set group control block. The SDSG is present if the user wants to have the DSG sequentially buffered. The SDSG is the control block that controls one sequential buffering buffer pool.
9. SRAN is a sequential buffering control block that describes references in one set of recently referenced consecutive data set blocks.
10. SBUF is a sequential buffering control block that describes one individual buffer.

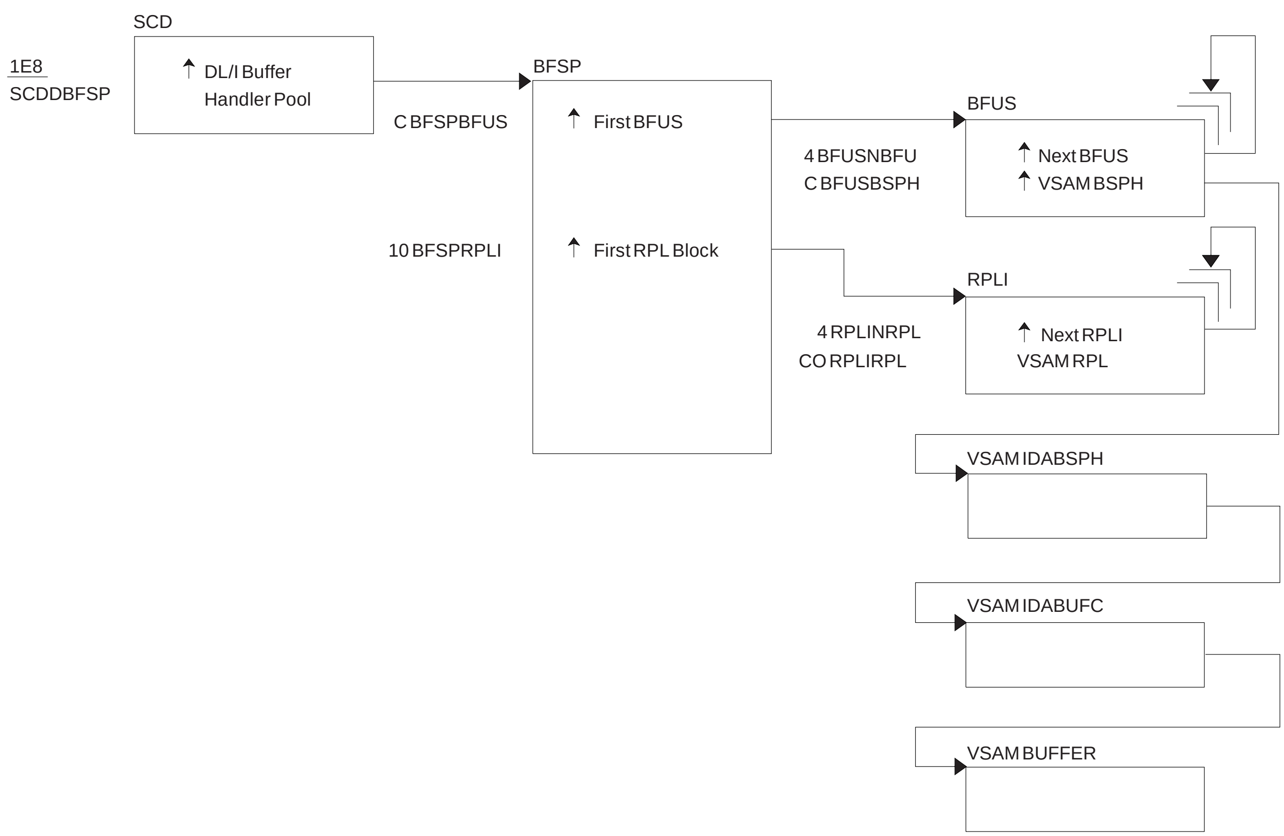


Figure 6. Buffer Handler Pool (VSAM)

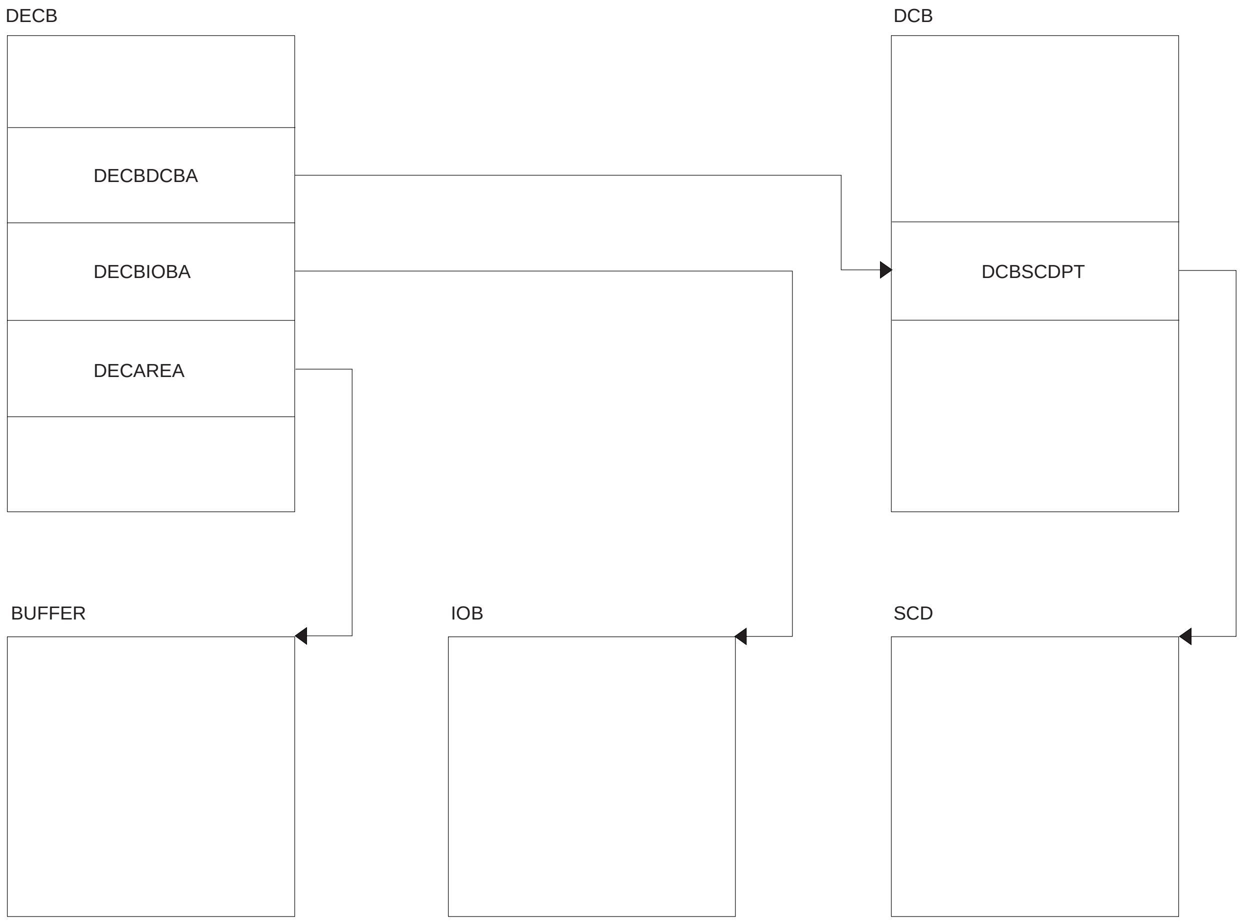


Figure 7. OSAM DECB with IOB in Use

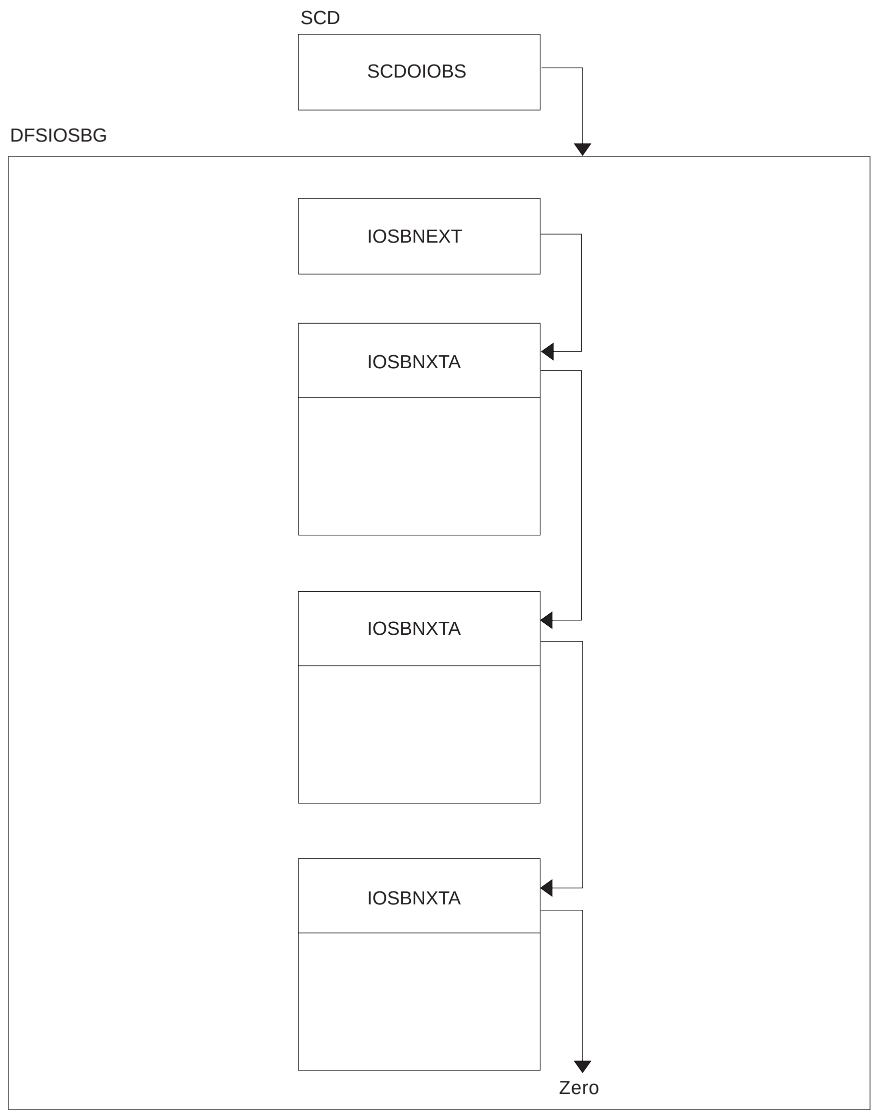


Figure 8. OSAM IOB Pool Showing Available IOBs

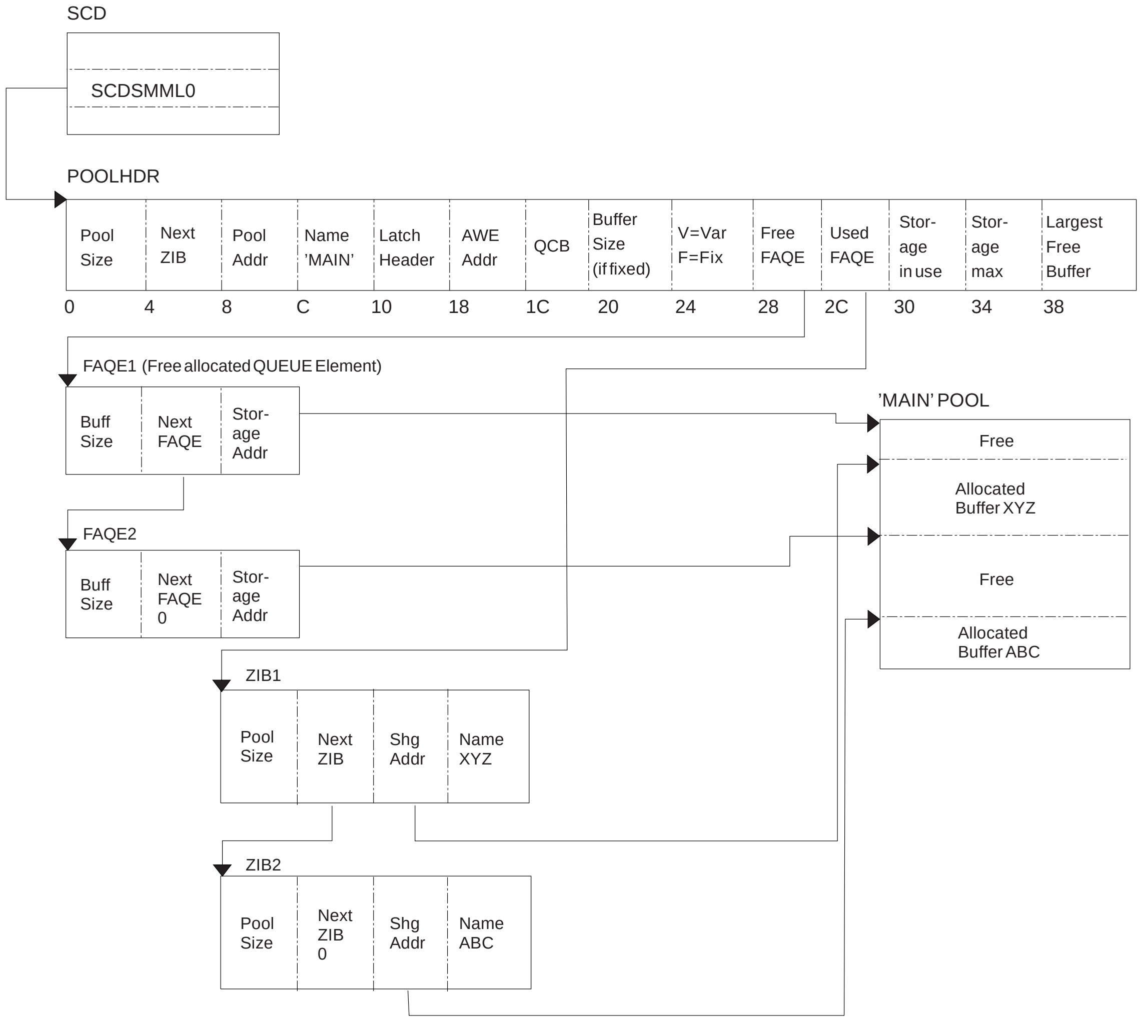


Figure 9. Storage Management Control Block Relationships Created for the MAIN Pool

Storage allocated using the ICREATE/IDESTROY macros is obtained from the MAIN (WKAP) pool. The control block relationship for the MAIN pool is shown in Figure 9.

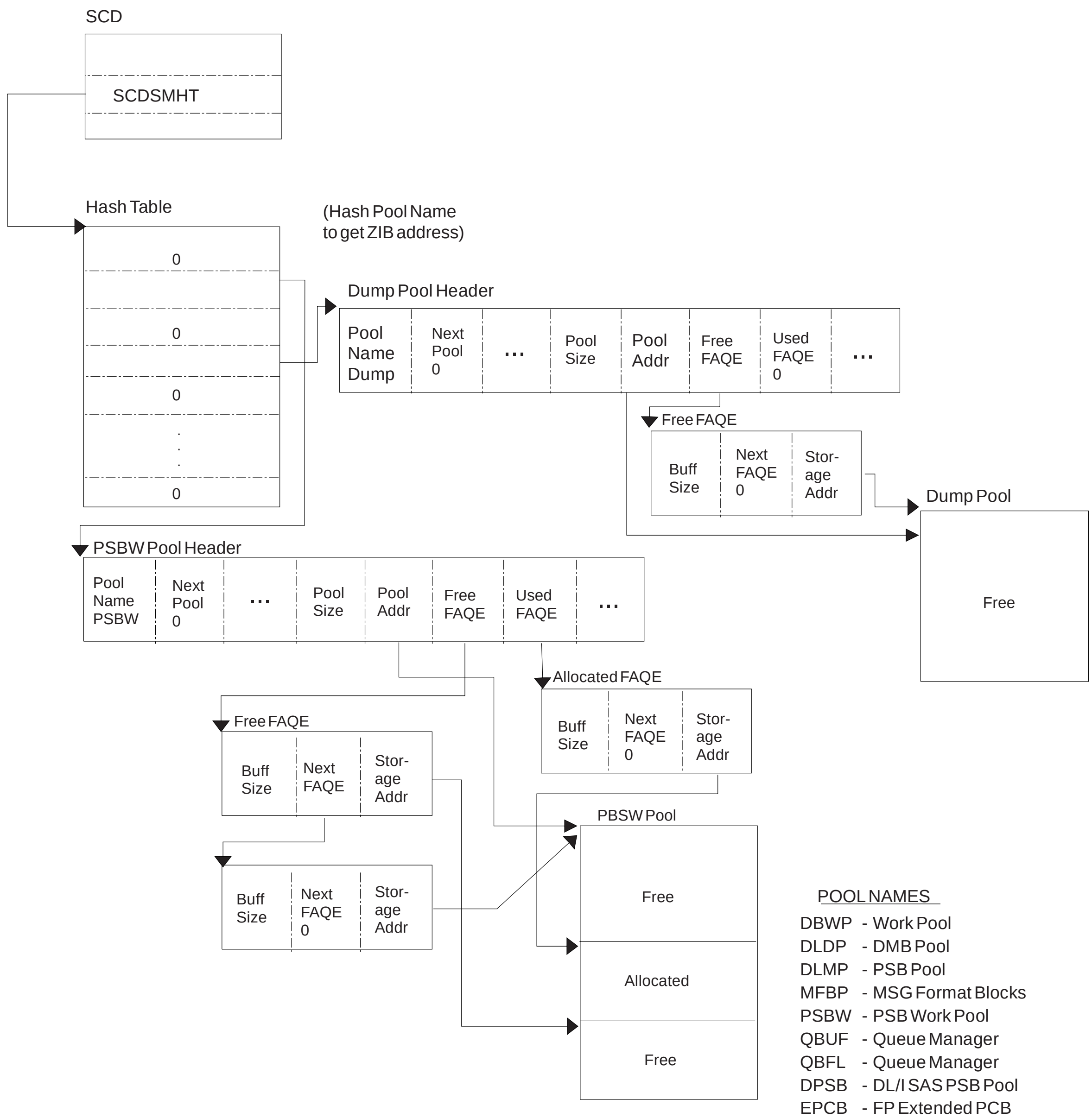


Figure 10. Storage Management Control Block Relationships for Preallocated Storage Blocks

Figure 10 shows the control block relationships for those pools managed by the DFSISMN0 Storage Manager.

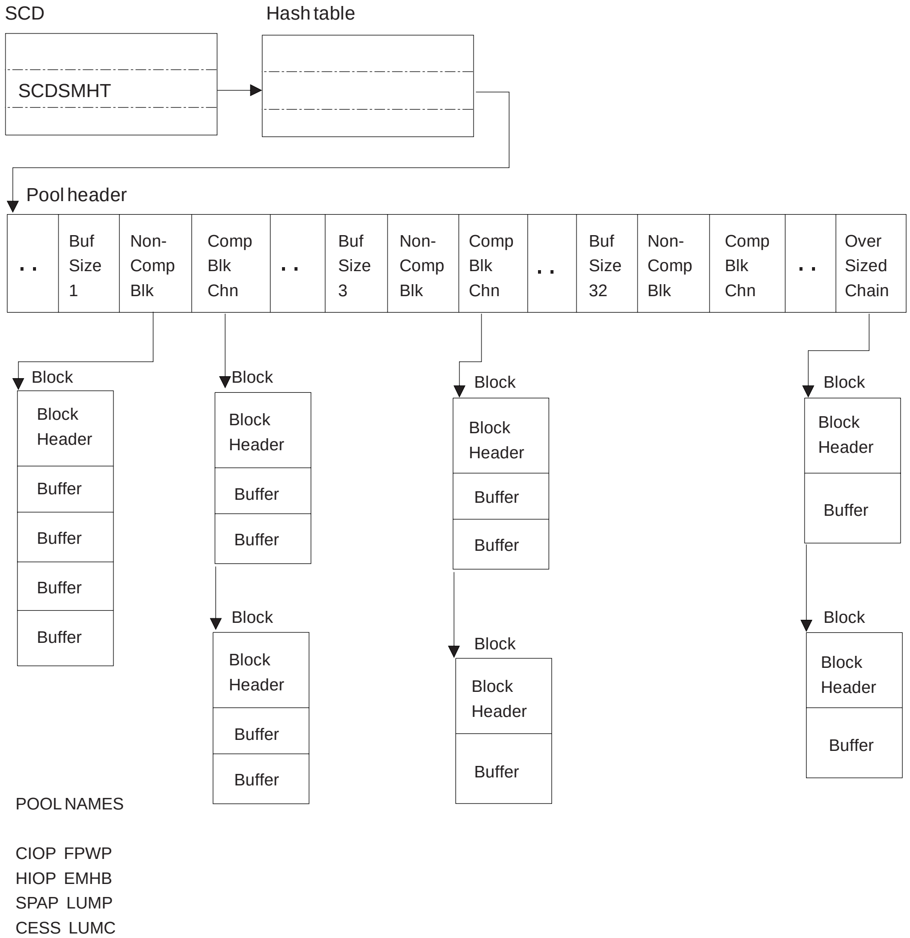


Figure 11. Storage Management Control Block Relationships (DFSPOOL Pools)

Figure 11 shows the control block relationship for pools managed by the DFSPOOL Storage Manager. Each pool consists of zero or more noncontiguous storage blocks anchored off a pool header. By obtaining new blocks and releasing unused blocks, you can expand and contract a pool as needed during the execution of IMS.

Each block is divided into a number of fixed-length buffers that are used to satisfy storage requirements. The size and number of buffers can vary from block to block within a pool. Each block also has a block header which contains various information on the block

Each pool can be allocated with a maximum of thirty-two different buffer sizes. The pool header contains a noncompressible block pointer and a compressible block chain anchor for each buffer size available.

The pool header also contains an oversized block chain anchor. If the request size is larger than the largest buffer size available, a block is obtained containing a single buffer of the requested size. Blocks obtained in this manner are placed on the oversized chain. The intention of the oversized chain is to allow for exceptional requests, since normal processing should not need any oversized buffers.

The first block allocated for each buffer size is referred to as the primary block. The number of buffers contained within the primary block can vary from any secondary blocks of the same buffer size. If the primary block is obtained when the pool is allocated, it is held until IMS termination. Because it cannot be compressed, serialization logic is not required when allocating or releasing a buffer from one of these blocks.

If the primary block is not obtained until the first GET request, it along with any secondary blocks are placed on the compressible block chain anchored off the pool header. Serialization logic must be used when scanning the blocks on the compressible chains.

An eight-byte prefix and an eight-byte suffix is added to each buffer. The prefix and suffix are used by the Storage Manager exclusively. The size of the prefix and suffix is included in the current pool size.

The buffer size used to satisfy an incoming request is determined on a best fit basis. Unless the size of the buffer requested is the same size as the actual buffer, there will be some unused storage between what the caller views as the end of the buffer and the actual end of the buffer. The buffer the user receives appears to be of the size requested. Any unused space is transparent.

The following pools are defined with user overlay detection: CIOP, HIOP, SPAP, EMHB, LUMC, and LUMP. If a pool is defined with user overlay detection, an eight-byte constant is added to the user portion of the buffer. As far as the caller is concerned, the length of buffer received is the length requested followed by an eight-byte constant. For example, if a caller requests a 100-byte buffer from a pool with a user overlay detection, and the smallest buffer size available to satisfy the request is 128 bytes, the user overlay detection constant is placed at an offset of 100 bytes into the buffer. Bytes 107 through 127 are unused.

The user overlay detection constant is used by IMS modules. The Storage Manager does not look at the user overlay detection constant.

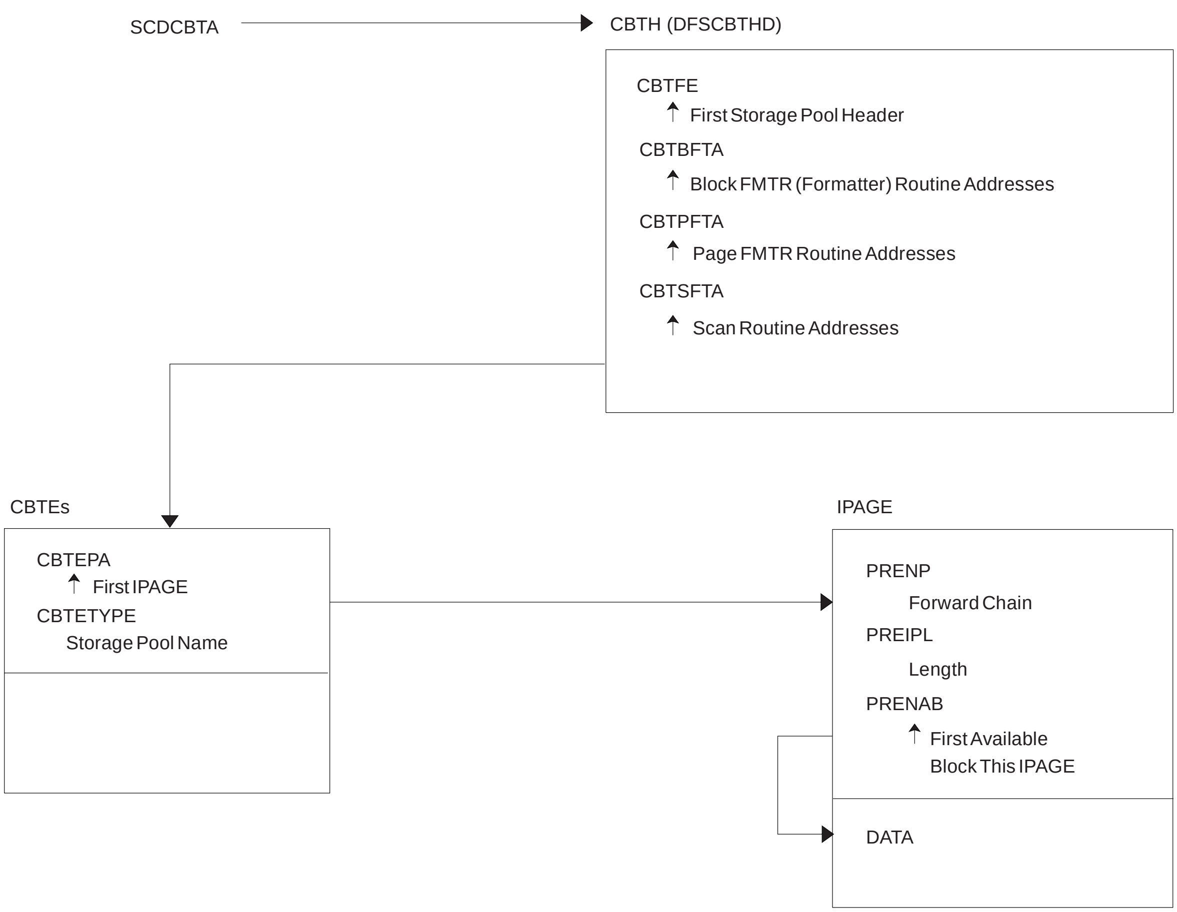


Figure 12. Storage Management Control Block Relationships (DFSCBT00 Pools)

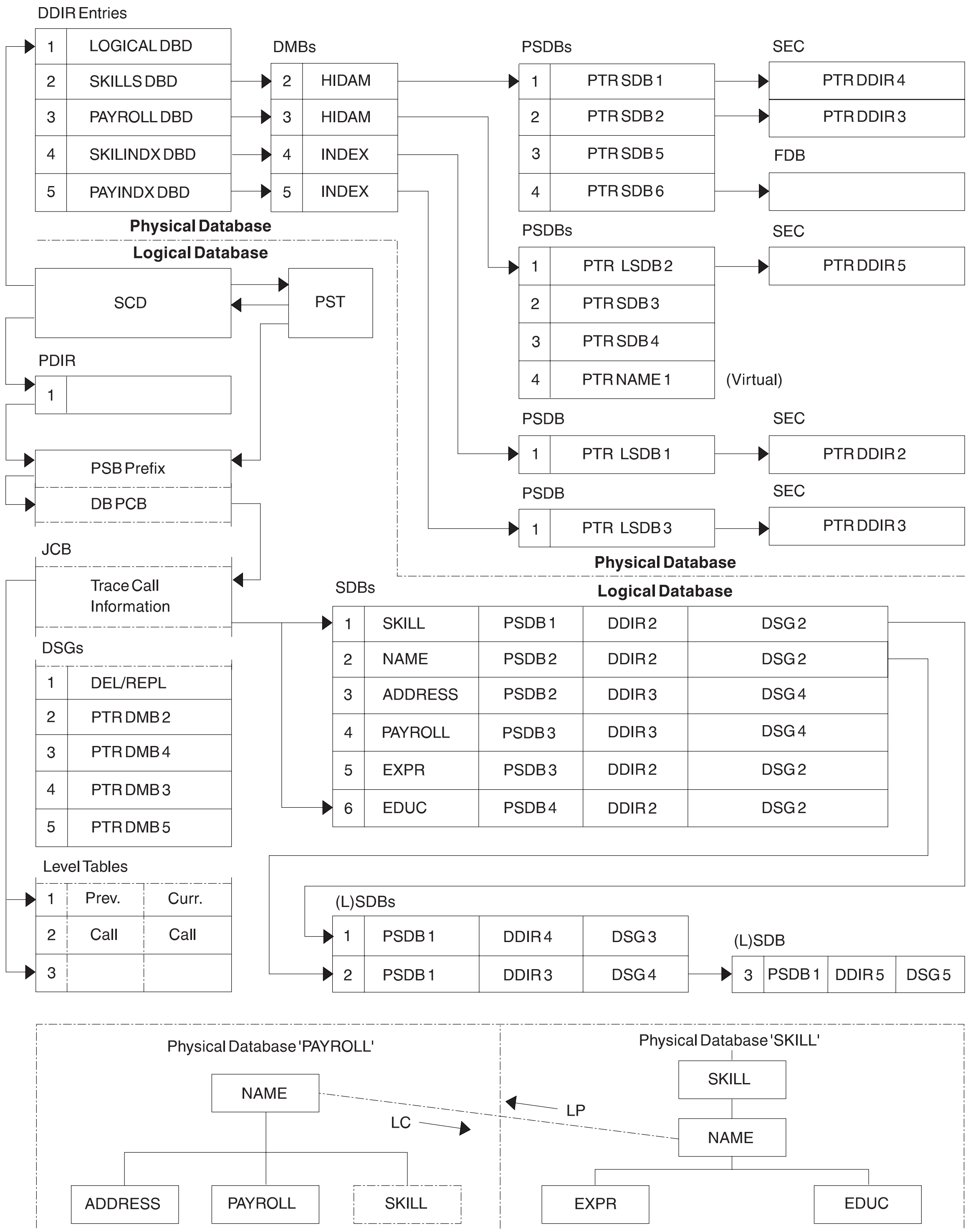
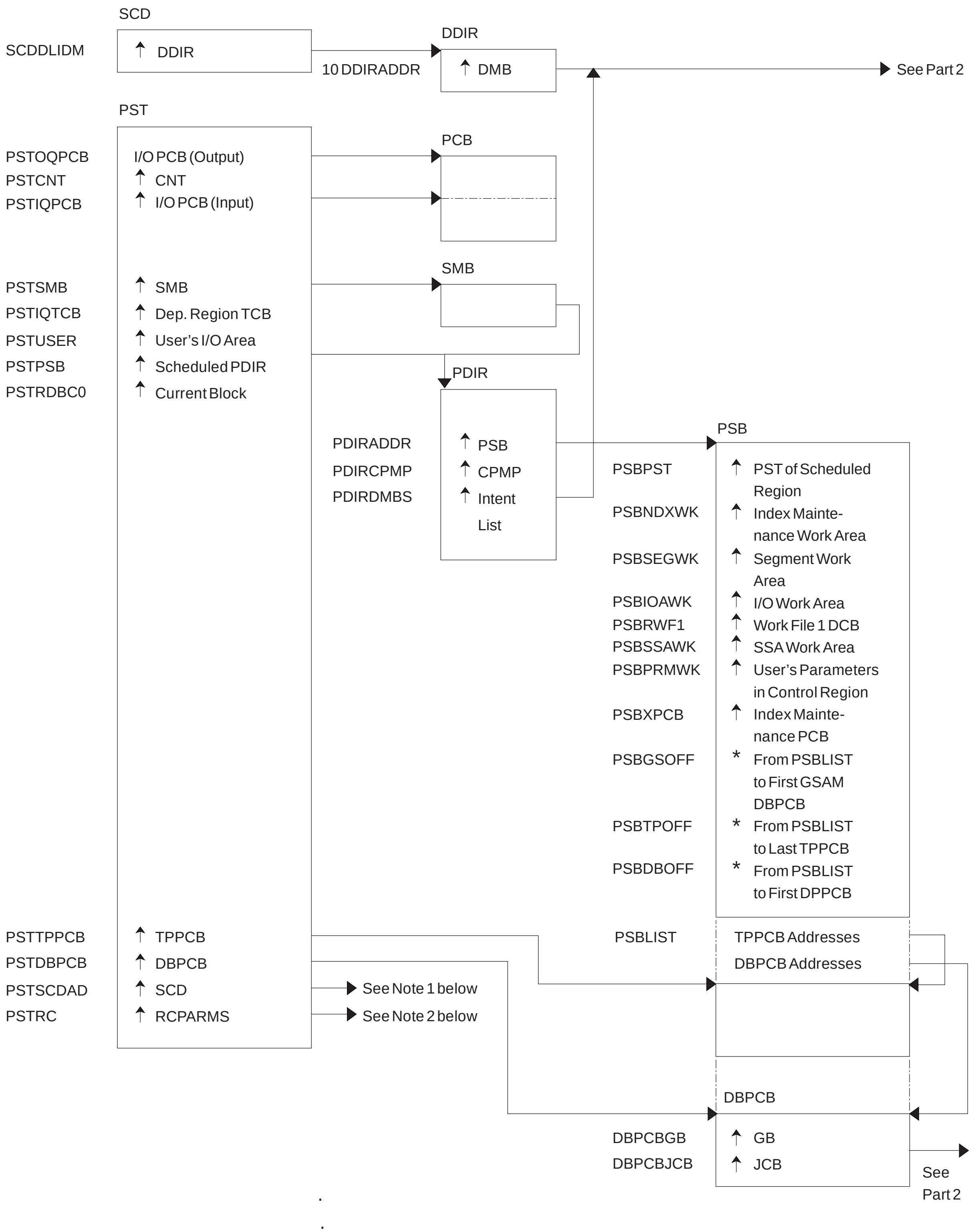


Figure 13. Database Manager Control Blocks for a Representative Database



- Note 1: See Figure 2 on page 63.
- Note 2: See Figure 3 on page 69.

Figure 14. Database Control Blocks (Part 1 of 2)

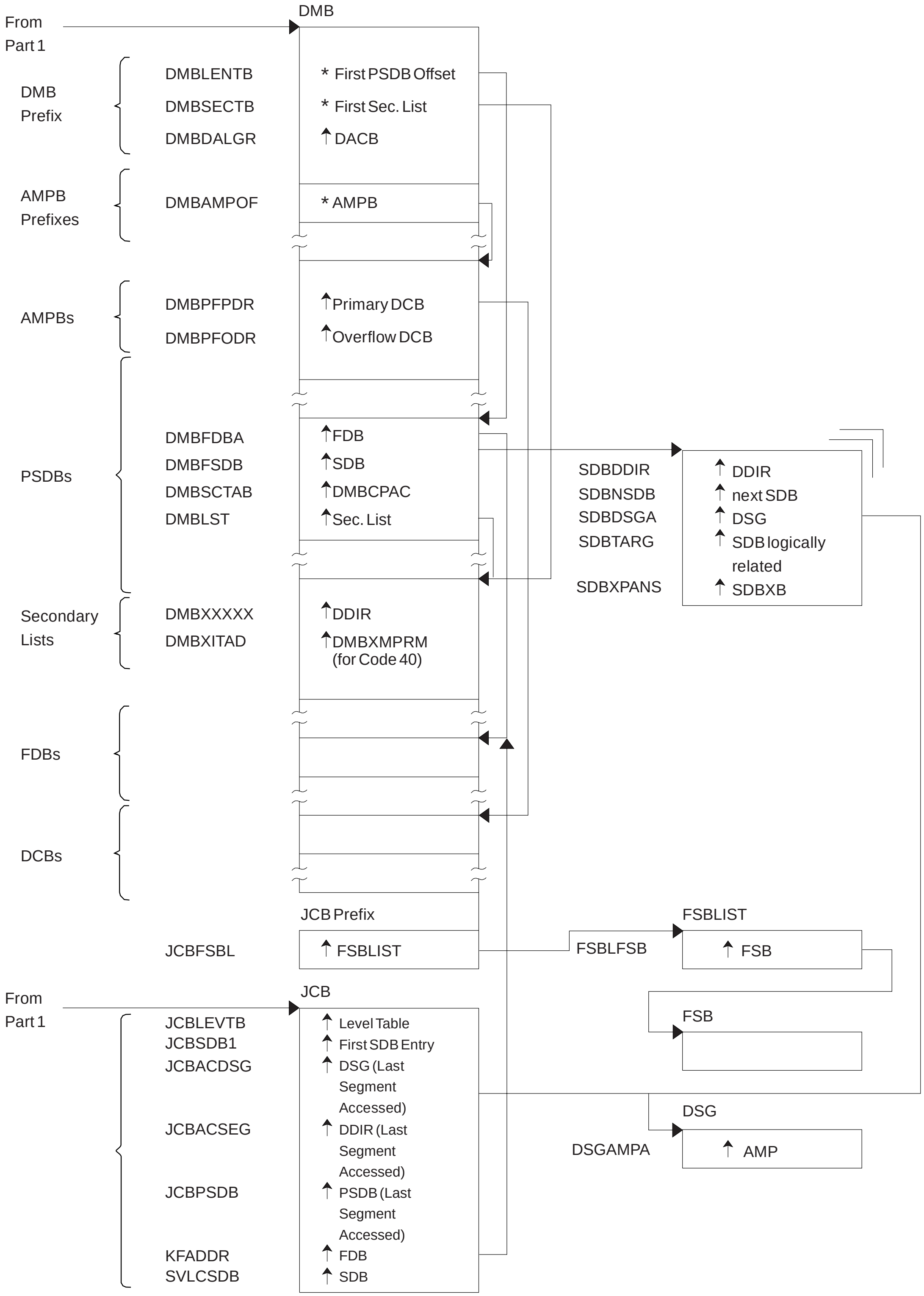


Figure 14. Database Control Blocks (Part 2 of 2)

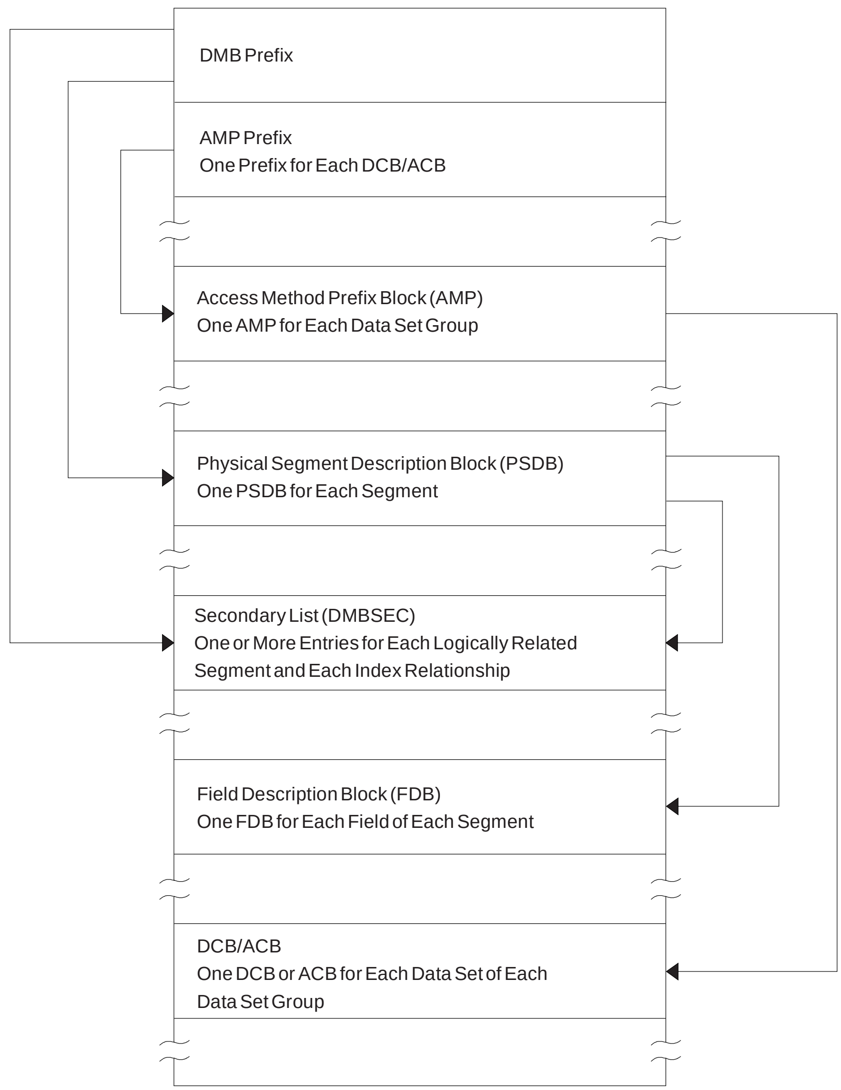


Figure 15. Diagram of a Data Management Block (DMB)

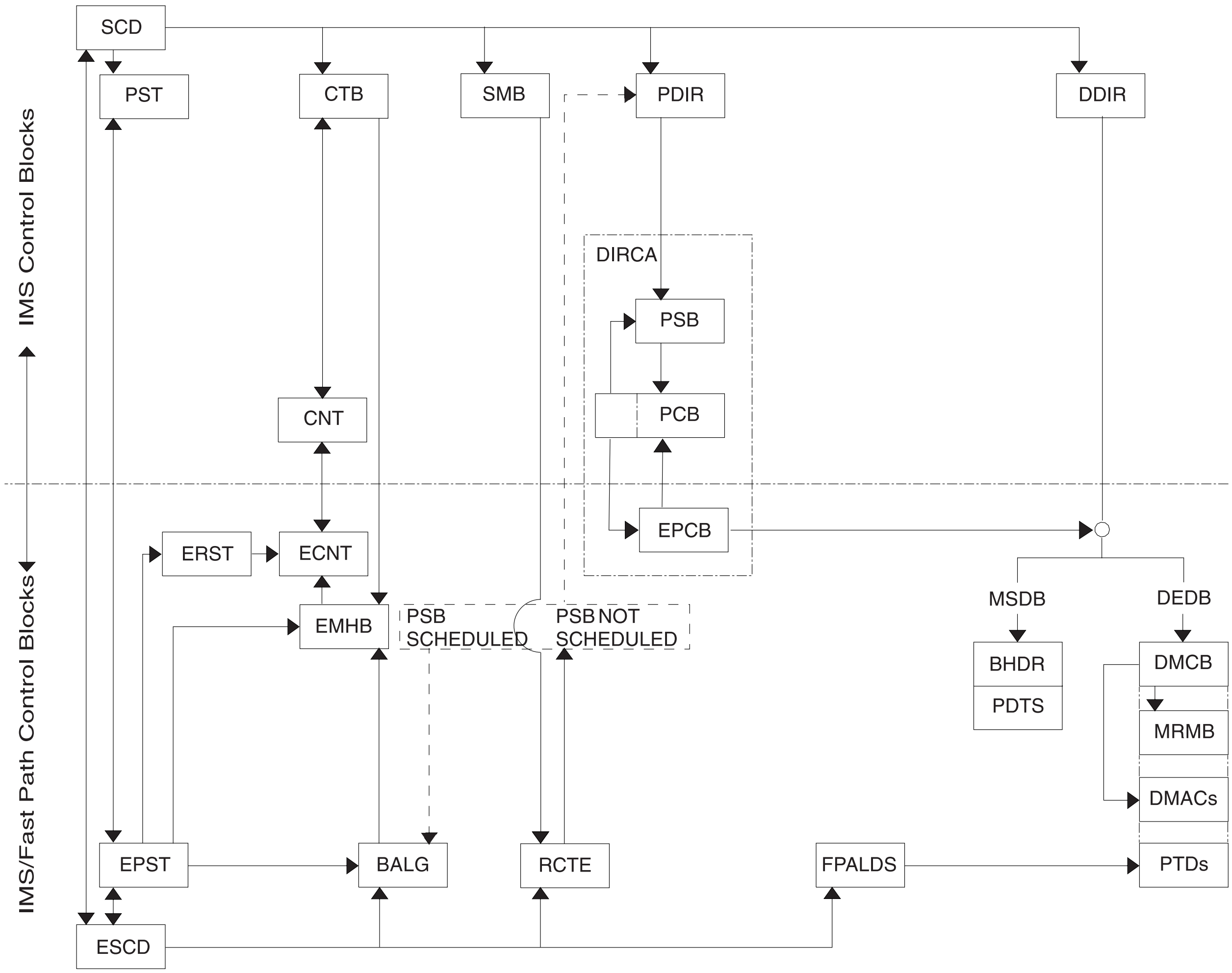


Figure 16. Overview of Fast Path Control Blocks

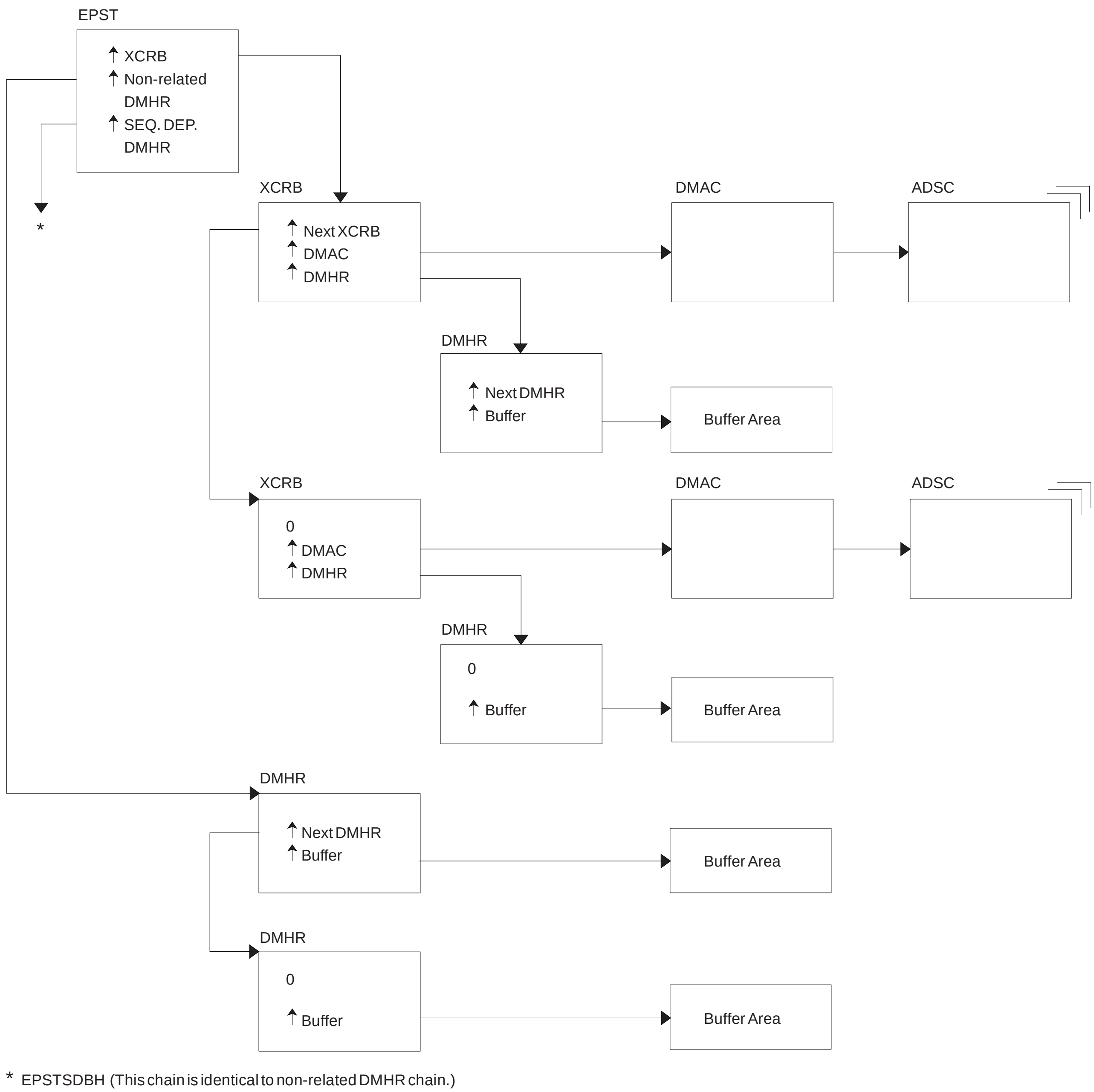


Figure 17. Relationships Between Buffer Control Blocks for Fast Path Databases

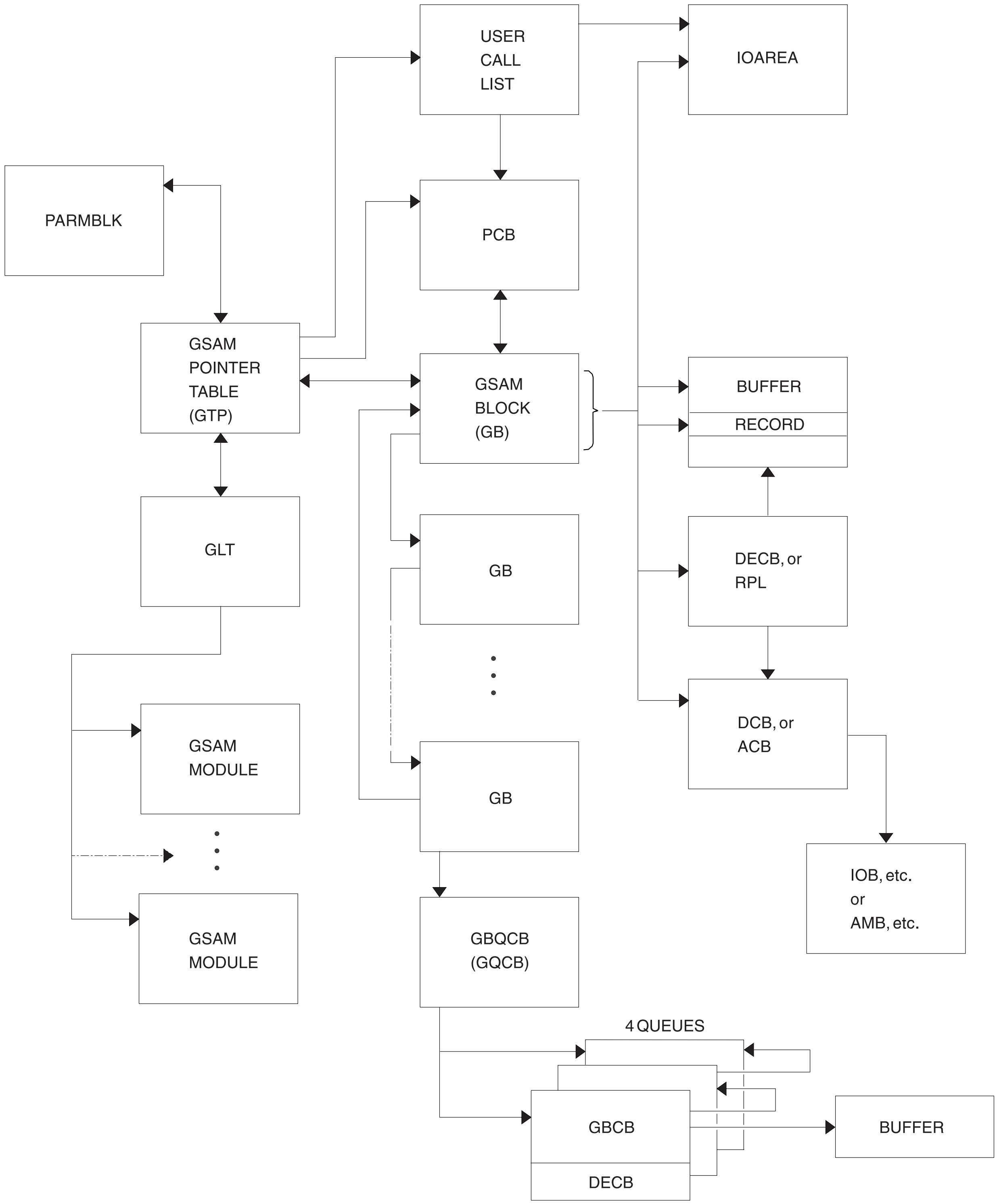


Figure 18. GSAM Control Block Overview

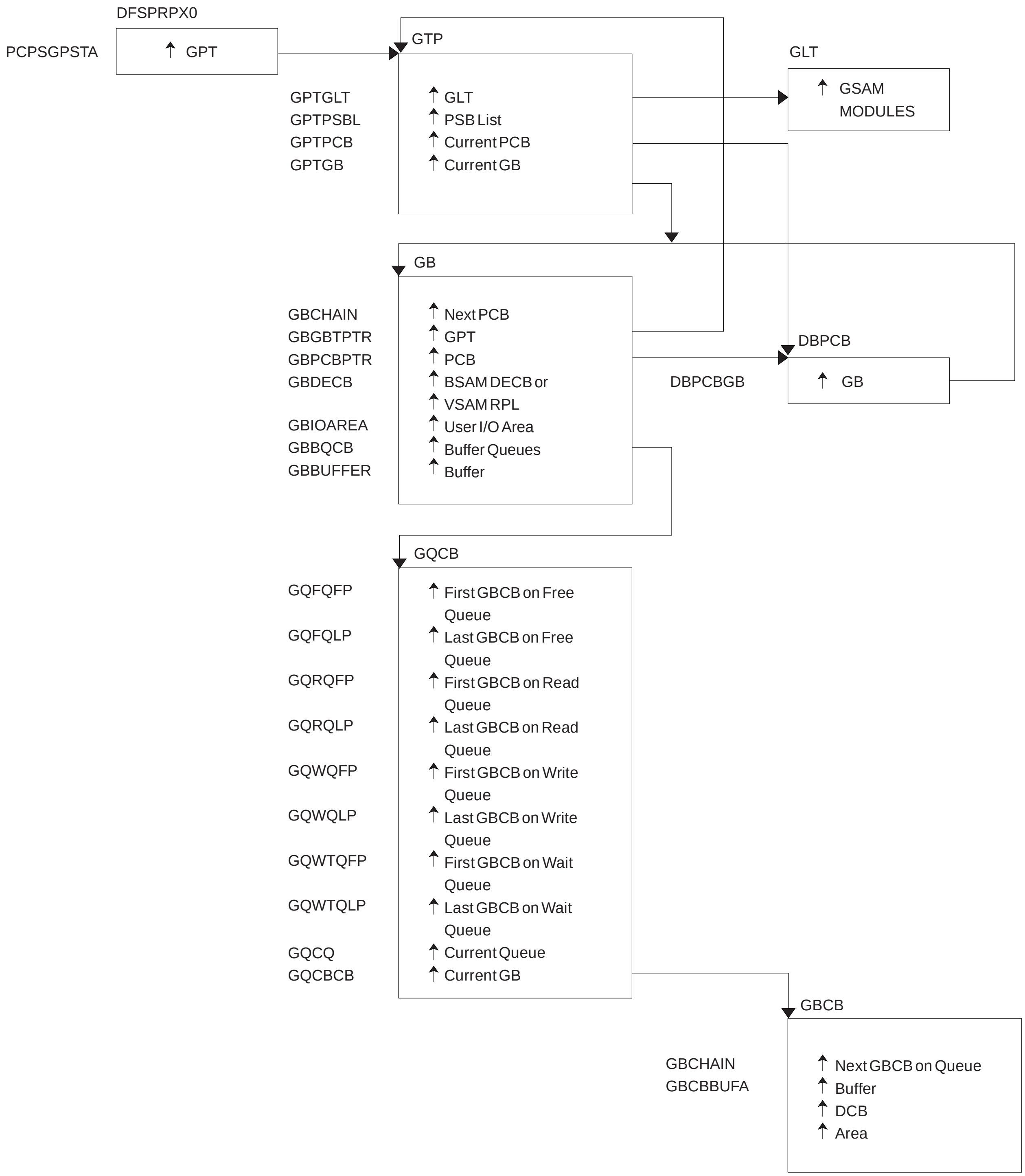
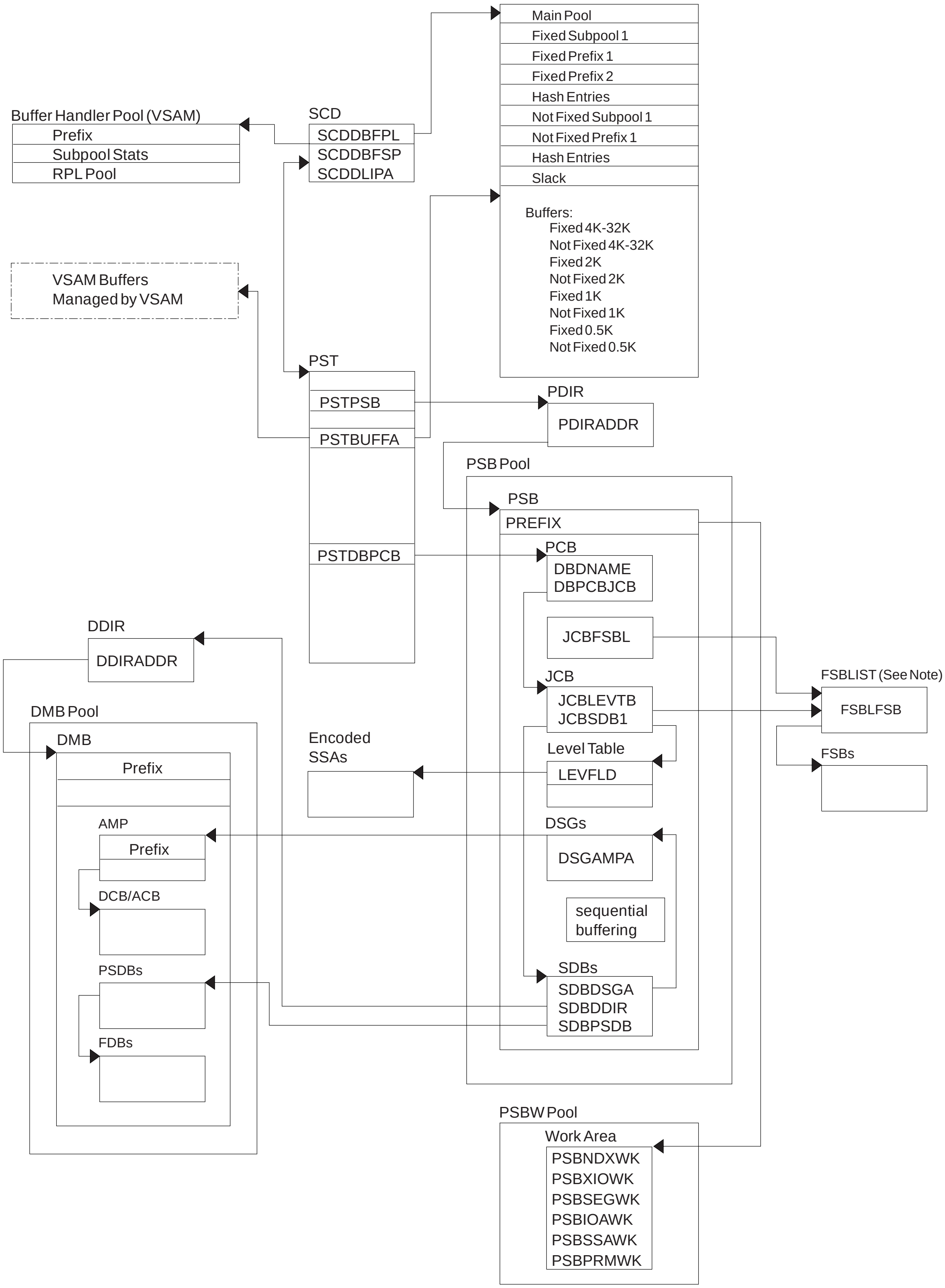


Figure 19. GSAM Control Blocks



Note: The FSBLIST contains pointers to the Field Sensitivity Block (FSB). The FSB describes this user's logical use of the sensitive field.

Figure 20. DL/I Control Block Relationships

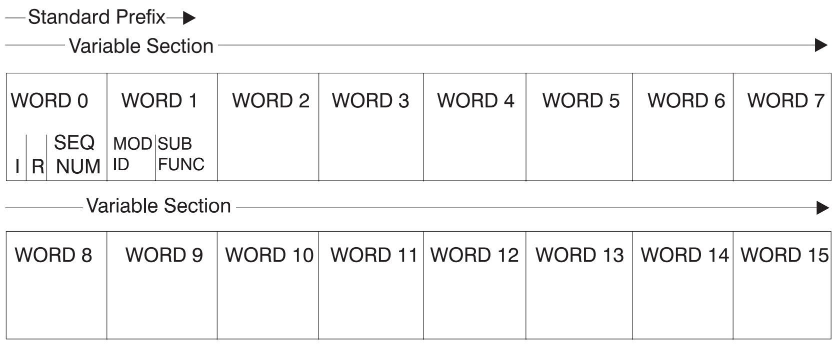
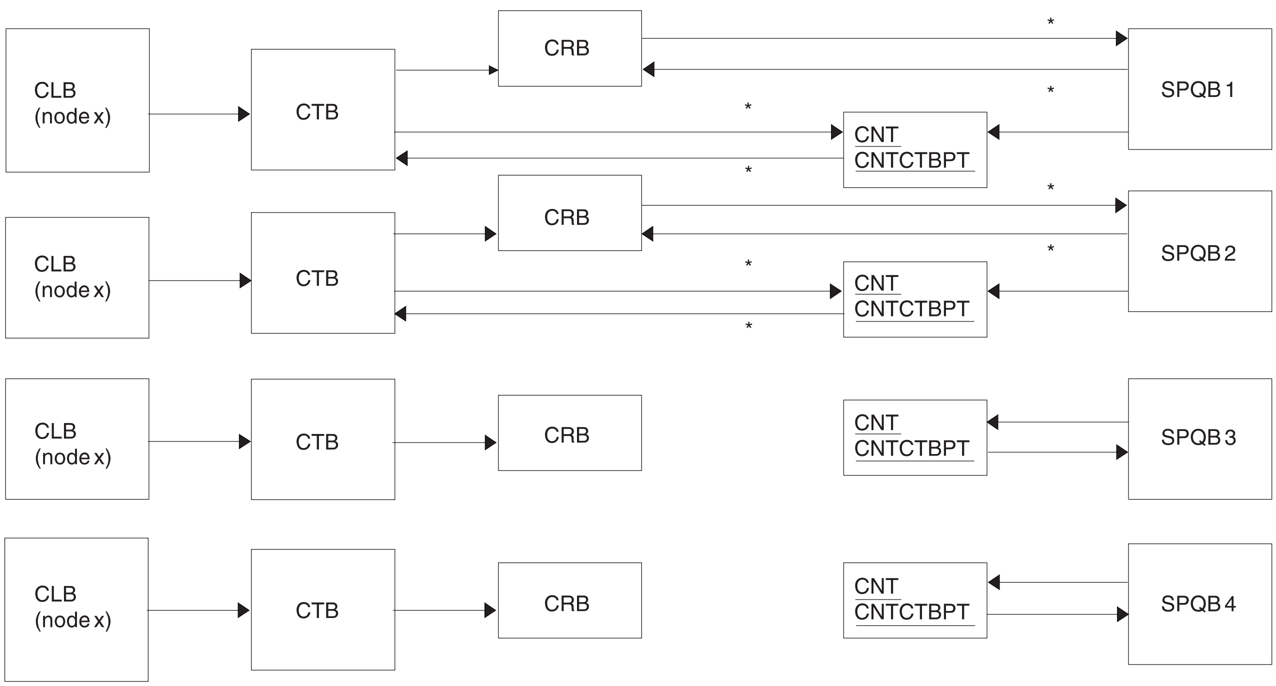


Figure 21. IMS Transaction Manager Control Blocks



Note

Subpool Queue Blocks (SPQB1 and SPQB2) are allocated for sessions. SPQB3 and SPQB4 are not. One SPQB is required for each parallel session.

* Asterisks indicate that these pointers are set when blocks are allocated.

Figure 22. Intersystem Communication Control Block Structure

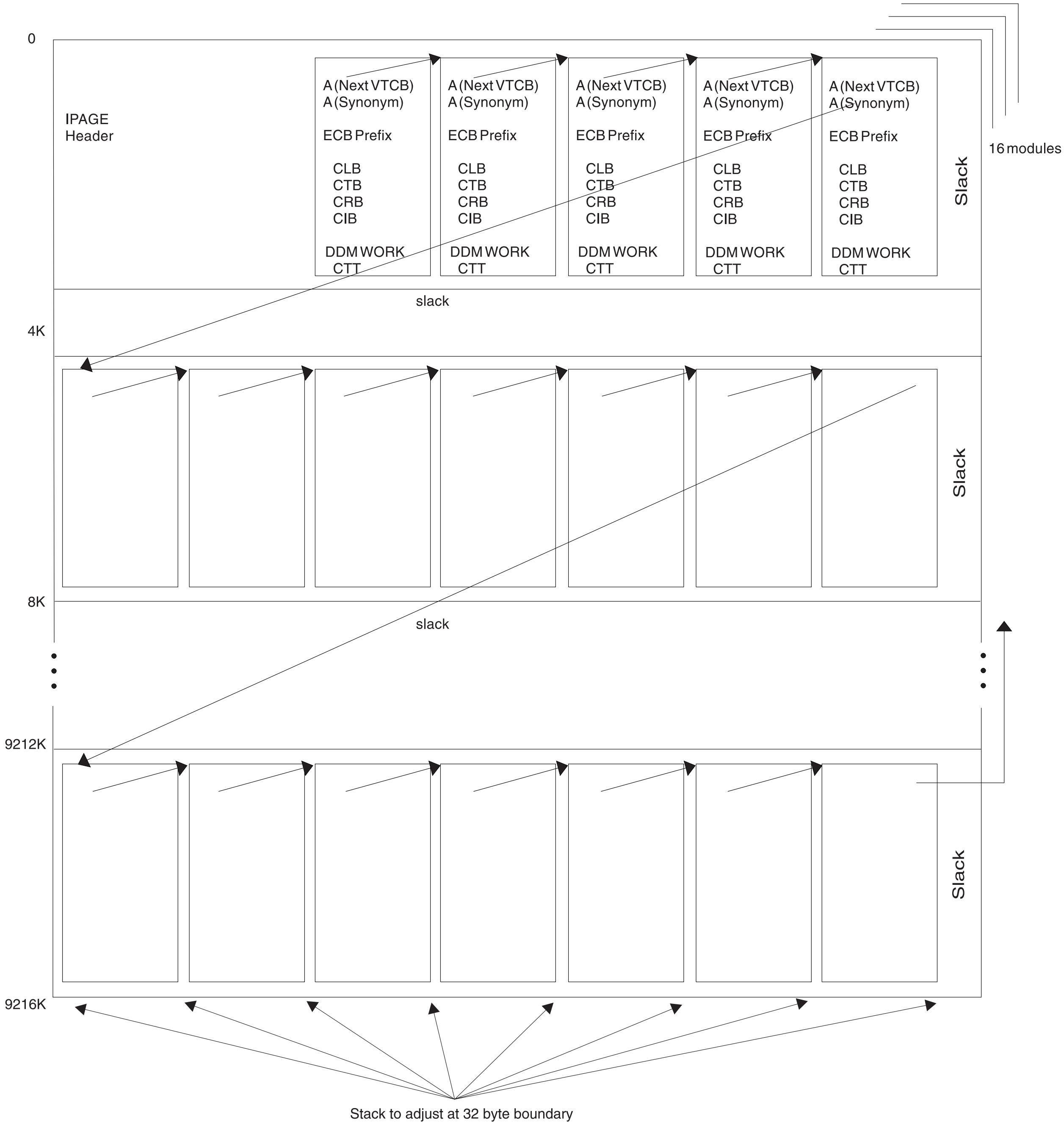


Figure 23. VTCB Load Module

As illustrated in Figure 23, IMS maintains a VTAM terminal control block (VTCB) for each VTAM terminal except MSC VTAM terminals. A VTCB can contain a:

- Communication line block (CLB)
- Communication terminal block (CTB)
- Communication restart block (CRB)
- Communication interface block (CIB)
- Device-dependent module (DDM) work area
- Communication terminal table (CTT) (used only for ETO terminals)

The system of pointers between blocks within a VTCB is the same as the system of pointers used for BTAM terminals.

Some terminals do not require all six blocks. For example, static VTAM blocks use a statically created CTT.

You can find the VTCB for a terminal through the terminal's node name. To do so, you use the DFSCBTS macro interface.

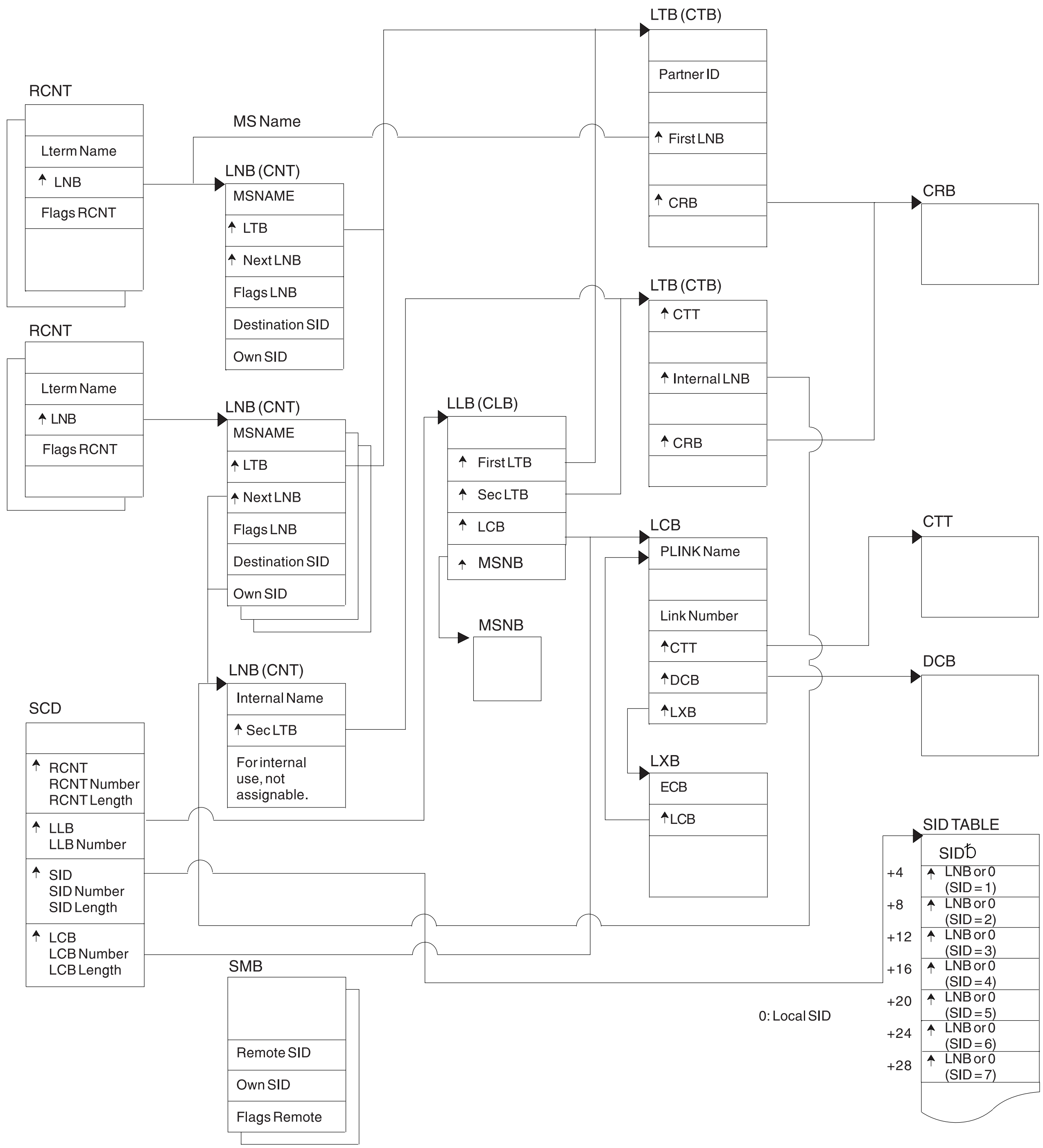


Figure 24. Multiple Systems Coupling (MSC) Control Block Overview

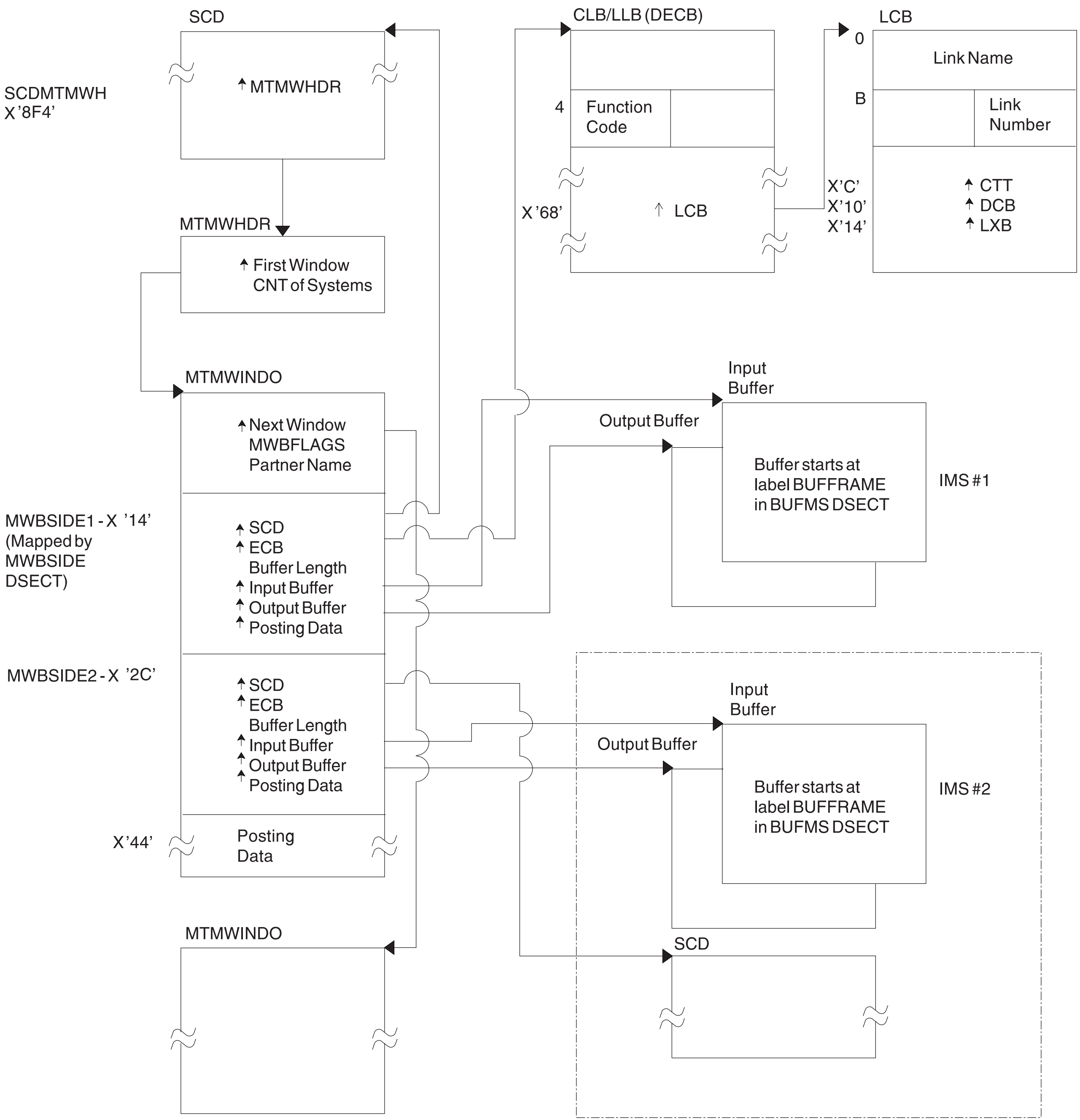


Figure 25. Multiple Systems Coupling (MSC) Main Storage-to-Main Storage Control Block Overview

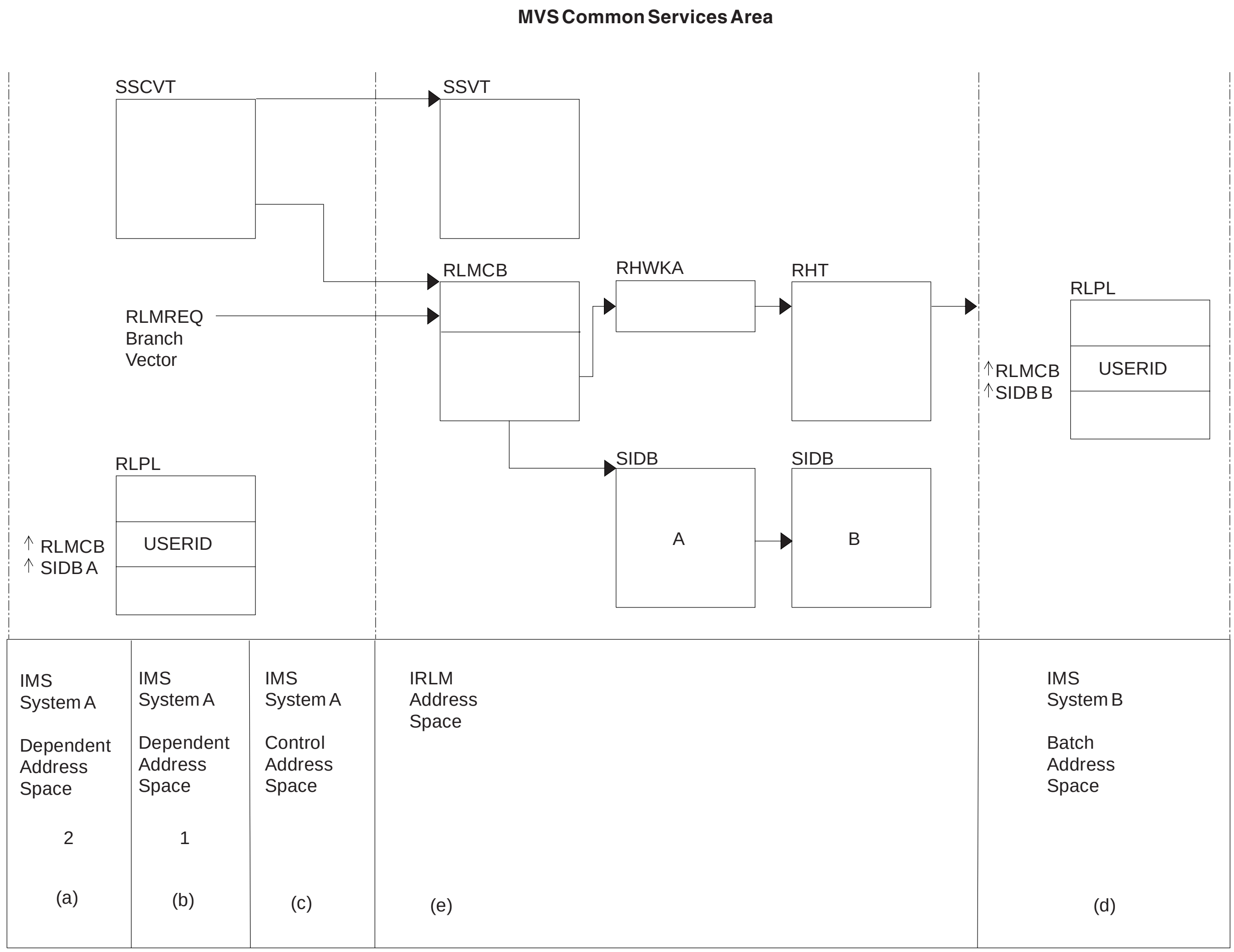


Figure 26. MVS Storage Map Showing IMS-to-IRLM Interrelationships

Notes to Figure 26:

1. (a), (b), and (c) are MVS address spaces that make up one online IMS subsystem.
2. (d) is an MVS address space containing an IMS batch subsystem.
3. (e) is an IRLM address space to which the two IMS subsystems are connected.
4. The RLPLs used by both IMS subsystems reside in the MVS common services area (CSA).
5. To obtain and release global locks, the IMS subsystems branch to the IRLM code (The subsystems enter the IRLM code through the RLMREQ branch vector within the RLMCB that resides in the CSA.)
6. The IRLM control block structure that controls the global locks resides in the CSA.
7. When PC=YES is in effect, the RHT is in a private address space.

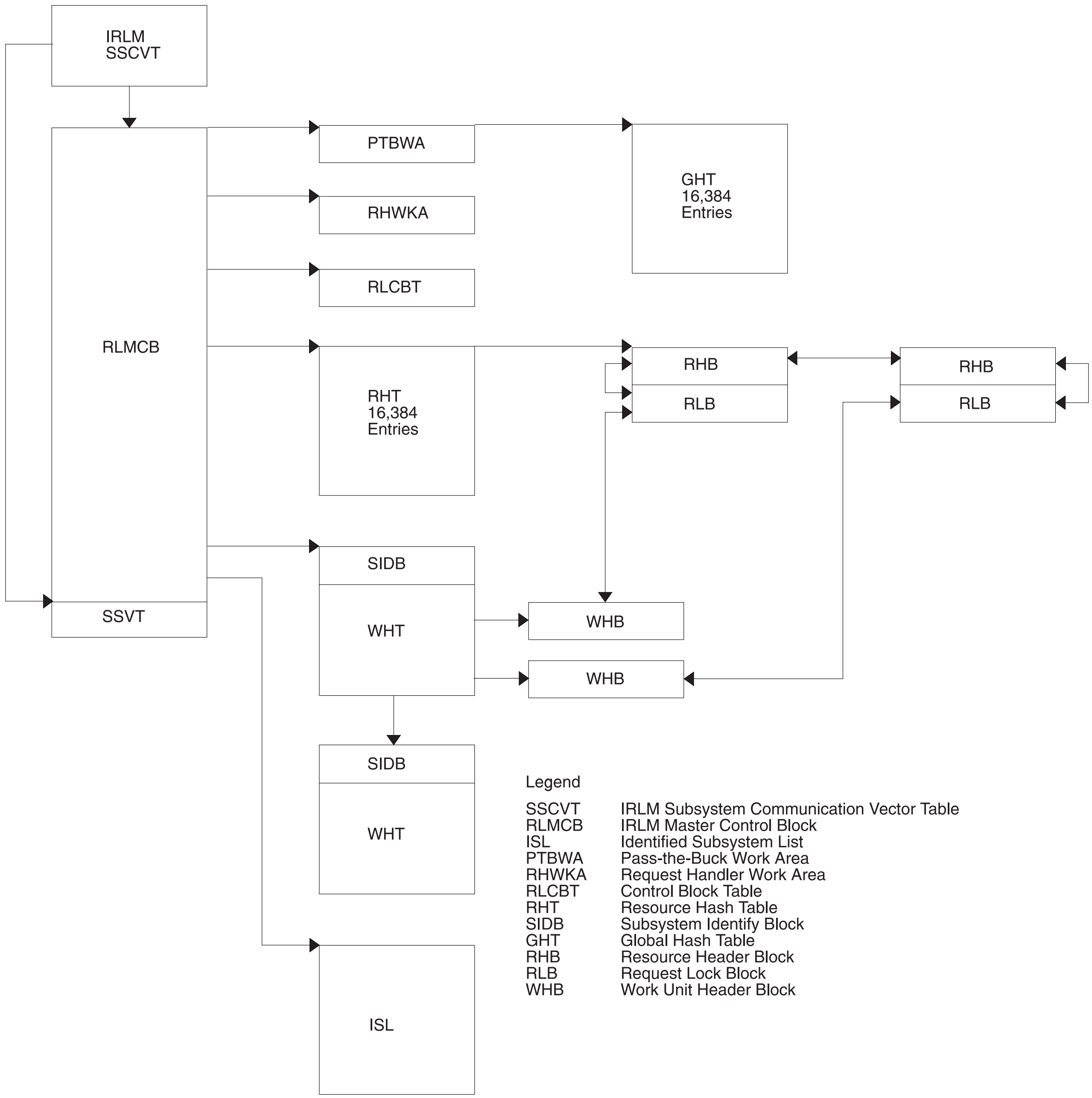


Figure 27. IRLM Overall Control Block Structure

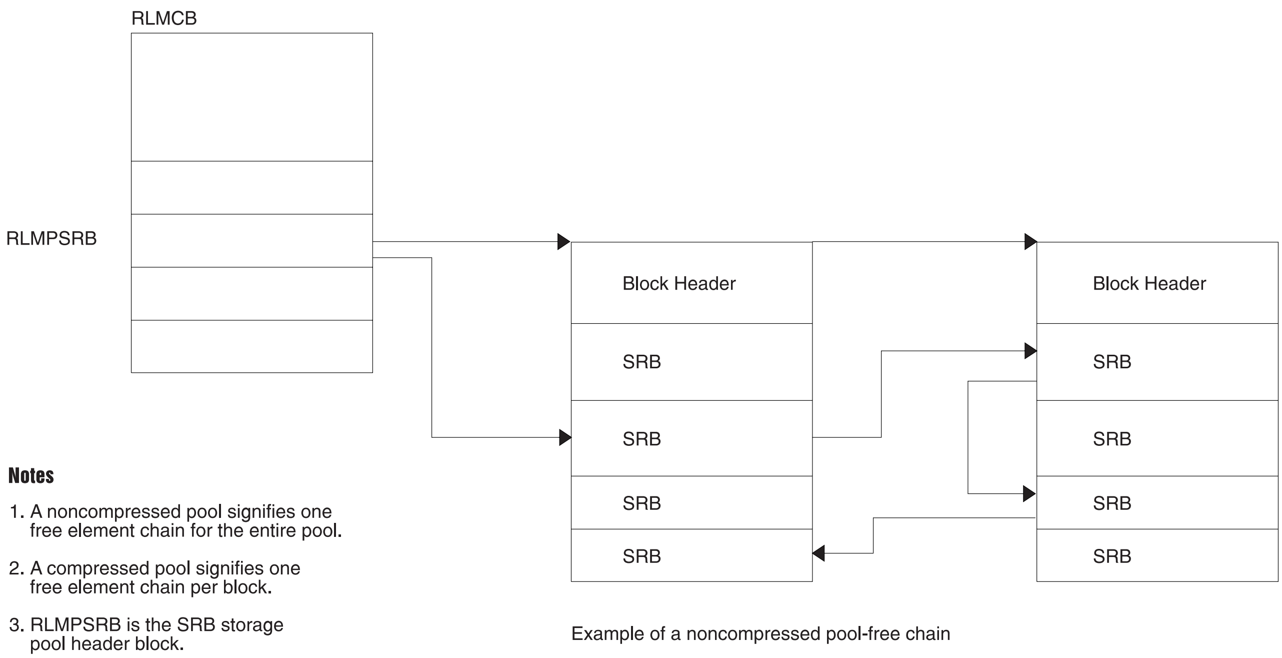


Figure 28. IRLM Storage Manager Pools

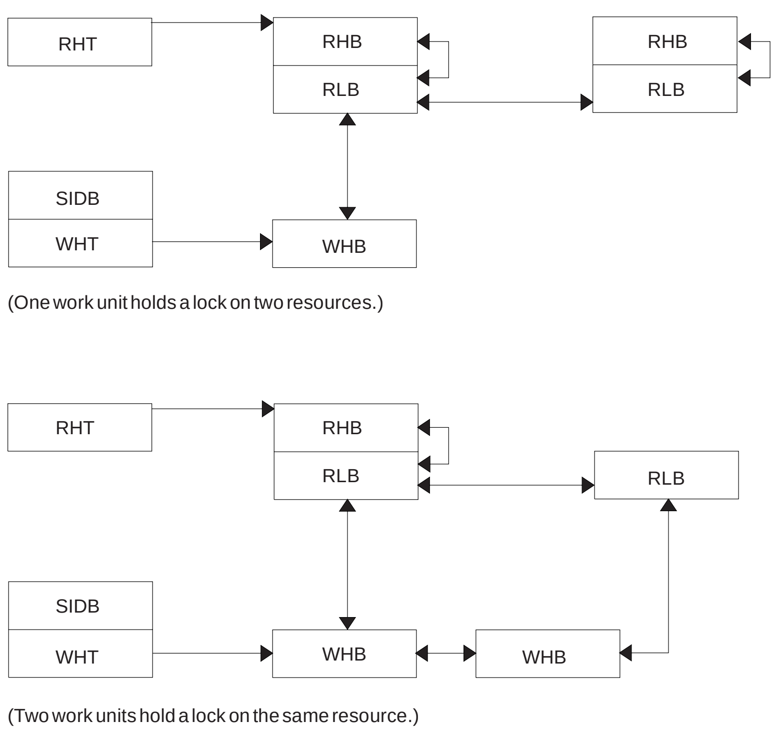


Figure 29. IRLM Lock Request Examples

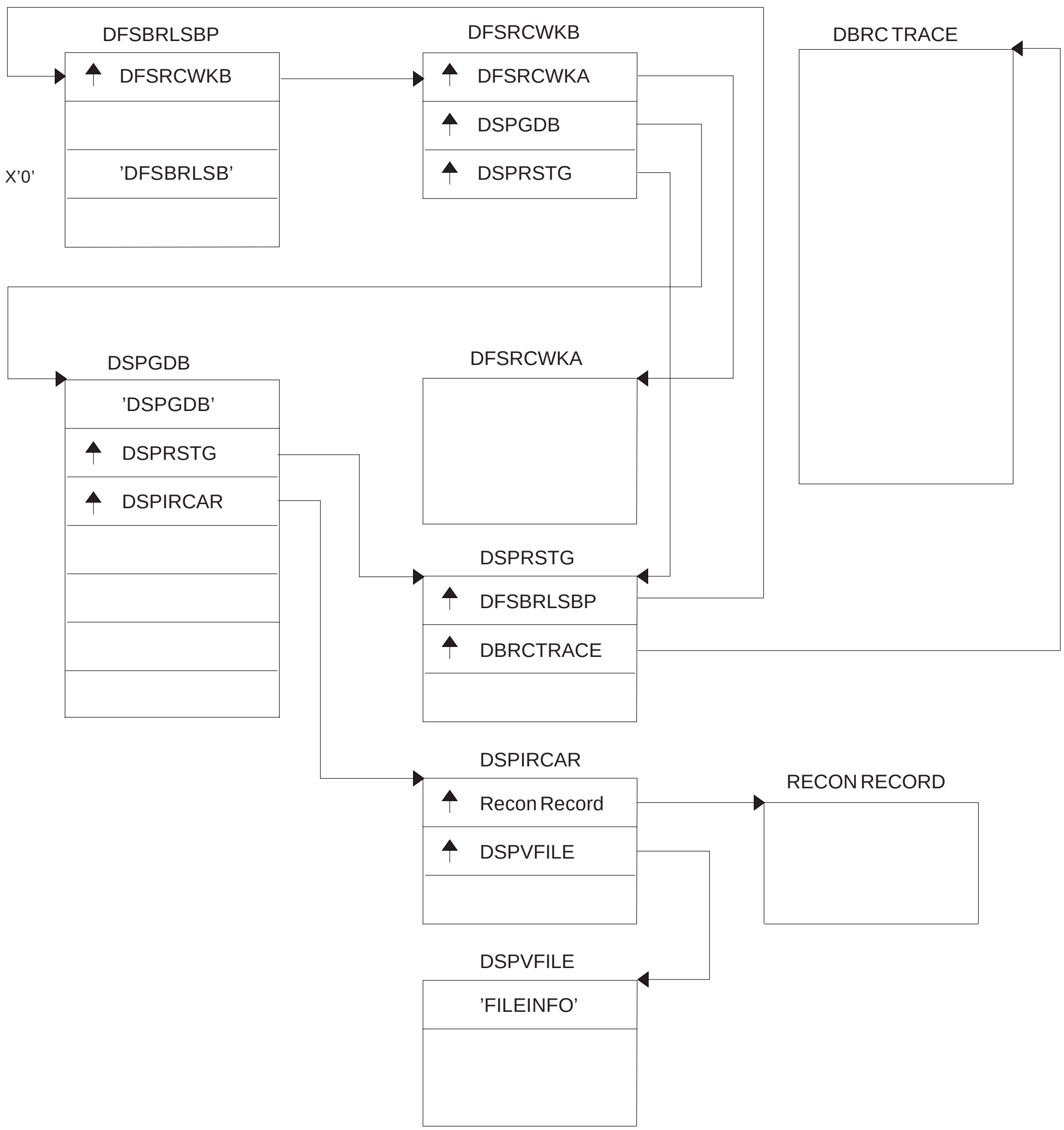


Figure 30. Control Block Overview of Database Recovery Control (DBRC)

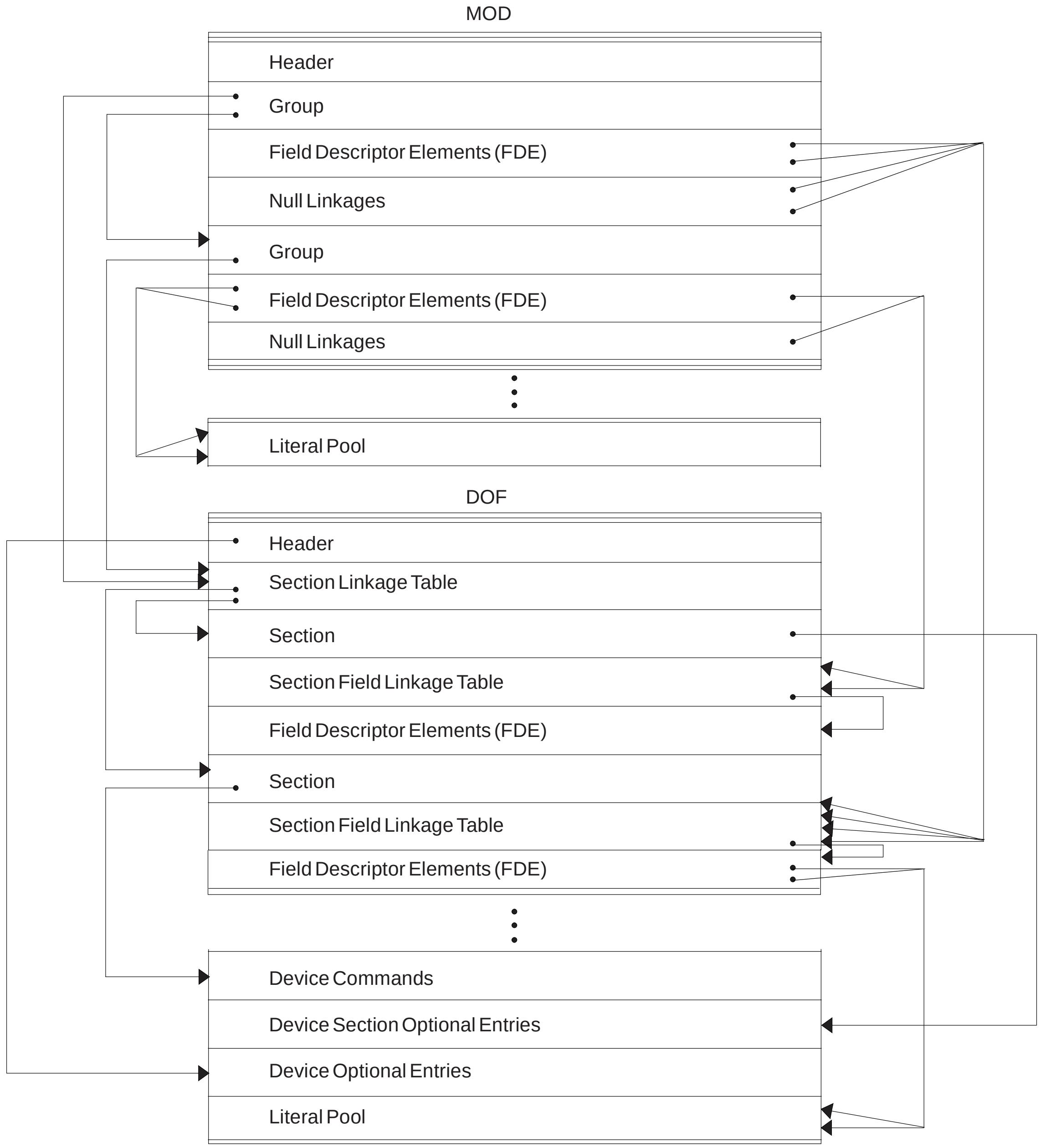


Figure 31. Organization and Basic Linkages: DOF (Device Output Format) and MOD (Message Output Descriptor)

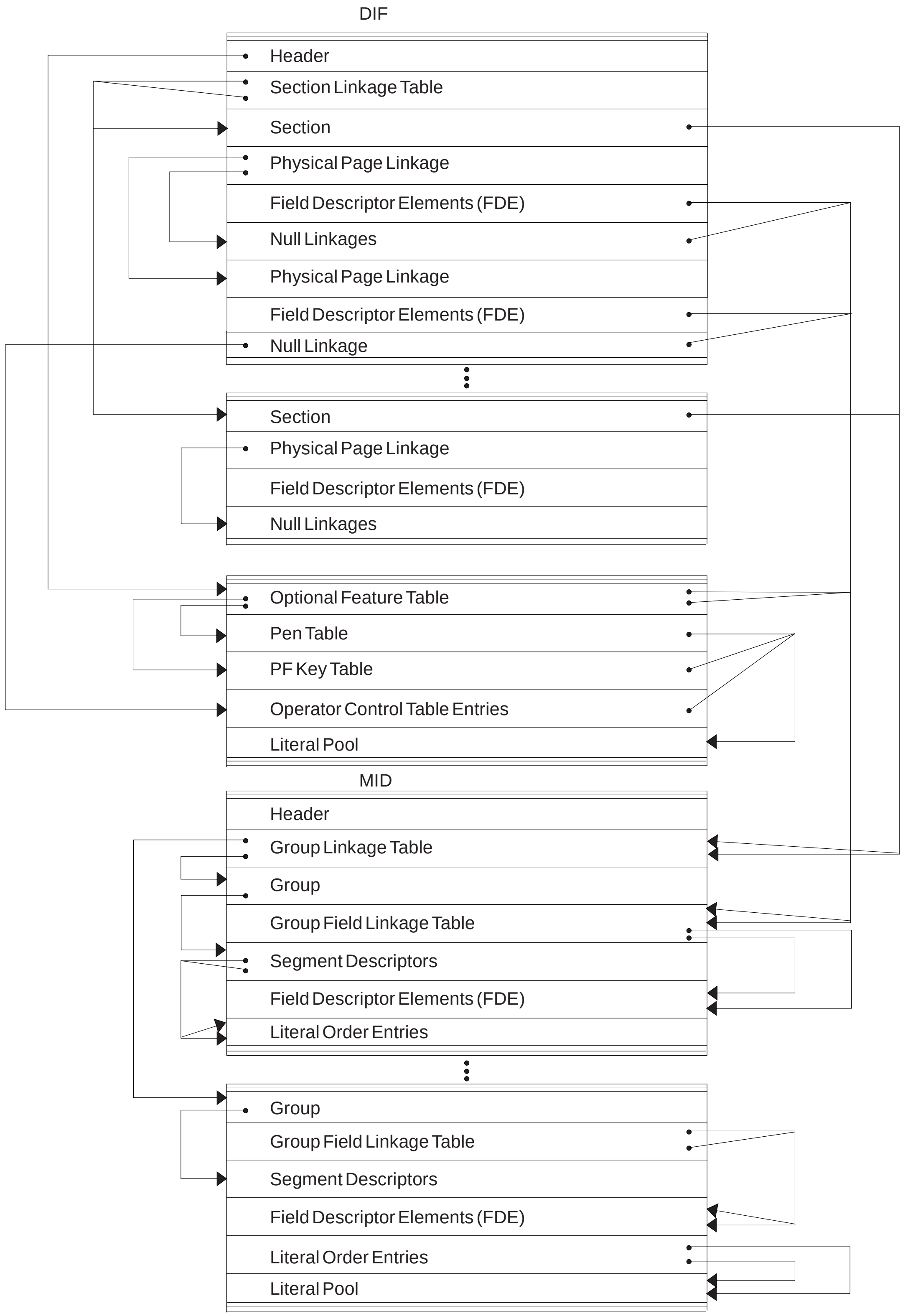


Figure 32. Organization and Basic Linkages: DIF (Device Input Format) and MID (Message Input Descriptor)

Edited Command Format

The edited command buffer is logged in the X'02' log record and is passed to the AOI user exit. You can use the edited command buffer to determine if any recoverable commands were issued for the resource you are analyzing. For example, if you are analyzing a hung terminal problem, look at any log records, including X'02' records, that apply to that terminal.

However, finding the applicable log records might be difficult. If the problem is repeatable, you can use the `/LOG` command to mark the log when certain activities are started or stopped. The `/LOG` command writes a comment to a X'02' log record. This narrows the range of log records you need to examine.

Example: If transaction XYZ results in a hung terminal, use the `/LOG` command to write a comment to a X'02' log record before the transaction is started and after the terminal is hung, as follows:

```
/LOG START XYZ TRAN THAT RESULTED IN HUNG TERMINAL.
/LOG TERMINAL IS NOW HUNG.
```

Look for these comments in the X'02' log record edited command buffers to determine the range of log records to examine.

Figure 33 shows the layout of the edited command.

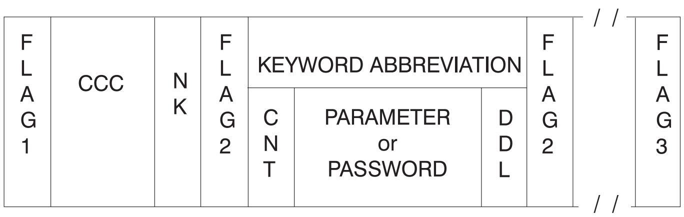


Figure 33. Edited Command Layout

Figure Number
Description

FLAG1
X'FE' to denote the beginning of the edited command. If any parameter contains an error, the command action modules set this byte to X'FC'. An exception is DFSICL40 processing of “ALL” expanded parameters.

CCC First 3 characters of entered command.

NK Hexadecimal value of number of keywords in the condensed buffer.

FLAG2
One of the following:

X'FC' Parameter that follows found in error.

X'FF' 3-byte keyword abbreviation follows.

X'FE' Count (CNT) field and parameter follow.

C'(' Count (CNT) field and password follow.

Keyword Abbreviation
First 3 characters of entered command. Consult DFSCKWDO to obtain the abbreviation; it is sometimes the first 3 characters of any keyword.

CNT Count of number of characters in parameter or password immediately following the CNT. It can be a comma, period, blank, or left parenthesis.

Parameter or Password
Exists exactly as entered from the terminal.

DDL The delimiter entered after the parameter or password. It may be X'80' if the keyword "ALL" was expanded to individual parameters.

FLAG3
Period indicating end of command.

Exception: Only parameter passwords (as in the /IAM command) are present in the condensed buffer; command passwords are not present.

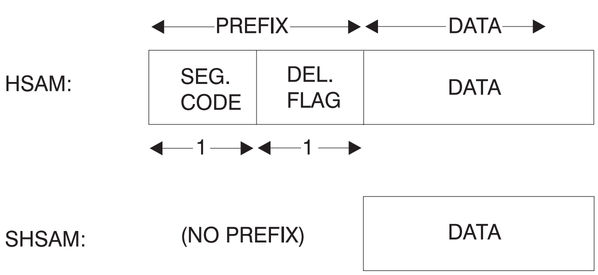
Record Formats

This section describes these DL/I data record formats:

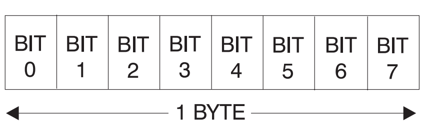
- HSAM and SHSAM database
- HISAM and SHISAM database
- HDAM and HIDAM database
- PHDAM and PHIDAM database
- HIDAM index database
- Secondary index database (VSAM only)
- PSINDEX
- Variable-length segments

HSAM and SHSAM Database

Segment Formats



Delete Byte (Flag) Format

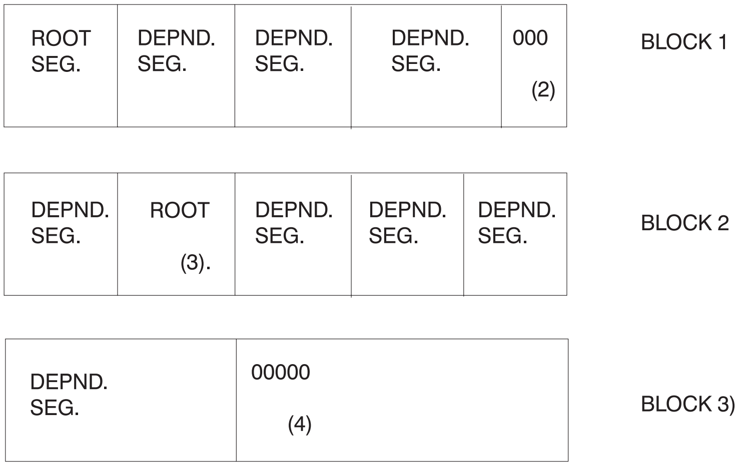


Bit	Description
0	Segment deleted (HISAM).
1	DB record deleted (HISAM).
2	Segment processed by DELETE.
3	Reserved.
4	Data and prefix are separated in storage.
5	Physical segment deleted.

6 Logical segment deleted.

7 Segment space available to be freed; bits 5 and 6 must also be set on.

Block Format for HSAM and SHSAM (1)



Notes:

1. For SHSAM there are no dependent segments. Block size must be a multiple of segment size.

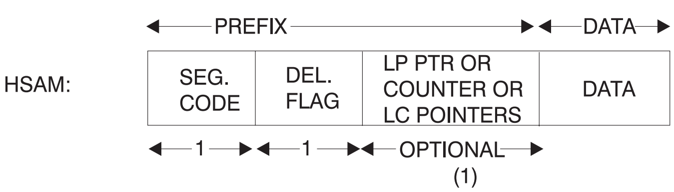
2. Pad with zeros if no room for next segment.

3. Next database record starts immediately.

4. Pad with zeros in last block, after last segment.

HISAM and SHISAM Database

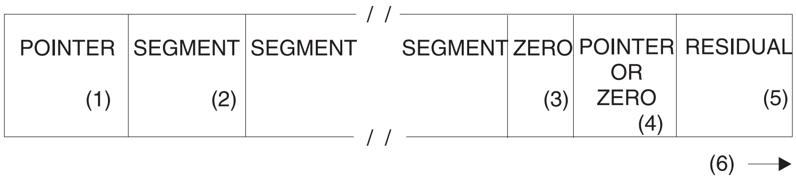
Segment Format



Note:

1. This field can be omitted, or it can be used to hold:
- A 4-byte LP pointer (if this segment is a LC).
- A 4-byte counter (if this segment is a LP).
- One or more 4-byte LC pointers (if this segment is a LP).

LRECL Format



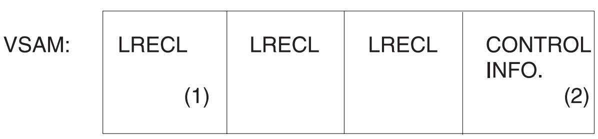
Notes:

1. VSAM: 4-byte RBA of ESDS record containing additional dependent segments for this root occurrence.

   SHISAM: This field is omitted.
2. HISAM: Segment includes prefix and data.

   SHISAM: Segment includes only data (no prefix). (See the preceding "Segment Format".)
3. 1-byte of zeros indicates the end of segments in this LRECL.
4. VSAM: This field is omitted.
5. Space not used.
6. VSAM LRECLs must have an even length.

Block Formats

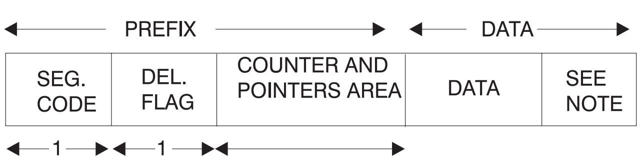


Notes:

1. LRECL length might change between KSDS and ESDS, depending on user definition.
2. Ten bytes if blocked data set; seven bytes if unblocked data set.

HDAM, HIDAM, PHDAM, or PHIDAM Database

Segment Format



In order for all segments to be half-word aligned, a slack byte is added to the end of any segment whose length is an odd number.

Prefix of a Segment

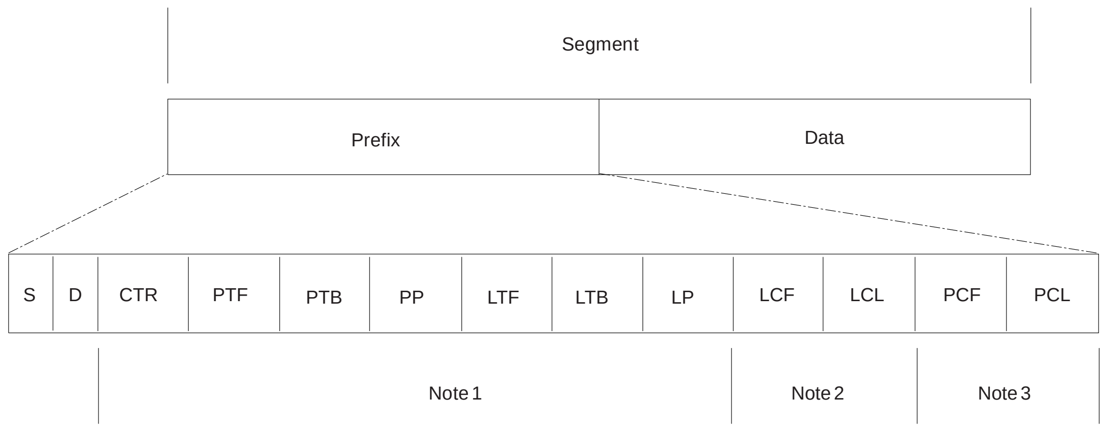


Figure 34. Mapping the Prefix of a Segment

Notes to Figure 34:

1. The pointers that exist in this section of the prefix are identified in the PSDB field DMBPTR (PSDB+7), as shown in the following list:

Prefix Flag	**Prefix Flag Description**
	Segment code (S)
	Delete flag (D)
X'80'	Counter (CTR) for logical relationships
X'40'	Physical twin forward (PTF)
X'20'	Physical twin backward (PTB)
X'10'	Physical parent (PP)
X'08'	Logical twin forward (LTF)
X'04'	Logical twin backward (LTB)
X'02'	Logical parent (LP)
X'01'	Hierarchical direct pointing (If twin-type pointing, this bit is off)

2. How to locate all logical children: logical child first (LCF); logical child last (LCL)

- **a.** At DMBFLAG (PSDB+20), if flag DMBLCEX (X'20') is on, then DMBLST points to a secondary list for this segment. Secondary lists are used for information concerning indexes, logical children, or the logical parents.
- **b.** Secondary list entries whose field DMBSCDE (SEC+0) has flag DMBSLC (X'02') on are descriptions of logical children for a logical parent. Within these secondary lists, the field DMBSLCFL (X'02') has the number of the first and last logical child pointers in the prefix of the logical parent.
- **c.** A logical parent can have multiple types of logical children; thus, there can be more than one logical child secondary list entry for a logical parent. The last secondary list for each segment has the DMBSND flag (X'80') set on in the field DMBSCDE (SEC+0).

3. How to locate all physical children: physical child first (PCF); physical child last (PCL)

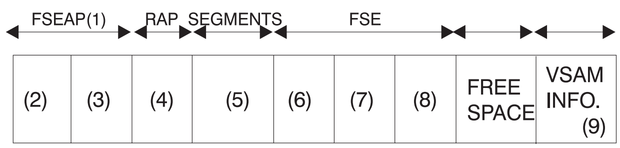
- **a.** Physical child pointers are only present if this segment uses twin-type pointing rather than hierarchic-type pointing. The PSDB entries for the children of the segment being mapped indicate the number of the pointer in their parents' prefix which points to the first and last occurrence of them.

b. The PSDB fields DMBPPFD and DMBPPBK are used for these numbers. The PSDB entries for the children of the segment being mapped can be found by scanning the PSDBs for those whose parent's segment code (PSDB+1) matches the segment code (PSDB+0) of the segment being mapped.

4 An EPS (extended pointer set) that is 28 bytes in length is present in the prefix of an LC segment prefix of a HALDB.

5 An ILK (indirect list entry key) that is 8 bytes in length is present in each segment of a HALDB.

OSAM and VSAM ESDS Block Format



Notes:

1. Free space element anchor point.
2. 2-byte offset to first free space element; contains zeros in a bit map block.
3. 2-byte length (see 7); value is zero.
4. 4-byte root anchor point (RAP). The number per block is specified in DBDGEN, except if HIDAM with TF (and not TB) is pointing at root level, one anchor point per block is provided and it heads a LIFO chain of roots inserted in that block. If HIDAM OSAM with TF and TB or no TF or TB is pointing at root level, there are no anchor points provided.
5. User database segments (prefix and data). In a bit map block, the bit map starts here and extends to the end of the block or to the VSAM control information.
6. 2-byte offset to next free space element (FSE) from start of block.
7. 2-byte length of free space, including 8-byte FSE.
8. 2-byte identification of task that freed this space.
9. 7 bytes of VSAM control data; omitted for OSAM.

This format applies at the conclusion of initial load. The subsequent deletion of segments can result in free space elements that alternate with user database segments.

VSAM LRECL Format

On Storage Device and in Buffer Pool

Note:

1. Four-byte RBA pointer to VSAM database root segment whose key value is the same as the value in the next field of this segment.

As Returned by Buffer Handler

(1)	PTR (2)	SEG. CODE	DEL. FLAG	PTR (1)	ROOT KEY VALUE

Notes:

1. Same as buffer pool format, except for pointer and segment code in front.
2. Four-byte pointer with value of zero.

VSAM Block Format on Device and in Buffer Pool

LRECL	LRECL	LRECL	VSAM INFO.

Secondary Index or PSINDEX Database (VSAM Only)

LRECL Format on Device and in Buffer Pool

One segment per LRECL.

PREFIX		DATA			
PTR (1)	DEL. FLAG	PTR (2)	EPS (3)	RKEY (4)	INDEX DATA (See segment data format)
4	1	4	28	1-255	

Notes:

1. Nonunique keys: This points to ESDS LRECL with the same key value. Unique keys: This field is omitted.
2. Direct pointer to index target segment. Omit this field if indirect pointing is used or if this is a HALDB PSINDEX.
3 The EPS is present only if this is a HALDB PSINDEX. The 4-byte pointer to the target segment is included in the EPS.
4 The RKEY field is present only if this is a HALDB PSINDEX. This is the key value for the root of the target segment and its length can be from 1 to 255 bytes.

LRECL as Returned by Buffer Handler

PTR (1)	SEG. CODE (2)	DEL. FLAG	PTR (3)	EPS (4)	RKEY (5)	INDEX DATA
4	1	1	4	28	1-255	

Notes:

1. Four-byte pointer contains zeros.
2. Code value is 01.
3. Direct pointer to index target segment. Omit this field if indirect pointing is used or if this is a HALDB PSINDEX.

4 The EPS is present only if this is a HALDB PSINDEX. The 4-byte pointer to the target segment is included in the EPS.

5 The RKEY field is present only if this is a HALDB PSINDEX. This is the key value for the root of the target segment and its length can be from 1 to 255 bytes.

Block Format on Device and in Buffer Pool

LRECL	LRECL	LRECL	VSAM INFO.
			◄— 10 —►

Segment Data Format

CONSTANT (Optional)	SEARCH FIELD	SUBSEQUENCE FIELD (Optional)	DUPLICATE DATA (Optional)	CONCAT. KEY (Optional)	USER DATA (Optional)

Variable-Length Segments

HISAM, HDAM, HIDAM, PHDAM, and PHIDAM Segment Format

◄— PREFIX		—►	◄— DATA	—►
SEG. CODE	DEL. FLAG	CNR. PNTRS	SIZE FIELD	OTHER DATA
			◄— 2 —►	

Note: Variable-length segment must have a 2-byte length field at the front of the DATA portion.

HDAM, HIDAM, PHDAM, and PHIDAM

When prefix and data are separated.

SEG. CODE	DEL. FLAG (1)	CNR. PNTRS	DATA PNTR (2)	FREE SPACE

◄— PREFIX	—►	◄— DATA	—►
SEG. CODE	DEL. FLAG	SIZE FIELD	OTHER DATA

Notes:

1. DEL FLAG containing X'08' indicates that the data has been separated from the prefix.
2. DATA POINTER is a direct pointer to the segment containing the “other data”.

Part 3. Diagnostic Aids

Chapter 7. SYS—System Service Aids

This chapter provides diagnostic hints and describes the service aids that can help you analyze IMS system problems. This chapter describes:

- The log records, their formats, and the modules that issue them
- The File Select and Formatting Print utility which prints various log records from the IMS log data set
- The Offline Dump Formatter
- The SNAP call facility
- The common trace table interface

Log Records

To diagnose some problems, you need to examine the content of log records in order to determine what was going on in the system prior to the problem. By knowing the layout of the log records, you can set up a DFSERA10 job that will produce the specific log records you need to examine.

In addition, the content of the log records frequently contains information that you can use in your keyword string or when reviewing existing APAR descriptions and comparing them to your own situation.

To view the log records you can assemble log records mapping macro ILOGREC. For Fast Path log record formats, you can assemble mapping macros DBFLSRT, DBFLGRQ, DBFLGRIM, DBFLGROM, DBFLGRSD, DBFLGSYN, and DBFBMSDB.

Table 6 lists each log record and:

- The DSECT that creates the record
- The conditions that cause the record to be created
- The module that issues the record

Table 6. IMS Log Records Used to Analyze IMS Problems

Type	DSECT Name	Why Written (Issuing Module)
X'01'	QLOGMSGP	Data was put in a message queue buffer. Caller is data communication. (DFSQLOG0)
X'02'	CMLOG	A /LOG command or a command that alters data required for restart was successfully completed. (DFSICLP0)
X'03'	QLOGMSGP	Data was put in a message queue buffer. Caller is DL/I. (DFSQLOG0)
X'06'	ACLOGREC	IMS was started or stopped, or FEOV was issued. The VTAM TPEND exit was entered or the IRLM failed in an IMS/XRF complex. A /SWITCH command was processed in an IMS/XRF complex. A /START command connected IMS to VTAM. Data sharing capability was quiesced. (DFSFLLG0, DFSFDLM0, DFSICA20, DFSICL, DFSRDSH0)
X'07'	DLREC	An application program terminated. (DFSRBLB0, DFSRBOI0, DFSSABN0, DFSDABN0, DFSDLA30, DFSTMAD0)
X'08'	LINTD	An application program was scheduled. (DFSSMSC0, DFSSBMP0, DFSDASP0, DFSDLA30, DFSTMAD0)
X'09'	SBLOGREC	An application potentially using sequential buffering terminated. The following subcodes, contained within the log record, identify the type of statistics written in the log record. (DFSSBTD0) **X'01'** Sequential buffering summary statistic for the PST. **X'02'** Sequential buffering detailed statistics for each SDSG.
X'0A07'	S0AREC	A CPI communications driven application program terminated. (DFSSABN0)
X'0A08'	S0AREC	A CPI communications driven application program was scheduled. (DFSSMSC0)
X'10'	SCREC	A security violation occurred. (DFSICIO0, DFSCMD30, DFSICLZ0, DFSTMAD0)

Table 6. IMS Log Records Used to Analyze IMS Problems (continued)

Type	DSECT Name	Why Written (Issuing Module)
X'11'	LCONVERS	A conversational program started. (DFSCON00)
X'12'	LCONVERS	A conversational program terminated. (DFSCON20)
X'14'	LNREC	A dial line was disconnected. (DFSICIO0, DFSICLA0)
X'15'	LNREC	A dial line was connected. (DFSICA10)
X'16'	LOG16	A /SIGN command successfully completed. (DFSICLZ0, DFSCBDL0)
X'18'	XLOG18	A user program established intent to use extended checkpoint and then issued a CHKP call. The user program issued a CHKP by issuing an XRST call with eight blank characters as a checkpoint ID value. (DFSZSC00)
X'20'	ILRDOC	A database was opened. (DFSDLOC0)
X'21'	ILRDOC	A database was closed. (DFSDLOC0)
X'24'	ERLGDSCT	The buffer handler detected an I/O error. (DFSDVSM0, DBFMER00)
X'25'	EEQLOG	An EEQE was created or deleted. (DFSTOLG0)
X'26'	IOTBUF	An I/O toleration buffer was created. (DFSTOLG0)
X'27'	DBXLOG	A data set was extended, according to these subcodes: **X'01'** Data set extend phase 1. (DFSDVSM0) **X'02'** Data set extend phase 2. (DFSDBHI0)
X'28'	PH1DC	The IMS restart facility updated the sequence numbers of input messages for response mode non-Fast Path transactions from STSN devices. (DFSFXC40)
X'30'	QLOGMSG1	A message prefix was changed. (DFSQLOG0)
X'31'	QLOGGETU	A GET UNIQUE was issued for a message. (DFSQLOG0)
X'32'	QLOGREJE	A message was rejected. It was presumed to have been the cause of an application program ABEND. (DFSQLOG0)
X'33'	QLOGFREE	The queue manager released a record. (DFSQLOG0)
X'34'	QLOGCANC	A message was canceled. (DFSQLOG0)
X'35'	QLOGENQU	A message was enqueued or re-enqueued. (DFSQLOG0)
X'36'	QLOGDEQS	A message was dequeued or saved or deleted. (DFSQLOG0)
| X'37'	DFSXFER QLOGXFER	Records marked as NO INPUT and NO OUTPUT are written by the sync point coordinator when all resource managers have completed Phase 1. (DFSFXC30) Records marked as NO INPUT and NO OUTPUT (for example, X'3730') are also written by the DBCTL sync point processor after receiving a phase 2 commit request. (DFSDSC00) Phase 2 DC processing. One output message was transferred for each message on the PST temporary output queue. All subsequent X'37' input/output messages are written by the QMGR. (DFSQLOG0, DBFSLOG0, DFSFXC30)
X'38'	QLOGRELI	An input message was put back on the input queue when the application abnormally terminated. (DFSQLOG0) Records marked as “Release with no input message” (for example, X'3801') are written by the DBCTL sync point processor (DFSDSC00) after receiving an abort request.
X'39'	QLOGRELO	The output queue was freed during cleanup processing of a RELEASE call. (DFSQLOG0)
X'3A'	QLFXFREE	A bitmap record was replaced after a queue record was freed at the end of DFSQFIX0 processing. (DFSQFIX0)
X'3B'	QLFXRERR	An invalid message record or a nonrecoverable message response was detected during queue validation. (DFSQFIX0)
X'3C'	QLFXBERR	A control block was changed during validation by DFSQFIX0. (DFSQIX0)
X'3D'	QLFXQBLK	A QBLK record was altered during DFSQFIX0 processing. (DFSQFIX0)

Table 6. IMS Log Records Used to Analyze IMS Problems (continued)

Type	DSECT Name	Why Written (Issuing Module)
X'40'	LOG01	A checkpoint was taken. The following subcodes, contained within the log record, precede and identify each type of information written in the log record. **X'01'** Checkpoint information begins here. (DFSRCP00) **X'02'** Message queue checkpoint record. (DFSQCP00) **X'03'** CNTs and/or LNTs follow. (DFSRCP30) **X'04'** SMBs follow. (DFSRCP30) **X'05'** Non-VTAM CTBs follow. (DFSRCP30) **X'06'** DMBs follow. (DFSRCP40) **X'07'** PSB follows. (DFSRCP40) **X'08'** Non-VTAM CLB, LLB, or both follow. (DFSRCP30) **X'09'** Password table and SMUPs follow. (DFSRCP30) **X'0A'** Password matrix follows. (DFSRCP30) **X'0B'** CTM matrix follows. (DFSRCP30) **X'0C'** CVB follows. (DFSRCP30) **X'0D'** CCBs follow. (DFSRCP30) **X'0F'** Message queues TTR and LCB follow. (DFSRCP30) **X'10'** Non-VTAM CRBs follow. (DFSRCP30) **X'14'** SPQBs and related CNTs follow. (DFSRCP30) **X'20'** Non-VTAM CIBs follow. (DFSRCP30) **X'21'** VTAM VTCBs follow. (DFSRCP30) **X'22'** Subcode for Queue Anchor Block (QAB) (DFS6CKP0) **X'23'** Subcode for LU 6.2 descriptors modified by /CHANGE Descriptor command. (DFS6CKP0) **X'25'** EEQE follows. (DFSTOLG0) **X'26'** I/O toleration buffer follows. (DFSTOLG0) **X'27'** Contains database updates for an in-doubt unit of recovery (DFSRCP40) **X'28'** Error queue elements (EQEL) for recovery in-doubt structure (RIS) (DFSRCP40) **X'30'** RREs follow. (DFSRCP50) **X'31'** SIDXs follow. (DFSRCP50) | **X'32'** TPIPE/YQAB follow. (DFSYCKP0) | **X'33'** MTE follow. (DFSYCKP0) | **X'34'** TIB follow. (DFSYCKP0) **X'40'** UOWEs follow. (DFSRCP30) **X'70'** MSDB record follows. (DBFHDMP0) **X'71'** ECNT follows. (DBFHDMP0)

Table 6. IMS Log Records Used to Analyze IMS Problems (continued)

Type	DSECT Name	Why Written (Issuing Module)
X'40' (cont'd)	LOG01	**X'72'** MSDB header follows. (DBFHDMP0) **X'73'** Pagefixed MSDBs follow. (DBFHDMP0) **X'74'** Pageable MSDBs follow. (DBFHDMP0) **X'79'** MSDB record ends. (DBFHDMP0) **X'80'** Fast Path checkpoint information begins here. (DBFCHKP0) **X'82'** EMHB follows. (DBFCHKP0) **X'83'** RCTE follows. (DBFCHKP0) **X'84'** DMCB and DMAC follow. (DBFCHKP0) **X'85'** MTO buffer follows. (DBFCHKP0) **X'86'** DMHR and DEDB buffers follow. (DBFCHKP0) **X'87'** ADSC follows. (DBFCHKP0) **X'88'** Fast Path IEEQEs. (DBFCHKP0) **X'89'** Fast Path checkpoint information ends here. (DBFCHKP0) **X'98'** Checkpoint information ends here. (DFSRCP10) **X'99'** The message queue checkpoint information ends here. (DFSQCP00)
X'41'	LOG41DSC	A batch program or BMP program issued a checkpoint. (DFSRDBL0)
X'42'	ATLOGREC	IMS switched from one OLDS to another, or a checkpoint was taken, or a shutdown checkpoint was taken. (DFSFDLS0, DFSRDS00, DFSRCP00)
X'43'	ADSETLOG	The log manager or the log archive utility created this log record. The following subcodes identify each type of record: **X'01'** Record contains status of current online log data set. (DFSFDLS0) **X'02'** Dummy record created by log archive utility. This record is created as a substitute for a record that is omitted because of control statement specifications. (DFSUARP0)

Table 6. IMS Log Records Used to Analyze IMS Problems (continued)

Type	DSECT Name	Why Written (Issuing Module)
X'45'	STLOGREC	Checkpoint statistics were gathered. The following subcodes within the log record mark the start of various types of statistics written in the log record (DFSSTAT0). **X'01'** Dynamic database log statistics. **X'02'** Queue buffer statistics. **X'03'** Format pool statistics. **X'04'** DL/I buffer pool statistics. **X'05'** Variable storage pool statistics. **X'06'** Application scheduling statistics. **X'07'** Logging statistics. **X'08'** VSAM buffer pool statistics. **X'09'** Program isolation statistics. **X'10'** RCF multi-TCB statistics. **X'0A'** Latch management statistics. **X'0B'** Selected dispatcher statistics. **X'0C'** Storage pool statistics. (DFSCBT00) **X'0D'** Receive Any Buffer (RECA) statistics. **X'0E'** Fixed storage pool usage statistic. **X'0F'** Dispatcher statistics. **X'10'** RCF Multi-TCB statistics. **X'21'** IRLM subsystem statistics. (DXRRSTAT) **X'22'** IRLM system statistics. (DXRRSTAT) **X'FF'** End of statistics records.
X'47'	CAPLOG	A checkpoint was just taken. This log record contains all the PSTs that were in the system. (DFSRCP10)
X'48'	PALOGREC	This is a variable-length padding log record. A X'48' log record at the end of a block contains log block descriptive information. (DFSFLLG0) **X'00'** OLDS padding X'48' record. **X'01'** X'4301' record space holder. **X'02'** Archived OLDS X'48' record. **X'03'** Batch SLDS padding X'48' record. **X'04'** Archived batch SLDS X'48' record.

Table 6. IMS Log Records Used to Analyze IMS Problems (continued)

Type	DSECT Name	Why Written (Issuing Module)
X'49'	DFSLOG49	This log record is written by the log router and the full-function database tracker at the RSR tracking site when an updated block has an invalid free space element (FSE) or free space element anchor point (FSEAP). **X'00'** Definition. **X'01'** Begin stream record. **X'02'** Begin OFR record. **X'03'** OFR milestone record. **X'04'** Log truncation start record. **X'05'** XRC tracking record. **X'06'** Data set services create data set record. **X'07'** Takeover record. **X'08'** Auto Archive Init Request record. **X'0A'** Last LSN of prilog record. **X'0B'** Data set sequence number record. **X'0C'** Open data set record. **X'0D'** DBRC hash table state record. **X'0E'** FF DB Tracker Update Sequence Number (USN). **X'20'** FP DB Tracker statistics record. **X'30'** FF DB Tracker FSE Error record. **X'31'** FF DB Tracker statistics record. **X'50'** OFR Stream Processing Time.
X'4C'	STDBLOG	Activity related to database processing, according to these subcodes: **X'01'** A backout for token was done. (DFSRBOI0) **X'02'** A backout error occurred. (DFSRBOI0) **X'04'** First update flag was reset. (DFSDBDR0) **X'08'** A share level or held state was changed. (DFSDBAU0, DFSDLOC0) **X'10'** A write error occurred. (DFSDBH40, DFSDVSM0) **X'20'** A program was stopped. (DFSRBOI0) **X'40'** A database was started. (DFSDBDR0) **X'80'** A database was stopped. (DFSDBDR0) **X'82'** A database backout failure occurred. (DFSRESP0)
X'4E'	SLOG	An event occurred during monitoring. This record is in the monitor log and contains statistical information about the system. (DFSMNTR0)
X'50'	DBLOG	The database was updated. This log record contains the new data on an insert and update call as well as the old data and FSE updates on a delete call. (DFSRDBL0) **X'52'** IMS is about to do an ISRT operation for a new root in a key sequence data set. This record contains a copy of the data before it was changed. (DFSRDBL0)
X'53'	SPLLOG	Bitmap write done for log record for alternate IMS tracking CI split on active IMS. (DFSRCHB0, DFSGGSP0, DFSFRSP0, DFSDVSM0)
X'55'	DFSETPCP	Record reserved for external subsystem information. (DFSESS30)

Table 6. IMS Log Records Used to Analyze IMS Problems (continued)

Type	DSECT Name	Why Written (Issuing Module)
X'56'	DFSETPCP	IMS external subsystem support recovery log record ID. The following subcodes, contained within the record, precede information in the log record. X'56' records are written by three IMS components. These components can represent the status of IMS external subsystem transactions, the status of the connection between IMS and CCTL, or the stages of IMS sync point processing. The subcodes listed below represent the X'56' record components and their purposes. They are contained in the record and precede data in the log record. **X'000001'** IMS began the commit process. (DFSESP10) **X'000002'** IMS finished the commit process. (DFSESP20) **X'000003'** IMS signed on to an external subsystem. (DFSESSO0) **X'000004'** IMS created a thread for an external subsystem. (DFSESCT0) **X'000005'** IMS resolved a RID. (DFSESI60) **X'000006'** An IMS dependent region abended. (DFSFESP0) **X'000007'** IMS deleted a residual recovery element (RRE) through the /CHA command. (DFSESI70) **X'000008'** IMS deleted a residual recovery element (RRE) by a restart or start command. (DFSIESI0) **X'000009'** An external subsystem disconnected. (DFSESI30) **X'00000A'** Commit found no work to do. **X'08'** A CCTL connected to DBCTL. (DFSDASI0) Mapping macro is DFSETPCP. **X'09'** A CCTL disconnected from DBCTL. (DFSDASD0) Mapping macro is DFSGTPCP. **X'10'** Phase 1 commit processing started. (DFSDSC00, DFSTMS00) **X'11'** Phase 1 commit processing ended. (DFSDSC00, DFSTMS00) **X'12'** Phase 2 commit processing ended. (DFSDSC00, DFSFXC30, DFSSLOG0, DFSSMSC0, DFSTMS00) **X'13'** Recoverable in-doubt structure (RIS) created. (DFSDRIS0) **X'14'** Recoverable in-doubt structure (RIS) deleted. (DFSDRID0) **X'15'** IMS restarted with RRS. (DFSRRSI0) **X'16'** Interest has been registered with RRS for this UOW. (DFSRRSI0) **X'37'** Phase 2 commit processing started by a resynchronization request. (DFSDRID0) **X'38'** Phase 2 abort processing started by a resynchronization request. (DFSDRID0)

Table 6. IMS Log Records Used to Analyze IMS Problems (continued)

Type	DSECT Name	Why Written (Issuing Module)
X'57'	DFSDBUR	Database updates in an RSR environment: **X'01'** Begin database update. (DFSRDBL0) **X'02'** End database update. (DFSRDBL0)
X'59'		**Mapping Macro** This is a Fast Path log record. The subcodes that follow, are contained within the record, and precede information in the log record:
	FLIM	**DBFLGRIM X'01'** An input message was received. (DBFSHSP0)
	FLOM	**DBFLGROM X'03'** An output message was sent. (DBFSHSP0)
	DBFL59X	**DBFL59X X'10'** I/O from a data space has started (DBFVXOC0, DBFVOCI0) **DBFL59X X'12'** A group of C/Is (control intervals) from a data space has been hardened to DASD (DBFVXOC0, DBFVOCI0, DBFERS21)
	DBFSQRIM	**DBFSQRIM X'11'** An input message was inserted on an EMHQ structure. (DBFHIEL0, DBFSYN20)
	DBFSQROM	**DBFSQROM X'16'** An output message was inserted on an EMHQ structure. (DBFATRM0, DBFHCTR0, DBFHCAS0, DBFERMG0, DBFSYN20)
	MSUPLOG	**DBFBMSDB X'20'** An MSDB was updated. (DBFSLOG0, DBFBMSDB)
	DOCL	**DBFDOCL X'21'** DEDB area data set was opened. (DBFMOCL0)
	DOCL	**DBFDOCL X'22'** X'22' DEDB area data set was closed. (DBFMOCL0)
	DOCL	**DBFDOCL X'23'** DEDB area data set status was changed. (DBFMOCL0)
	EQE	**DBFEQE** X'24' An ADS error queue element (EQE) was created. (DBFMEQE0)
	FLDQ	**DBFLGRDQ X'36'** An output message was dequeued. This log record also contains information that is necessary to run the Fast Path Log Analysis utility in a shared EMH environment. (DBFHQMI0, DBFHTMG0)
	SYNC	**DBFLGSYN X'37'** A synchronization point operation completed. (DBFSLG20)
	SYNC	**DBFLGSYN X'38'** A synchronization point operation was unsuccessful. (DBFSLG20)
	HICL5947	**DBFLGRIC X'47'** Contains a bitmap of CIs that have updates in an HSSP image copy data set. (DBFSLGE1)
	LSRT	**DBFLSRT X'50'** A DEDB was updated—DMAC status log record for DMACOCNT or DMACNXTS. (DBFSLOG0, DBFARDB0, DBFMLOP0)

Table 6. IMS Log Records Used to Analyze IMS Problems (continued)

Type	DSECT Name	Why Written (Issuing Module)
	LSRT	**DBFLSRT X'53'** An online utility updated a DEDB. (DBFUMAL0, DBFUMAI0)
	LSRT	**DBFLSRT X'54'** A log record was created each time an area containing sequential dependent buffers was opened. (DBFMLOG0)
	FLSD	**DBFLGRSD X'55'** A new buffer for sequential dependent segments was obtained. (DBFSYP20)
	LSRT	**DBFLSRT X'56'** Indoubt SDEP buffer from the resynchronization process. (DBFMLOG0) (DBFSYP20)
	LSRT	**DBFLSRT X'57'** Local/Global portion of DMAC logged. (DBFARDB0, DBFUMAL0)
	L56X	**DBFL56X X'58'** An SDEP buffer was successfully written. (DBFSYP20)
	FLRE	**DBFLGRRE X'70'** The MSDB relocation factor for XRF is shown. (DFSRLP00)
X'5E'	SBLI	Sequential buffer image capture record. A sequential buffer-handler function has been called, according to these subcodes (DFSSBIC0): **X'00'** Application start record. **X'04'** Search/Read. **X'0C'** OSAM buffer-handler crossed a buffer boundary. **X'18'** New logical position. **X'1C'** Application stop record.
X'5F'	DLTRLOGR	A DL/I call was completed. This record contains DL/I call image capture trace data. (DFSDDLT0)
X'63'	S3REC63	Log session initiation and termination. When X'02' is on in the second byte, the X'63' record represents only the deletion of a VTCB. (DFSCVLG0)
X'64'	SMREC	An inconsistency was found in processing associated with MSC. (DFSCMS00)
X'65'	SSREC	A message is about to be enqueued (applicable for System/3 and System/7 only). (DFSCRSV0)
X'66'	SXREC	A message is about to be enqueued or dequeued (applicable for 3614, FINANCE, and SLU P nodes, MSC links, or ISC sessions). (DFSCVFD0, DFSCVFI0, DFSCVFN0, DFSCVLG0, DFSCMSV0, DFSCMSF0)

Table 6. IMS Log Records Used to Analyze IMS Problems (continued)

Type	DSECT Name	Why Written (Issuing Module)
X'67'	CTREC	This log record is a service trace record (see Figure 35 on page 125 for log record physical layout). The following subcodes, contained within it, identify what conditions caused a particular part of the log record to be written:
		X'01' There are three situations in which X'6701' is written: • A /TRACE command was issued. This record can also indicate that error blocks were written unconditionally by device-dependent code when a major error condition was detected. (Applicable to System/3 and System/7, MSC, and VTAM.) (DFSCFEZ0) • Errors were detected in AOI module DFSAOUE0. • Errors were detected in AOI module DFSAOE00.
		X'03' A 3270 error was detected. More information about this condition is contained in “Terminal Communication Task Trace” on page 251. (DFSCFEZ0)
		X'04' An IMS notification exit failed to obtain an AWE for restart processing. IMS was unable to post the deferred unit of recovery with RRS/MVS.
		X'06' An I/O error occurred on a Fast Path area data set. The record prefix format is the same as the X'6701' type. The contents of the data portion is the DMHR associated with the I/O error.
X'67'		**X'05'** A thread terminated abnormally. The data portion of the log record contains diagnostic information for dependent regions. All blocks logged have eye-catchers preceding them. Normal IMS DSECTs map the logged information. (DFSASK00, DFSDTTA0, DFSSDA20)
X'67'	DFSL6740	**X'40'** This log record represents an IMS UOW that was placed on the Common Queue Server’s (CQS) cold queue because CQS found UOWs on its private queues on a cold start of either TM (COLDSYS or COLDCOM) or CQS. CQS moves these UOWs to the CQS cold queue and passes the UOW values to IMS. IMS logs these UOWs in the type X'6740' log record for audit purposes. The customer can then process these log records to determine what action to take for these UOWs. (DFSSQ030, DBFSQ030)

Table 6. IMS Log Records Used to Analyze IMS Problems (continued)

Type	DSECT Name	Why Written (Issuing Module)
X'67'	DFS67D0	**X'D0'** Indicates the diagnostic record of a failed service request. **X'01'** Failure during a DB DL/I call. **X'02'** Failure during a DC DL/I call. (DFSCPY00, DFSDLA30, DBFHGU10, DFSTMAP0) **X'03'** Failure during a SYS DL/I call. **X'04'** An exit failure occurred. (DFSRRSI0) **X'05'** Failure during SPOOL API processing. (DFSIAFP0) **X'06'** Failure during Transaction Manager schedule processing. (DFSTMAS0, DFSTMCD0) **X'07'** Failure during Service Logical Unit Manager (SLUM) processing. **X'08'** Failure during Asynchronous Logical Unit Manager (ALUM) processing. **X'09'** Failure during coupling facility processing. (DFSDCFR0, DFSDMAW0) **X'0A'** Failure during queue manager processing. **X'0B'** Failure during shared queues interface processing. (DBFIPQS0, DFSITQS0, DBFILQS0, DFSILQS0) **X'0C'** Failure during NDM user exit interface processing. (DFSNDMI0) **X'0D'** Failure during shared queues CQSINFRM processing. **X'0E'** Failure during shared queues request processing. (DBFHCAS0, DBFHGU10, DBFHSQS0) **X'0F'** Failure during UOWE resync processing. (DBFHGU10, DBFHCAS0) **X'10'** Shared EMH XCF communication error. (DBFHXCS0) **X'11'** An unsolicited output message was detected. (DBFHSQS0) **X'12'** In-flight input message deleted. (DBFHCAS0)
X'67'	SNREC	**X'ED'** Sequential buffering SNAP, created during a periodical evaluation of the sequential buffering process by the SBESNAP option. (DFSSBSN0) **X'EE'** SNAP of a call to the sequential buffering buffer-handler created by the SBSNAP option. (DFSSBSN0) **X'EF'** SNAP created when the sequential buffering COMPARE option detects a mismatch between the results of a call to the buffer handler and the DASD block as stored on DASD. (DFSSBSN0) **X'FB'** An invalid AWE was detected. Some of the possible causes of the invalid AWE include conflicting parameters, missing addresses, or bad pointers. The log record indicates which of the processing modules detected the invalid AWE. **X'FD'** A SNAP call was issued. (DFSERA20) **X'FF'** A pseudoabend or dependent region abnormal termination occurred. Further information of this condition is contained in “SNAP Call facility—DFSERA20L”. (DFSERA20)

Table 6. IMS Log Records Used to Analyze IMS Problems (continued)

Type	DSECT Name	Why Written (Issuing Module)
X'67'	DFSTRHD	**X'FA'** Contains images of the incore trace tables. These tables are written to the log when requested by the OPTIONS statement in the VSPEC=parm member or the /TRACE command. (DFSTRA20)
X'69'	JM	An unauthorized 3275 terminal dialed into a line specified as VERIFY=YES. (DFSDS060)
X'6C'	CMSCREC	MSC partner systems were started. (DFSCMSW0)
X'6D'	SURVLOG	This log record is used in an XRF environment when: XRF surveillance was started or stopped. A write error occurred on the active subsystem. The interval or time-out values on the active subsystem were changed by a /CHANGE command. (DFSHIC40, DFSHSRV0, DFSISL60) **X'04'** Fast DB recovery creates this log record to indicate which TASK or ITASK received a TIMEOUT or is in a wait or loop for more than one second.
X'6E'	LUMLOG	One of the following SNA commands was processed: QEC, QC, RELQ, RSHUT, SHUTD, SHUTC, LUS. (DFSHCLG0)
X'70'	QLOGRECI	**X'00'** An online change /MODIFY command sequence completed successfully. The IMS.MODSTAT data set is being updated. (DFSICV80) **X'01'** Allows the XRF primary to signal the alternate that the transaction has been PSTOPPED by module DFSSMSC0.(DFSICV90)
X'71'	TCFLREC	Contains the name of the script member that is being processed by the Time-Controlled Option (TCO). (DFSTTIM0)
X'72'	DFSLOG	Used by dynamic terminals during sign on create, sign off delete, and sign on modification. The following subcodes identify the conditions that caused a particular log record to be written and the content of the log record: **X'01'** ETO user structure dynamically created. Contains the SPQB name and one or more CNTs. **X'02'** ETO user structure dynamically deleted. Contains only the SPQB name. **X'03'** ETO user structure modified. Contains the SPQB name and one or more CNTs. **X'04'** One or more CNTs added to an ETO user structure. Contains the SPQB name and the CNTs that were added.
X'99'	None	Created by the logging option on the EXIT= parameter on the DBDGEN. This allows a user to capture database changes that can then be propagated to another environment (for example, DB2). The subcodes indicate the type of record being logged: **X'04'** Changed data **X'28'** End of job (EOJ) **X'30'** SETS call **X'34'** ROLS call This log record is mapped by the macro, DFSDXBLK, which is not shipped. The log record layouts are explained in *IMS Version 7 Customization Guide*.

Format of Log Record Prefix Area for X'49'

The log record prefix area format for X'49' is shown in Table 7 on page 125.

Log Record Prefix Area Format

Table 7. Log Record Prefix Area Format for X’49’

Offset (Hex)	Length	Description
00	2	Record length
02	2	X'0000'
04	1	X'49' record type
05	1	X'30' record sub-type
06	2	Not used
08	8	DBD name
10	8	DD name
18	4	RBA/RBN
1C	8	Log sequence number
24	8	Subsystem ID
2C	12	Prilog time
38	4	Update sequence number (USN)

Format of X'67' Log Record

Figure 35 shows the layout of the X'67' log record. A physical log record consists of one or more subrecords. Each subrecord is followed by its associated data.

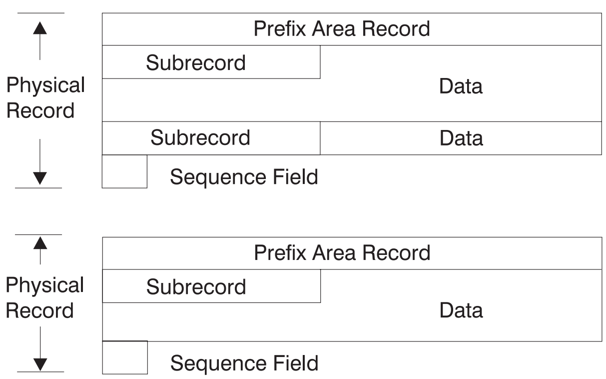


Figure 35. X’67’ Log Record Layout

Log Record Prefix Area

The format of the X'67FA', X'67FB', X'67FD', and X'67FF' records are shown below in Figure 36. All other X'67' records have individual differences.

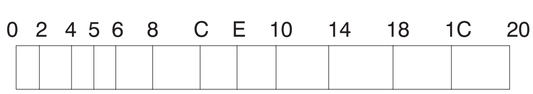


Figure 36. Log Record Prefix Area

Log Record Prefix Area Format:

Table 8. Log Record Prefix Area Format for X’67’

Offset (Hex)	Length	Description
00	2	Length of record, including sequence number
02	2	Reserved
04	1	X'67' record type
05	1	X'FB' X'FD' X'FF'
06	2	Reserved
08	4	Requestor identification
0C	2	Record segment number
0E	2	Reserved
10	4	Time
14	4	Date
18	4	Reserved
1C	4	Condition indicator

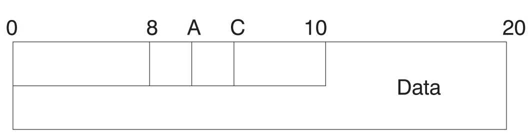
For X'67FA' records, the order of the fields from offset X'08' through X'14' is shown in Table 9.

Table 9. Log Record Prefix Area Format for X'67FA' Records

Offset (Hex)	Length	Description
08	4	Date
0C	4	Time
10	2	Table identification
12	2	Flag bytes

Log Subrecord and Data Area

Log Subrecord and Data Area Layout:



Log Subrecord Area Format:

Table 10. Log Subrecord Area Format

Offset (Hex)	Length	Description
00	8	Element identification
08	2	Reserved
0A	2	Element data length, excluding descriptor
0C	4	Main storage address of data when logged; zero when continued from previous element

Log Data Area Format:

Table 11. Log Data Area Format

Offset (Hex)	Length	Description
10	(variable)	Logged data

Log Sequence Field

Log Sequence Field Layout:

0	n	n+4	n+16
	Tstmp	Seq	

Log Sequence Field Format:

Table 12. Log Sequence Field Format

Offset (Hex)	Length	Description
n	8	STCK time stamp representing the time the log record was written. The time stamp is not necessarily on a word boundary.
n+8	8	Sequence number within the IMS control region.

File Select and Formatting Print Utility

The primary function of the File Select and Formatting Print utility (DFSERA10) is to print log records from the IMS log data set or the CICS system log.

The utility can:
- Print an entire log data set.
- Print from multiple log data sets based on control statement input.
- Select and print log records based on data contained within the record itself, such as the contents of a time, date, or identification field.
- Select and print log records based on sequential position in the data set.
- Temporarily transfer control to exit routines for special processing of selected log records.
- For CICS, print IMS records from the journal tapes.

Control statements allow you to define input and output options, selection ranges, and various field and record selection criteria.

For detailed information about this utility, see *IMS Version 7 Utilities Reference: System* and *MVS/ESA Diagnosis: Tools and Service Aids*.

Exit Routines

IMS supplies five exit routines for the File Select and Formatting Print utility: DFSERA30, DFSERA40, DFSERA50, DFSERA60, and DFSERA70. A summary of each follows.

DFSERA30 DFSERA30 formats trace records, general purpose records (type X'6701'), and SNAP records (types X'67FD', X'67FF', X'67ED', X'67EE', and X'67EF'). It also formats log records in dump format.

DFSERA40 DFSERA40 formats program isolation (PI) trace records (type X'67FA').

DFSERA50 If DL/I call image capture data is sent to the log data set, DFSERA50 formats these X'675F' log records for input to the IMS DL/I test program.

DFSERA60 If the common trace interface records are written to the log data set or the external trace data sets, DFSERA60 formats the trace entries (X'67FA').

DFSERA70 DFSERA70 selects type X'5x' log records based on search criteria. The selected records can be printed or written to tape or DASD. This exit can also convert X'5x' records to the format used in previous releases.

For a detailed description of each exit, see *IMS Version 7 Utilities Reference: System*.

Figure 37 and Figure 38 on page 129 show examples of unformatted and formatted log records. Unformatted log records include the prefix area record, the subrecord, data, and a table offset in hexadecimal. The formatted record contains the data area with its actual offset address and the table offsets.

```
DFSERA10 - PRINT PROGRAM
 000000  00540000 67FF0000 C1C2D5C4 00010000 16435280 0087049F 00000000 800000FC *........ABND.........G..........*
 000020  E2D5C1D7 C9C4407E 00000020 000D7000 E2D5C1D7 C9C4407E C1C2D5C4 40D9C5C7 *SNAPID =........SNAPID =ABND REG*
 000040  0107015D 0000010B 00000000 00B18330 0000015C                            *...}..........C....*
 000000  04140000 67FF0000 C1C2D5C4 00020000 16435280 0087049F 00000000 900000FC *........ABND.........G..........*
 000020  E2C3C440 40404040 000003D0 00B16698 E2E2C3C4 00B166A4 1BFF07FE 0AE707FE *SCD     ........QSSCD...U....X..*
 000040  00009301 02203821 02008400 E2E8E2F1 40404040 00B14FB0 00B13230 00000000 *..L.......D..SYS1    ..|........*

             physical displacement
. . .
             prefix record          record sequence for this abend

 000000  04140000 67FF0000 C1C2D5C4 00060000 16435280 0087049F 00000000 800000FC *........ABND.........G..........*

                                                             ABENDU0252

             subrecord             PST address

 000020  D7E2E340 40404040 000003E0 008DD050 00000D7B 2480501A 0000000D 008DD9F4 *PST     .......&...#...&.......R4*
 000040  00964280 0000003C C0000000 00000000 008DD0B8 00080008 008DD0B0 00100010 *.O................Y.............*
 000060  008DD0B2 00020002 008DD0B4 008DD0B8 008DD0BC 008DD0C0 00080008 008DD0C8 *...............................H*
 000080  00080008 00000000 40404040 40404040 10004040 0000000F 0000000F 00000000 *...........      .. . ..........*
 0000C0  00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 *................................*

. . .
             prefix record

 000000  04140000 76FF0000 C1C2D5C4 0000700016435280 0087049F 00000000 800000FC *........ABND.........G..........*

             subrecord                                      no address on block (PST) continuation

 000020  D7E2E340 40404040 000003E0 00000000 00000000 00AF9CC6 0000000D 00953B3F *PST    .................F.....N..*
 000040  00953A48 00953B3F 00953A40 00000000 07000000 00000000 00000258 00000000 *.N...N...N. ....................*
 000060  00000000 00953D40 00000000 0095340  00000000 00000000 00000000 00000000 *.....N.  .....N.................*
. . .
```

Figure 37. Unformatted Output Using DFSERA10

```
                                                   ABENDU0252

DFSERA30 - FORMATTED LOG PRINT
MP/BMP REG ABEND REC. AB CODE SYS = 0000 USER = 0252  RECNO = 0000015C    TIME  16.43.52  DATE 87.049
SCD
00B16698 000000   E2E2C3C4 00B166A4 1BFF07FE 0AE707FE 00009301 02203821 02008400 E2E8E2F1 *SSCD...U.....X....L.......D.SYS1*
00B166B8 000020   40404040 00B14FB0 00B13230 00000000 0000C0C0 00B16770 00B16818 00B168A4 *    ..|........................U*
...
PST
008DD050 000000   00000D7B 2480501A 0000000D 008DD9F4 00964280 0000003C C0000000 00000000 *...#..&.......R4.O.............*

                         table displacement
             original address displacement
008DD070 000020   008DD0A8 00080008 008DD0B0 00100010 008DD0B2 00020002 008DD0B4 008DD0B8 *...Y............................*
008DD090 000040   008DD0BC 008DD0C0 00080008 008DD0C8 00080008 00000000 40404040 40404040 *...............H.......         *
008DD0B0 000060   10004040 0000000F 0000000F 00000000 00000000 00000000 D7C2E5C4 E2C1D3D9 *..  .....................PBVDSAL*
008DD0D0 000080   00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 *................................*
...
```

Figure 38. Formatted Output Using DFSERA10 with Option Statement, Exit=DFSERA30

Formatting IMS Dumps Offline

This section discusses the following two methods of formatting IMS dumps offline:

- Interactive formatting, performed through a series of panels which provide formatting choices
- Formatting using JCL

You can also format IMS dumps online. For more information on online formatting, see “Formatting IMS Dumps Online” on page 156. The topics include an introduction to the dump formatter, how to use the formatted dump to analyze IMS problems, and the sections included in the formatted dump.

Introduction to the Offline Dump Formatter

The IMS Offline Dump Formatter (ODF) is a dump formatting option that reduces IMS control region abnormal termination processing. During abend processing, IMS calls the SDUMP system service of MVS to create a dump data set. Since SDUMP dumps the requested address spaces without formatting them, the processing time of an abnormal termination is shortened. After abend processing finishes, you can use the IMS Offline Dump Formatter to format (and print if you desire) either the complete dump or only those sections needed to analyze the problem.

One advantage of the IMS ODF is that you can make multiple formatting passes at the dump. This means you can first format a summary and then go back one or more times to format the control blocks you think will help you most to analyze the problem IMS encountered. See “Solving IMS Problems with the Dump Formatter” on page 130 for more information on problem solving.

Some other advantages of the Offline Dump Formatter include:

- You get an integrated IMS dump that contains the address spaces of the IMS control region, DBRC, DL/I, and IRLM address spaces. Previously, you got a separate dump for each address space.

  Also, the formatting modules are included in the dump data set. This ensures that the modules used for formatting the dump match the level of the dumped IMS control blocks. If you specify the REFRESH parameter on the user control statement for IPCS, you will get a fresh copy of the modules from the program library.
- You can use an MVS stand-alone dump, SVC dump, or SYSMDUMP to produce the dump data set for the ODF to format.
- After formatting, you can either print the dump or use interactive aids such as IPCS and ISPF browse to view the dump. See “Using IPCS and the Dump Formatter” on page 131 for more information.

Formatting dumps offline is the recommended option. If you want to format dumps online during abnormal termination, you must change the FMTO= parameter to request a SNAP dump. See *IMS Version 7 Installation Volume 2: System Definition and Tailoring* for more information.

You cannot use the ODF to format MVS trace and control block areas, the IRLM control blocks, or the VSAM modules.

Input for the Offline Dump Formatter

The dump data set you use for input to the Offline Dump Formatter must include Key 0 and Key 7 CSA, the CVT, and SQA. CSA is not required for batch or CICS-local DL/I. The dump must be machine readable.

Your most common input data sets are taken by SDUMP, because the IMS control region automatically takes an SDUMP when one of its address spaces fails.

Even if a primary SDUMP request fails, the data dumped to the point of failure can still allow successful dump formatting. Some of this information might not be included in the data sets from a secondary SDUMP request, because on the secondary request only the abending address space is dumped.

SYSMDUMPs, stand-alone Dumps (SADMP), and dumps taken by the MVS DUMP command usually produce acceptable input data sets.

For details of the SDUMP support job stream, refer to *IMS Version 7 Installation Volume 2: System Definition and Tailoring*.

Invoking the ODF

To use the Offline Dump Formatter, you must have:

- An acceptable dump in a data set
- A proper IMSDUMP entry in the IPCS Exit Control Table
- An IMS offline dump formatting control data set, which contains the FMTIMS verb followed by the options stating which subset of IMS to format
- The IMS execution library with the dump formatting modules might need to be allocated to IPCS with the ddname ISPLLIB.

You then invoke the dump formatter by executing a VERBX control statement from IPCS, or through the interactive panels. See *IMS Version 7 Utilities Reference: System* for more information on invoking the IMS Offline Dump Formatter.

Solving IMS Problems with the Dump Formatter

This section outlines how you can use the ODF to help solve IMS problems. The sections “Choosing FMTIMS Parameters” on page 131 and “Sample FMTIMS Statements” on page 133 list the FMTIMS options you could choose for particular problem areas. “Contents Formatted for FMTIMS Options” on page 136 lists the FMTIMS options alphabetically and shows the control blocks and areas formatted for each option.

Approaching the Problem

The recommended diagnostic approach with the IMS Offline Dump Formatter is:

1. Use IEBGENER or IPCS COPYDMP to transfer the dump from the SYS1.DUMPxx data set to your own data set.
2. Get an overview of the problem by formatting the dump with the subset option SUMMARY.
3. Use the abend code or reason for abnormal termination, the CALLER=id, and the TCB=id from the dump title to determine the needed subset options. “Sample FMTIMS Statements” on page 133 lists the FMTIMS statements for some specific problems.

4. Format the dump again with the subset options you determined in the previous step. Use the MIN qualifier (where possible) to reduce the output size. You can always format the data again if you need more information.

   You might also need to format the MVS trace and control block areas, the IRLM control blocks, or the VSAM modules. These blocks cannot be formatted with the IMS Offline Dump Formatter. See “Other Problems” on page 135 for more information.
5. The formatted output is spooled. You can either print the output or use ISPF to browse it. See “Using IPCS and the Dump Formatter” for more information.
6. Do additional IMS subset formatting on following jobs if necessary.
7. If you still cannot locate or fix the problem, keep the dump data set because you will need it when discussing the problem with the IBM Support Center representative.

Using IPCS and the Dump Formatter

See *OS/390 MVS IPCS User's Guide* for information on running IPCS.

Method 1: Run the IMS Offline Dump Formatter as an IPCS verb exit to format and print the dump. You can then use IPCS to view unformatted dump storage referenced in your printed dump.

Method 2: Format, but do not print the dump. Invoke split screen mode on your terminal. On one half, use ISPF browse to view the formatted control blocks. On the other half, use IPCS to view any unformatted storage referenced in the formatted control blocks.

Invoking the Offline Dump Formatter Under IPCS

There are two methods for invoking Offline Dump Formatter under IPCS; by using a VERBX command or by using menus.

Using a VERBX Command: Enter FMTIMS and the valid IMS format options after the jobname and any refresh, debug, half line, and nonheader options. The following is an example.

```
VERBX IMSDUMP, 'imsname,D,H,R,FMTIMS (SAP,ADDRESS,1234580)'
```

Choosing FMTIMS Parameters

You should know what the general problem is before attempting to choose FMTIMS parameters. If you are unsure of the problem area, format the dump with the SUMMARY option.

Table 13 shows the FMTIMS parameters recommended for general types of problems. For example, if you suspect the problem is with your logger, then give the DISPATCH, LOG, and SYSTEM parameters on the FMTIMS statement.

The control blocks and areas formatted with particular options are listed in “Contents Formatted for FMTIMS Options” on page 136.

To use Table 13, locate your problem area on the top line. Then go down that column to find the suggested formatting options (marked with an X) for that problem.

Table 13. FMTIMS Parameters for General Problems

Parameters	Problem Area							
	Checkpt/ Restart	DB	DC	FP	Log	System/ Other	Batch	CICS
CBT		X	X			X	X	X
CBTE			X					
DB		X					X	X
DBRC		X				X	X	X
DC			X				[2]	
DEDB		X		X				

Table 13. FMTIMS Parameters for General Problems (continued)

Parameters	Problem Area							
	Checkpt/ Restart	**DB**	**DC**	**FP**	**Log**	**System/ Other**	**Batch**	**CICS**
DISPATCH	X	X	X	X	X	X	[3]	
EMH		X	X	X				
LOG					X		X	
MSDB		X		X				
QM			X				[2]	
RESTART	X						[2]	
SAP			X					
SAVEAREA[1]	X	X	X	X	X	X	[2]	
SB		X				X	X	X
SCD[1]	X	X	X	X	X	X	X	X
SPST	X			X			[2]	
SUBS						X	[2]	
SUMMARY[1]	X	X	X	X	X	X	X	X
UTIL			X	X			[2]	

Notes:

1. You can use the single parameter (SYSTEM) to get the three areas (SAVEAREA, SCD, SUMMARY).
2. This parameter is ignored for batch.
3. (DISPATCH, MIN) is ignored for batch.

See "Contents Formatted for FMTIMS Options" on page 136 for a list of the modules formatted with each of the parameters. See "Syntax Restrictions on the FMTIMS Statement" on page 135 to understand the syntax rules for FMTIMS statements.

Using the Dump Title to Choose FMTIMS Parameters: When you are deciding which areas to format for your problem, you can use the CALLER= and TCB= fields of the dump title (described in "Understanding the Dump Title" on page 145) as a guide. Unless one or both of these fields specify "unknown", they should indicate why a dump was taken.

Table 14 shows the options you could choose based on valid CALLER= and TCB= information in the dump title.

Table 14. FMTIMS Parameters Based on CALLER= and TCB= Fields

CALLER=	TCB=	Recommended FMTIMS Options[1]
CTL	CTL LOG ESS LSD LSM RDS RST STC STM	DC[2], Dispatch[2], QM[2], Summary, System[2] Dispatch[2], SPST, System[2], SUBS, Summary Dispatch, Log, Restart, Summary, System Dispatch[2], MSDB, Savearea, SCD[2], Summary Dispatch[2], MSDB, Savearea, SCD[2], Summary Restart, Savearea, SCD[2], Summary Restart, Savearea, SCD[2], Summary CBT, Dispatch[2], Savearea, SCD[2], Summary CBT, Dispatch[2], Savearea, SCD[2], Summary
CURR[3]	DYA	Dispatch[2], System[2]
DBRC	DBR	DBRC[2], System[2]
DL/I	DLI STC	DB[2], Dispatch[2], SB[2], System[2] CBT, Dispatch[2], Savearea, SCD[2], Summary
DP	BMP DEP	DB[2], System[2] DB[2], System[2]

Table 14. FMTIMS Parameters Based on CALLER= and TCB= Fields (continued)

CALLER=	TCB=	Recommended FMTIMS Options[1]
FP	BMP DEP[4] XFP	DB[2,] DEDB, MSDB, System[2] DB[2,] DEDB, MSDB, System[2] DB[2,] SPST, System[2]
LOG	LOG	Log[2,] System[2]

Notes:

1. Any time you have a WAIT or LOOP problem, add SAVEAREA to your list of FMTIMS options.
2. Use the MIN qualifier for these options.
3. Normally dynamic allocation.
4. Can be either the MPP or the BMP region.

If CALLER=CURR, the current address space and IMS control region are dumped. This happens when no CALLER parameter is provided or no IMS DUMP parameter list is passed and DFSFDMP0 cannot match the caller's TCB address and ASID with the TCBs in the IMS TCB table. You can still format the dump data set, using the abend number and PSW as a guide in solving the problem. Dynamic allocation also causes CURR to be placed in the CALLER= field. In this case, format the areas listed in the above table.

If CALLER=DP, the abend occurred under the task of a dependent region address space.

If CALLER=IRLM, you need to use the IRLM Offline Dump Formatter to format the IRLM modules.

If CALLER=TRAP, a diagnostic trap for an address space abended.

Offline Dump Formatter Parameters: The Offline Dump Formatter provides the option of choosing an 80 column output format in addition to the default value of 120/132 columns. This option allows viewing of formatter output on an 80 column width screen without needing to shift left or right.

The 80 column format mode is normally selected when the IMS dump formatter is run under IPCS and the IPCS default is set to TERMINAL NOPRINT or TERMINAL PRINT. This allows dump and MVS formatting to be similar under IPCS. To select the 80 column format mode, add an "H" to the IMSDUMP formatter verb parameter string between the IMS jobname and the FMTIMS keyword. The following are examples of 80 column format option requests under IPCS.

```
VERBX IMSDUMP 'imsname,R,H,D'
VERBX IMSDUMP 'imsname,H,FMTIMS SCD'
VERBX IMSDUMP 'imsname,D,H,R,FMTIMS (AUTO,MIN)'
```

Sample FMTIMS Statements

You might be able to identify a problem area more precisely by using the CALLER= and TCB= identification from the dump title along with the abend number and explanation. (For a description of the dump title, see "Understanding the Dump Title" on page 145.) For example, you might see CALLER=CTL in the dump title and have an abend number that shows an error in the checkpoint restart processing. In this case, you can try giving the statement:

```
FMTIMS (RESTART,SAVEAREA,(SCD,MIN),SUMMARY)
```

Following is a list of possible subsets you could format for specific error situations. This list is not exhaustive and is not meant to represent every possible error situation. Use the lists in Appendix C, "Module-to-Function-to-Subfunction List," on page 465 to map the failing module (from the dump title) to a function/subfunction.

IMS Control Region Problems (CALLER=CTL): An IMS control region address space task abended. A common definition is SYS—System Services.

SYS/CHKPT System Service Checkpoint Restart Processing
FMTIMS (SUMMARY,SAVEAREA,(SCD,MIN),RESTART)

SYS/CNTRL System Service Control
FMTIMS (SUMMARY,SAVEAREA,(SCD,MIN),(DISPA,MIN))

SYS/ESS System Service External Subsystem Support
FMTIMS ((SYSTEM,MIN),SPST,(DISPA,MIN),SUBS)

SYS/INIT System Service Initialization
FMTIMS (SUMMARY,SAVEAREA,(SCD,MIN))

SYS/QMGR System Service Message Queue Management
FMTIMS (SUMMARY,SAVEAREA,(SCD,MIN),(DISPA,MIN),QM)

SYS/SCHD System Service Scheduling
FMTIMS ((SYSTEM,MIN),SPST,(DISPA,MIN))

SYS/SMGR System Service Storage Management
FMTIMS ((SYSTEM,MIN),SPST,CBT)

DBRC Problems (CALLER=DBRC): A DBRC address space task abended. You would use the same FMTIMS statement for all of the following problems with Database Recovery Control.

DBRC/CMD Database Recovery Control Command Processing
DBRC/CNTRL Database Recovery Control Processor
DBRC/EXIT Database Recovery Control Exit Processing
DBRC/SER Database Recovery Control Services
FMTIMS ((SYSTEM,MIN),(DBRC,MIN))

Data Communication Problems (CALLER=CTL): An IMS data communication task abended under the CTL TCB.

DC/CMD Data Communication Command Processing
FMTIMS ((SYSTEM,MIN),DC)

DC/CNTRL Data Communication Control
FMTIMS ((SYSTEM,MIN),(DC,MIN),(DISPA,MIN),(QM,MIN))

DC/CONV Data Communication Conversational Processing
FMTIMS ((SYSTEM,MIN),(DC,MIN))

DC/LMGR Data Communication Line Manager
FMTIMS ((SYSTEM,MIN),(DC,MIN))

DC/MFS Data Communication Message Format Services
FMTIMS ((SYSTEM,MIN),(DC,MIN))

DC/TPCALL Data Communication DL/I Telecommunications Call Processing
FMTIMS ((SYSTEM,MIN),(DC,MIN),(DB,MIN))

DL/I Problems (CALLER=DL/I or CALLER=DP): A DL/I address space task abended.

DB/ACSMTH Database Access Method Interface
FMTIMS ((SYSTEM,MIN),(DB,MIN))

DB/ANAL Database Call Analyzer
FMTIMS ((SYSTEM,MIN),(DB,MIN))

DB/CMGR Database Call Resource Management
FMTIMS ((SYST,MIN),(DB,MIN),(DISPA,MIN),(SB,MIN))

DB/DBCALL Database Call Action Processing
FMTIMS ((SYSTEM,MIN),(DB,MIN))

DB/INTRF Database Application/Scheduling Interface
FMTIMS ((SYSTEM,MIN),(DB,MIN),(DISPATCH,MIN))

Fast Path Problems (CALLER=FP): A Fast Path task abended.

FP/CNTRL Fast Path Control
FMTIMS ((SYSTEM,MIN),(DB,MIN),SPST)

FP/DEDB Fast Path Data Entry Database Processing
FMTIMS ((SYSTEM,MIN),(DB,MIN),(DEDB,MIN))

FP/EMH Fast Path Expedited Message Handling Call Analyzer
FMTIMS ((SYSTEM,MIN),(DB,MIN),(EMH,MIN))

FP/MSDB Fast Path Main Storage Database Call Analyzer
FMTIMS ((SYSTEM,MIN),(DB,MIN),(MSDB,MIN))

Log Problems (CALLER=LOG): An IMS control region address space log TCB task abended. Log is part of SYS—System Services.

SYS/LOG System Service Logging
FMTIMS ((SYSTEM,MIN),(LOG,MIN))

Other Problems: If you suspect that the failure was in VSAM, you do not need to run AMBLIST to secure a listing of VSAM modules IDA019L1 and IDA0192A of the failing system. Data Facility Products (DFP) formats the entry points for these modules. IMS includes LPA modules in offline dump data sets only if LPALIB is listed in the SDUMP options for your system. However, this is not recommended because the LPA modules occupy so much space in the dump data sets.

Refer to *MVS/ESA Diagnosis: Tools and Service Aids* if you need an MVS trace.

Syntax Restrictions on the FMTIMS Statement

The control statements in the format control data set must abide by the following syntax rules:

- The first record must contain "FMTIMS".
- A comma (,) must separate parameters from their qualifiers (MIN or cbteid).
- The number of leading blanks on both the initial record and on subsequent records is not limited.
- The last 8 bytes of all records are ignored by the formatter; you can use them for sequence numbers or any other purpose.
- A comma after the last parameter on any record indicates continuation to the next record. You can split a parameter and its qualifier, but you cannot split the spelling of a parameter over two records. For example:

  ```
  FMTIMS ((SYSTEM,MIN),(LOG,
               MIN))
  ```

  is acceptable, but the following is not:

  ```
  FMTIMS ((SYS
               TEM,MIN),(LOG,MIN))
  ```

  Notice that you can insert blanks between the last parameter in a record and the end of that record.
- The order in which the options are specified in the control statement data set has no effect on the dump formatting output order.
- Blanks imbedded within the parameters on a given record cause the formatter to assume the control statement is ended.
- The options can be upper or lowercase EBCDIC; they are translated to uppercase before being processed.
- Options can be specified by any unique number of the option's lead characters. If a nonunique abbreviation is passed, the first matching option is chosen. The FMTIMS verb cannot be abbreviated.
- Enclose an option that has a qualifier in parentheses.

Contents Formatted for FMTIMS Options

The options are listed below in alphabetical order. They can be specified on the FMTIMS statement in any order. The requested options are printed in the order stated under “Formatted Dump Output Order” on page 147. See “Table of Control Block Definitions” on page 55 for the description and mapping macro of the individual control blocks.

Some options state they “are ignored for batch”. If the dump was taken because batch processing (IMS DB or CICS) failed, the control blocks for these options are either meaningless or not included in the dump data set; therefore, the control blocks are not formatted even if you specify that option on the FMTIMS statement.

Most options can be specified with the MIN qualifier. Whenever possible, specify this qualifier to reduce the number of control blocks formatted. You can always format the dump data set again if you decide you need the additional information.

ALL

Causes a full, formatted dump.

(ALL,MIN) formats the dump as if each option were specified with the MIN qualifier.

AOI

Formats the storage for the Type 2 Automated Operator Control blocks.

AUTO

Provides an optimal subset of the IMS dump formatting options without having to first analyze the dump and without having to understand the content or use of all of the IMS dump formatting options.

This option uses the failing ITASK type information to choose one of the formatter’s functional areas, and selects the appropriate dump formatter options.

CBT

Formats storage management area control blocks, including:

- Control Block Table Header
- Individual Control Block Table entries

Output is the same if (CBT,MIN) is specified.

CBTE,cbteid

Formats all the IPAGEs for the identified CBTE type (cbteid), including:

- Individual Control Block Table entries
- All IPAGE storage of the requested CBTE type

For example, if you specify (CBTE,DPST), all DPST IPAGEs are formatted.

This option can be repeated as needed and has no defaults. The requested IPAGEs must be part of the dump data set. MIN is not valid for the CBTE option.

CLB/LLB

Permits formatting of an individual Communication Line Block or Link Line Block and its subordinate blocks. Select this option by the following:

- Address
- Node name
- LTERM name
- Communication ID or Line Number (BTAM only)

Select the LLB by address or link number.

The CLB/LLB format creates eye-catchers and index entries similar to the following:

```
**CLB/LLB                    REQUESTED CLB/LLB
```

DB

Formats areas and control blocks used for IMS Database functions. The following table shows the areas formatted under the (DB) and (DB,MIN) FMTIMS options.

Table 15. Formatted Areas under the FMTIMS options DB and DB,MIN

(DB)	(DB,MIN)
PSB Directory	same
DMB Directory	same
Intent List	not formatted
BFSP	same
DL/I Trace	same
Fast Path Trace (if Fast Path is active)	same
OSAM Pool Control Blocks and buffers	OSAM Pool Control Blocks only
Program Isolation blocks	same
All PSTs and related control blocks, including PCBs, SDBs, Savearea set, alternate DL/I DECB, DSGLRKEY, hierarchical holder, delete work area, RPLI, VSAM PLH, and retrieve trace	Active PSTs, with the same related control blocks
If Fast Path is present: EPSTs and related control blocks, including EPCBs, ESRTs, EMHBs, message buffers, XCRBs, DMHRs, and DEDB buffers	If Fast Path is present: EPSTs and related control blocks, including EPCBs, ESRTs, EMHBs, XCRBs, and DMHRs
VSAM buffer pool control blocks	same
RLPL for IRLM requests	same

In a DL/I–SAS environment, DPST formatting does not format related control blocks if the DL/I address space was not included in the dump data set.

DBRC

Formats records used by DBRC in its processing, including:

DFSRCWKB block

DFSBRLSB block

Dump Router storage

Global Data block

GDBDLTAR block

GDBDSAAR block

GDBRECAR block

GDBLISAR block

DSPEXIAG block

DSPEXOPM block

VFYWSPAC block

DSPOCPAG block

DSPJCLAR block

GDBGPDAR block

GDBRUPAR block

GDBOLCAR block

GDBMNPTR block

GDBESAVE block

GDBISAVE block

GDBCSAVE block
GDBRSAVE block
DSPCMPAG block
DSPVFILE block
DBRC Internal Trace

Output is the same if (DBRC,MIN) is specified. DBRC blocks must be present in the dump data set to be formatted.

DC

Formats the data communication areas listed in Table 16. This option is skipped if the CTL address space is not included in the dump data set.

Table 16. Data Communication Areas Formatted by DC and DC,MIN

(DC)	(DC,MIN)[1]
All CLBs, LXBs, and LCBs, with subordinate control blocks: • Current CTB or LTB, and CNT • Allocated I/O buffers • CIB, if using MFS processing • CCB, if using conversational processing • MFS work buffers • ECNT, EMHB, and message buffer, if the CTB shows a Fast Path terminal	Active CLBs, LXBs, and LCBs, with the same subordinate control blocks except that current CTB or LTB and CNT are not formatted.
SMB table	not formatted
CTT table	not formatted
SPQBs and the CNTs chained off unallocated SPQBs	not formatted

Note:

1. (DC,MIN) formats control blocks only for those lines, nodes, and links that meet at least one of the following criteria:
   a. MSC links
   b. Nodes in OPNDST or CLSDST processing
   c. Lines or nodes with allocated input, output, or receive any buffers
   d. CLBs that have an active SAP

Both DC options are ignored for batch.

DEDB

Formats the DEDB control blocks and areas. The areas included are listed in Table 17.

Table 17. DEDB Control Block Areas Formatted by DEDB and DEDB,MIN

(DEDB)	(DEDB,MIN)
ALDS	same
DMCBs, SGTs, FDTs, and MRMBs for open DEDBs	same
DMACs and ADSC for open DEDB areas	same
XCRBs, DMHRs, and buffers	XCRBs and DMHRs only
SRBs and ESRBs	same

DISPATCH

Formats areas relating to the IMS Dispatcher and its functions. Table 18 on page 139 shows the areas formatted under this FMTIMS option.

Table 18. Areas Formatted by DISPATCH and DISPATCH,MIN

(DISPATCH)	(DISPATCH,MIN)
Dispatcher work areas	not formatted
Dispatcher Trace	same
Scheduler Trace	not formatted
Latch Trace	same

(DISPATCH,MIN) is ignored for batch.

DPST,*jobname*
DPST,N,*dependent region number*
DPST,A,*address*

Permits formatting of an individual Dependent Region Partition Specification Table and its subordinate blocks for PSTs related to MPPs, BMPs, IFPs, and batch DL/I. You can specify one of the following choices:

Jobname

Dependent region number

DPST address

Output follows the DB formatting output in the dump formatter. The eye-catchers and index entries appear as follows:

```
**DPSTS                REQUESTED DPSTS
```

EMH

Formats the Expedited Message Handler areas used by IMS Fast Path, as shown in Table 19. The CTL address space must be included in the dump data set for this option to be formatted.

Table 19. Areas Formatted by EMH and EMH,MIN

(EMH)	(EMH,MIN)
RCTEs	same
BALGs, EMHBs, and message buffers	BALGs and EMHBs only

The CTL address space must be included in the dump data set for this option to be formatted.

LOG

Formats control blocks and areas used by the IMS logger. The areas included are listed in Table 20. These areas, except for the WADS and the DLOG trace, are repeated in the dump when the IMS Monitor is active.

Table 20. Areas Formatted by LOG and LOG,MIN

(LOG)	(LOG,MIN)
LCD	same
Restart Log Work Area	same
WADS and the data necessary to manage it	WADS only
OLDS prefix and the buffer associated with it	OLDS prefix only
Log DSET, which defines all OLDS currently available for use	same
Message work areas and Logger message areas	same
DLOG trace	same

MSDB

Formats the Main Storage Databases used by IMS Fast Path. The areas included are listed in

Table 21.

Table 21. Main Storage Databases Formatted by MSDB and MSDB,MIN

(MSDB)	(MSDB,MIN)
MSDB headers	same
all MSDBs	not formatted

POOL, NAME, poolid

Invokes formatting of the storage manager control blocks and the pool storage for any of the following pools:

ALL	FPWP
CESS	HIOP
CIOP	MFBP
DBWP	PSBW
DLDP	QBFL
DLMP	QBUF
DPSB	SPAP
EMHB	LUMC
EPCB	LUMP

NAME is an optional keyword indicating the pool name parameter. If NAME is omitted, the first parameter is assumed to be the pool name.

The poolid is a required 4-character pool name of an existing storage manager pool or the keyword ALL. If ALL is specified, the following storage pools are formatted:

HIOP	DLMP
CIOP	DPSB
CESS	DLDP
SPAP	DBWP
EMHB	MFBP
FPWP	EPCB
QBUF	LUMP
QBFL	LUMC

ALL triggers the formatting of any storage manager trace table entries along with the storage manager control blocks and pool storage.

MIN is an optional keyword. If MIN is specified for one of the dynamic pools (HIOP, CIOP, EMHB, FPWP, CESS, SPAP, LUMC, LUMP) only the storage manager pool header and block headers are formatted. If MIN is omitted, the pool header control block is formatted along with the blocks and block headers representing the dynamic storage pool.

QM

Formats the IMS queue manager’s control blocks and areas. The formatter skips this option if the CTL address space is not included in the dump data set. The areas included are listed in Table 22.

Table 22. Areas Formatted by QM and QM,MIN

(QM)	(QM,MIN)
Qpool Prefix	same
Qpool Buffer Prefix	same
Qpool Buffer	not formatted

Both QM options are ignored for batch.

RESTART

Formats the IMS restart control blocks and related areas, including:

- Checkpoint ID table
- SIDXs and their subordinate blocks:
  - All LCREs for the SIDX entry being processed
  - All RREs for the SIDX entry being processed
- All RPSTs for the SIDX entry being processed
- FRB, if present

Output is the same if (RESTART,MIN) is specified. Both RESTART options are ignored for batch.

SAP, ECBADR, ecbaddr
SAP, ADDRESS, sapaddr

The SAP option can be invoked using either the SAP address or the SAP's ECB address (providing that the ECB is a valid ITASK and has a prefix pointing to a SAP). The SAP option request can be placed either on the IMSDUMP verb line after FMTIMS or in the DFSFRMAT data set. The following are examples of SAP option requests:

```
VERBX IMSDUMP'imsjname,II,N,FMTIMS (SAP,ADDRESS,20864C0)'
VERBX IMSDUMP'imsjname,FMTIMS SCD,(SAP,ECBADR,3064250)'
```

For compatibility reasons, the MIN qualifier is allowed, but the output is the same. Individual SAP option formatting is also available on the IMS Low Level Panel of the IMS Interactive Dump Formatter dialog. The ADDRESS parameter can be omitted since ADDRESS is the default TYPE for the SAP option.

Individual SAP/save area formatting allows complete formatting of SAP/save areas when additional information is required. The output from individual SAP formatting is the same as the SAVEAREA option output. Individual SAP formatting provides the following eye-catcher/index entry:

```
**SAPS      REQUESTED SAPS
```

SAVEAREA

Formats the save area information, including:

- Formatted SAPs and any UEHBs anchored off the SAPs.

  Restriction: The UEHBs cannot be formatted if the CTL address space is not included in the dump data set.
- Formatted Save Area Sets associated with each SAP.
- Unformatted dump of the IPAGEs containing the SAPs.

If the DL/I address space is not in the data set, then the DL/I SAPs are not formatted. If the CTL address space is not in the data set, then the non-DL/I SAPs are not formatted. Output is the same if (SAVEAREA,MIN) is specified. Both SAVEAREA options are ignored for batch.

The SAVEAREA also comes with a summary option that allows a faster overview scan of the IMS ITASK status within a dump. The SAVEAREA SUMmary output reduces the SAP/Savearea formatting to minimal data while adding keyword scan capability and automatic computation of the exit offsets. This reduces keystroke resources required to overview the ITASK status and ITASK module flow. The SAVEAREA SUMmary and individual SAP formatting provides the following eye-catcher/index entry:

```
**SSS       SAP/SAVE CONDENSED SUMMARY
```

SAVEAREA SUMmary formatting contains the following scannable keywords with their associated meanings:

RUN
: ITASKs that are active are given a RUN indicator. Abend and loop analysis is usually concerned only with running ITASKs.

LATCHREQ
ITASKs that are waiting for an IMS SLX latch (not checkpoint restart LATE latches) are given a LATCHREQ indicator. Enabled wait problem analysis often requires analyzing ITASKs that are waiting for latches.

LATCHOWN
ITASKs that own an IMS SLX latch (not checkpoint restart LATE latches) are given a LATCHOWN indicator. Enabled wait problem analysis often requires analyzing ITASKs that own SLX latches.

ITASK type
The ITASK type is in the summary and is scannable. The ITASK type names are not at the end of the scan list, however. The ITASK type is preceded by the label “type”. The possible type names can be gotten from the DFSCIR macro prolog.

SB

Formats the control blocks, areas, and buffers of the Sequential Buffering function (SB) of IMS. This option also formats those DL/I control blocks which are important for debugging the SB function.

The SB information is divided into four sections. Table 23 shows which sections are formatted with the SB and SB,MIN options. A description of the sections follows Table 23.

Table 23. Sections Formatted by SB and SB,MIN

(SB)	(SB,MIN)
Subsystem overview	same
PST overview[1]	same[2]
Sorted blocks[1]	same[2]
Sorted buffers[1]	not formatted

Note:

1. The DL/I address space must be included in the dump data set for these areas to be formatted.
2. Formatted only if you requested a conditional SB activation for that application or PST.

The SB information is divided into the following sections:

1. Subsystem Overview of SB—provides an overview of SB control blocks from an IMS subsystem point-of-view. The SDCBs appear in the order in which they are anchored in the SBSCD. Each SDCB is followed by its SDSGs. The section contains the following information:

   SB section of the SCD

   SBSCD, including the SBHE blocks

   SDCBs

   SDSGs
2. PST Overview of SB—formats the SB control blocks (and other IMS control blocks significant to SB) for each active PST. These blocks are sorted in hierarchical order. For example, the first DBPCB and its JCB, DSGs, EDSGs, and SDSGs; then the second DBPCB with its subordinate blocks, and so on. The section contains the following information:

   SB and buffer-handler sections of the PST

   PST DECB prefix

   SB extensions to the PST

   SB work area

   SBPARMS

   DBPCBs and their JCBs, DSGs, ESDGs, and SDSGs
3. Sorted SB Blocks—contains SB control blocks (and other IMS control blocks significant to SB) sorted according to their virtual storage address. The section contains the following information:

   DBPCBs

   DCB with its OSAM extensions

   DSGs

ESDGs
JCBs
OV-IO DECB prefix
PST DECB prefix
SB extensions to DCBs
SB extensions to DSGs
SB extensions to the PST
SB work area
SBPARMS
SBUFs
SCARs
SRANs

4. Sorted SB Buffers—contains the SB buffers of each SB buffer pool. The SB buffers of one SB buffer pool are contiguous in storage and are formatted as one entity. The buffer pools are then sorted by virtual storage address.

SCD
Formats the IMS SCD and related areas. The areas included are listed in Table 24.

Table 24. Areas Formatted by SCD and SCD,MIN

(SCD)	(SCD,MIN)
SCD	same
Latch Extensions	same
Scheduler Sequence Queues	not formatted
Fast Path SCD Extension, if Fast Path is active	same
Formatted dump of the batch key 7 SCD	same
LU 6.2 SCD extension	same

SPST
Formats the system PSTs, which are ITASKs used by IMS. This includes:
- Global system PSTs
- Local control region address space PSTs
- Local DL/I address space PSTs
- Areas related to the above PSTs, including LWA and IRLMA

Some SPSTs are not formatted if the CTL address space is not in the dump data set. Output is the same if (SPST,MIN) is specified. Both SPST options are ignored for batch.

SUBS
Formats the areas and control blocks that IMS uses to manage subsystems, including:
- Subsystem trace
- Global ESET block

Output is the same if (SUBS,MIN) is specified. Both SUBS options are ignored for batch.

SUMMARY
Formats the current diagnostic section.

The SUMMARY data areas are not formatted if the SDWA address space is not part of the dump data set. (For abends and batch processing, the SDWA address is saved by the ESTAE module. For online processing, the dump must be taken by DFSOFMD0, and the SDWA parameter must be passed at DFSDUMP time.)

The areas formatted with this option include:

- Failing PSW
- Abend code
- Module name
- Registers at time of abend
- 256 byte instruction area—128 bytes above and below the failing PSW
- 16 register storage areas—512 bytes above and 256 bytes below the registers at time of abend
- IMS's SDWA
- Failing SAP and its UEHB
- Failing ITASK when the ITASK is a DPST, system PST, CLB, or LLB (dependent region errors, some systems services errors, terminal process errors, and MSC errors)

  The SUMMARY option names the ITASK type when it is determined, even if it is not one of the ITASK types that provide for additional formatting. The ITASK type name is two to four characters. If it is unknown, the type name is "UNKN".

Output is the same if (SUMMARY,MIN) is specified.

SYSPST

Permits formatting of an individual system partition specification table and some of its subordinate blocks. Select this option by address or system PST name. This option creates eye-catchers and index entries similar to the following:

```
**SYSPSTS                 REQUESTED SYSTEM PSTS
```

SYSTEM

Formats the SUMMARY, SAVEAREA, and SCD areas as one group. The areas and control blocks formatted are the same as if each of the options were invoked separately.

(SYSTEM,MIN) is formatted as though each of the options were specified with MIN.

See the individual options for a list of the areas formatted.

TRACE, NAME, table-id

Gets a new search module that invokes the normal trace format control module (DFSATRA0) to format trace tables separately. This option enables viewing of trace table data without having to format the entire option that usually includes the formatted trace table. The TRACE option request uses the two-character trace table EBCDIC ID code from the Trace Selection panel. The dump formatter ISPF panels also accept an option of "ALL" to format all IMS trace table traces. The Interactive Dump Formatter dialog TRACE SELECTION panel provides a selectable list of IMS trace tables with the trace name, internal ID, and description. The following are sample TRACE format requests, followed by comments for each. In each case, the NAME keyword can be omitted since NAME is the default TYPE parameter. The following is a request for the DL/I trace table.

```
FMTIMS...(TRACE,NAME,DL),...
```

The following is a request for the dispatcher trace table and the DL/I trace table with a MIN option that is ignored.

```
FMTIMS...,(TRACE,NAME,DL,MIN),(TRACE,NAME,DS)...
```

UTIL

Formats the control blocks for the IMS Partial Database Reorganization utility, including:

Common area

Database table

Segment table

Action table

Output is the same if (UTIL,MIN) is specified. Both UTIL options are ignored for batch.

Using the Formatted Dump

This section describes the formatted dump's title, how to locate specific control blocks and areas in the formatted dump, and the order in which formatted control blocks are presented. A sample formatted dump is at the end of the section.

Understanding the Dump Title

The contents of the dump titles created by the dump assist module (DFSFDMP0) and the initialization routines vary, depending on the internal DFSDUMP parameters provided and the SDUMP errors met.

Following are three possible dump title formats.

Title Format 1: DFSFDMP0 issued the SDUMP and passed the SDWA parameter. The CALLER parameter was either passed to DFSFDMP0 or the routine generated the parameter using the IMS TCB table.

```
ljjjjjjjj ABEND SYS sss USER uuuu-rrr, DATE.TIME: ddd.tttttt,
                CALLER=cccc, TCB=xxx, MODULE=mmmmmmmm,i
```

where:

```
        l: length of title in hexadecimal - here 91 decimal
 jjjjjjjj: jobname
      sss: system abend code
     uuuu: user abend code
      rrr: optional user abend reason code
      ddd: Julian day of year
   tttttt: time, in the form HHMMSS
     cccc: DFSDUMP caller parameter or blanks
      xxx: abending TCB or 'UNK'
 mmmmmmmm: abending module or 'UNKNOWN', using the SDWA
        i: indicator if primary (P) or secondary (S) request
```

Title Format 2: DFSFDMP0 issued the SDUMP, but did not have an SDWA. The CALLER parameter was either passed to DFSFDMP0 or the routine generated the parameter using the IMS TCB table.

```
ljjjjjjjj  DATE.TIME: ddd.tttttt, IMS DUMP REQUESTED,
                CALLER=cccc, TCB=xxx, REASON=rrr,i
```

where:

```
        l: length of title in hexadecimal - here 80 decimal
 jjjjjjjj: jobname
      ddd: Julian day of year
   tttttt: time, in the form HHMMSS
     cccc: DFSDUMP caller parameter or blanks
      xxx: abending TCB or 'UNK'
      rrr: optional user reason code
        i: indicator if primary (P) or secondary (S) request
```

Title Format 3: This format is generated for a DBCTL Database Resource Adapter (DRA) SDUMP.

```
ljjjjjjjj DRAthd tnnnn mmmm...mmRTKN=rrrrrrrrxxxxxxxxxxxxxxxx
```

where:

```
        l: length of title in hexadecimal - here X'5D'
 jjjjjjjj: DBCTL jobname
   DRAthd: abend component of DRA:
           DRA    - DRA control processing abended
           DRATHD - DRA thread abended
        t: abend type
           S = system abend
           U = user abend
     nnnn: abend code
           for system abend, nnnn=hex
           for user abend, nnnn=decimal
```

```
mmm...m: message text (up to 40 characters) that describes the
         error - See the possible texts following this example.
  RTKN=: 16-byte recovery token
         (present only for DRA thread abends)
rrr...r: first 8 bytes of the recovery token in characters -
         identifies the ID of the CCTL region
xxx...x: second 8 bytes of the recovery token in hexadecimal.
```

The possible error messages for `mmm...m` follow. The issuing module precedes the message text.

DFSPRRA0, DBCTL FAILURE DURING DRA TERM

DFSPRA10, DBCTL FAILURE DURING IDENTIFY

DFSPRA20, DBCTL FAILURE DURING RESYNC

DFSPRA50, DBCTL FAILURE DURING PURGE

DFSPINI0, FAILURE ESTABLISHING ESTAE

DFSPAT00, GETMAIN FAILURE

DFSPINI0, SSI FAILURE DURING SONCRT

DFSPINI0, DBCTL FAILURE DURING SONCRT

DFSPSCH0, SSI FAILURE DURING SCHED

DFSPSCH0, DBCTL FAILURE DURING SCHED

DFSPUSC0, SSI FAILURE DURING UNSCHED

DFSPUSC0, DBCTL FAILURE DURING UNSCHED

DFSPSYN0, DBCTL FAILURE DURING SYNC

DFSPDLI0, DBCTL FAILURE DURING DLI

DFSPPTK0, DBCTL FAILURE DURING PRIME

DFSPTTH0, SSI FAILURE DURING TERMTHD

DFSPTTH0, DBCTL FAILURE DURING TERMTHD

DFSPRA40, PQE CANNOT BE PROCESSED

DFSPRRA0, PQE OR PAPL IS INVALID

DFSFPRA0, CONTROL TCB ESTAE INVOKED

DFSFPAT0, THREAD TCB ESTAE INVOKED

DFSFPRA0, DRA ESTAE FAILED TO ESTABLISH ESTAE

NO OTHER DRA MESSAGE

Locating Control Blocks in the Dump

The Offline Dump Formatter output includes eye-catchers and an index to help you locate individual control blocks.

Eye-catchers: To assist you in rapidly locating areas that are dumped, eye-catchers are printed near the major control blocks in the formatted dump. Eye-catchers are also useful when you are using IPCS to view the formatted dump. Examples of eye-catchers are:

****SCD** System Contents Directory Area

****SSA** SAP and Save Area

****SB-1** Subsystem Overview for Sequential Buffering

Eye-catchers are also listed at the front of the formatted dump.

Index: The formatted dump also contains an index created by the MVS Index Service Routine. Index entries are created at the following points:
- Each time an eye-catcher is processed during formatting
- After the Offline Dump Formatter is finished with its processing

Entry length is limited to 40 decimal characters.

The index is located at the end of the formatted dump.

Formatted Dump Output Order

The following list shows the order in which the Offline Dump Formatter prints control blocks. If you specify **FMTIMS ALL** and all necessary data is available to the formatter, you get all of the areas listed. The order does not change when you specify subset options, but only the areas you specify are formatted. Descriptive information has been added for some control blocks where it would be useful.

ODF Initialization Messages
These messages appear when the formatter is unable to find particular address spaces in the dump data set. For an explanation of individual messages, see *IMS Version 7 Messages and Codes, Volume 1*.

Copy of FMTIMS Control Statement

Eye-catchers
Eye-catchers of the areas you requested formatted on this pass of the formatter.

An eye catcher could be included in this list even if the dump formatter was unable to format the control block, because the list is built from the parameters you include in the FMTIMS statement.

Diagnostic Area
Contains the PSW, system and user completion codes, save area ID of the module that was executing, and registers in use when abnormal termination occurred.

Instruction Area
Contains the area of storage from 128 bytes before to 128 bytes after the address of the failing instruction in the PSW.

Register Area
This area contains 512 bytes above and 256 bytes below each register value in the passed SDWA. The ASID used is the one passed in the SDWA.

System Diagnostic Work Area
The mapping DSECT is IHASDWA.

Referenced SAP
The mapping DSECT is ISAP.

System Contents Directory
The mapping DSECT is ISCD.

SCD Latch Extension
The mapping DSECT is ISCD.

Scheduler Sequence Queues
Controls the status of each region. The mapping DSECT is ISCD.

FP ESCD
The mapping DSECT is DBFESCD.

Control Block Table
Contains entries of control blocks that macro DFSCBTS uses for tracking. The mapping DSECT is DFSCBTS.

Control Block Table Pools
All IPAGEs for CBTE types requested with the (CBTE,cbteid) option.

Save Area Trace

SAPs with their Active UEHBs

Save Area Prefix
All SAPs are SNAPed. Each SAP is followed by its save area set. At the end of this section, all of the SAP IPAGEs are dumped.

IMS Task Dispatch Work Area
The mapping DSECT is IDSPWRK.

DBRC Task Dispatch Work Area
If present in the system, it is mapped.

IMS Control Task Dispatch Work Area
Contains the same information as the IMS log task dispatch work area.

Dependent Region Dispatch Work Area
For every dependent region in IMS, the dispatcher work area is mapped.

Dispatcher Trace Data
DSECT IDSPWRK contains the function codes associated with the dispatcher and an explanation of each code.

Scheduler Trace Data
Scheduler trace data is mapped by DFSSCHED. The trace entries contain scheduler function codes.

Latch Trace Data
The trace entries contain latch and unlatch function codes. The mapping DSECT is IDLIVSAM TRACENT.

Timer Work Areas
These are control blocks used by the internal IMS timers.

System PSTs
These are system work areas for any online or batch region. The mapping DSECT is IPST.

Restart Work Areas
See RESTART on page 141 for a list of these areas.

Log Control Directory
Contains information about the IMS log. The mapping DSECT is LCDSECT.

Log Work Areas

Log Buffers
Each log buffer contains buffer information and the log control DECB. The mapping DSECT is LCDSECT.

Open Record
Contains the type 06 log record. The mapping DSECT is ILOGREC.

Control Record
Contains the type 42 log record. The mapping DSECT is ILOGREC.

Monitor Log Directory
Contains the same information as the log control directory.

DLOG Trace Data
Trace table used to show IMS logging activity. The mapping DSECT is ILOGREC (67FA).

Subsystem Control Table

Attach Work Areas

PSB Directory
A SNAP of the PSB directory. The mapping DSECT is PDIR.

DMB Directory
A SNAP of the DMB directory. The mapping DSECT is DDIR.

Intent List
The DL/I address space must be in the dump data set for this list to be formatted.

Fast Path Trace

Dependent Region PST formatting
For each DPST:
- PST
- Savearea
- PDIR
- Intent List
- PSB prefix
- PSB Index Maintenance, Index I/O, I/O, SSA, and User Parms work areas
- SMB
- DB PCB blocks
- Delete work area
- Retrieve Trace
- HD Space Trace
- FLDS
- RPL
- IRLM area
- PST log work area
- Fast Path EPST and chain addresses, ECNTs, EMH message, EPCBs, XCRBs, and DMHR

BFSP
Formats the buffer pool prefix. The mapping DSECT is BFSP.

BFUS
Formats the subpool prefix. The mapping DSECT is BFUS. The mapping DSECT is RPLI.

DL/I Data
A dump of the DL/I lock activity and program isolation trace table. The mapping DSECT is IDLIVSAM TRACENT.

Lock Activity Trace Data
See DL/I Data.

Program Isolation Data
Includes the QEL, QCB and REQ areas. The mapping DSECT is XC00.

OSAM Control Blocks
The system attempts to follow the main pool, the subpool header, and the buffer prefix, and to dump the buffer. However, if an error is encountered during formatting, the entire buffer pool is SNAPed from the last valid subpool address.

DL/I Trace Table

Sequential Buffering Blocks
Sequential Buffering information is grouped into the following four sections. (See the explanation of the (SB) FMTIMS option on page 142 for a complete list of the blocks dumped in each section.)

1. Subsystem Overview for Sequential Buffering
2. PST Overview of Sequential Buffering control blocks
3. Formatted Sequential Buffering control blocks
4. Sequential Buffering buffers

DEDB Formatting

Fast Path EMH Formatting [1]

Fast Path MDSB Formatting [1]

Communication Line Blocks and Subordinate Blocks [1]
For each CLB line, all the control blocks associated with that line are formatted.

CTB [1]
The mapping DSECT is ICLI CTBBASE=0.

Input Buffer [1]
A SNAP of the input buffer, if input is active.

Output Buffer [1]
A SNAP of the output buffer, if output is active.

CCB [1]
Present if a conversation is active or held. The mapping DSECT is ICLI CCBBASE=0.

CIB [1]
Present if MFS is in use. The mapping DSECT is ICLI CIBBASE=0.

Communication Terminal Table [1]
Defines terminal characteristics. The mapping DSECT is ICLI CTTBASE=0.

SPQB Entries [1]
Entries on the subpool queue block chain. Unallocated CNTs are also formatted here.

SMB Table [1]
This table defines transaction characteristics in the IMS system. The mapping DSECT is IAPS SMBBASE=0.

Queue Manager Pool Prefix and Buffers [2]
The mapping DSECTs are ICLI POOLBASE=0, ICLI BFRBASE=0, and QPOOL. The buffer prefix list contains the address of each buffer's prefix, status byte, and first and last pending and current DRRN.

Batch Utility Areas

1. These areas are not dumped in a DBCTL environment.

DBRC Work Areas

LUM Trace
Allows LU 6.2 activities to be analyzed with the MVS/ESA™ APPC trace entries by the LU manager.

Interactive Dump Formatter

The interactive dump formatter provides ISPF dialog support for offline dump formatter requests. This simplifies the process of making requests by providing menus for format option selection, help members for online option explanation, automatic terminal and spool output control, and a configuration panel to provide interactive assistance in defining the IMS environment.

The IMS Interactive Dump Formatter menu is available from the component analysis section of the IPCS dialogs (IPCS ISPF selection 2.6). The primary menu includes the following entries:

- A configuration and initialization entry for IMS formatting control and initialization
- An IPCS BROWSE entry for speed of use
- A high-level formatting entry for traditional IMS formatting requests of large functional areas
- A low-level entry for ITASK-level and single-element formatting
- An analysis entry for IMS-provided summary or analysis formatting
- A user panel for user-controlled use
- An EDA entry for invoking the IMS enhanced dump analysis menu
- An entry for IMS dump formatting tutorial assistance
- An entry for exiting dump formatting
- An entry for formatting other IMS component address spaces, such as CQS and BPE.
- An entry for formatting other IMS-related products, such as IMS Connect, database recovery service, and their associated BPEs.

Using Interactive Dump Formatter Menus

To use the menus, do the following:

1. Go to the IPCS Component Analysis panel.
2. Select DFSAAMPR. The panel in Figure 39 appears.

```
DFSAAMPR -------------  IMS DUMP FORMATTING PRIMARY MENU   --------------------
OPTION  ===>

   0  INIT        - IMS formatting initialization and content summary
   1  BROWSE      - Browse Dump data set (IPCS norm)       *******************
   2  HI-LEVEL    - IMS Component level formatting        *USERID  - SKONO
   3  LOW-LEVEL   - IMS ITASK level formatting            *DATE    - 00/01/06
   4  ANALYSIS    - IMS dump analysis                     *JULIAN  - 00.006
   5  USER        - IMS user formatting routines          *TIME    - 15:00
   6  OTHER COMP  - Other IMS components (BPE, CQS...)    *PREFIX  - SKONO
       7  OTHER PROD  - Other IMS-related products            *TERMINAL- 3278
   E  EDA         - IMS Enhanced Dump Analysis            *PF KEYS - 24
   T  TUTORIAL    - IMS dump formatting tutorial
   X  EXIT        - Exit IMS dump formatting


Enter  END  command to terminate IMS component formatting
```

Figure 39. IMS Dump Formatting Primary Menu PanelIMS Dump Formatting Primary Menu Panel

2. These areas are not dumped in a DBCTL environment.

3. If this is the first time you are reading the dump, select 0 (Initialization). The panel in Figure 40 appears:

```
DFSAAEI0 ----------------  IMS DUMP CONTENT STATUS    ------------------------
COMMAND ===>

  Enter the IMS CTL/BATCH or DL/I jobname to cause the IMS symbols to
  be set for this dump. Request subsystem list for possible IMS names.

 IMS SUBSYSTEM LIST DESIRED? (Y or N)===>  N

          JOBNAME        ID         ASID        DUMPED?
-------------------------------------------------------------------------------
  CTL
  DL/I
  DBRC
  IRLM



  ABEND CODE =  SYS               USER
  MODULE     =

  IMS SDWA ADDRESS -               IMS RELEASE -
  IMS SCD  ADDRESS -
  ABENDED ASID     -
```

Figure 40. IMS Dump Formatting Initialization/Content Panel - InactiveIMS Dump Formatting Initialization/Content Panel - Inactive

4. Enter the IMS jobname in the row marked CTL, or the DLI jobname in the row marked DL/I, and press enter. Either jobname is sufficient. If unknown, enter a `Y` next to the `IMS SUBSYSTEM LIST DESIRED` prompt to scan for dumped IMS address spaces. When valid information has been supplied, the panel has several fields filled in, as shown in Figure 41. Press PF3 to return to the primary menu.

```
DFSAAEI0 ----------------  IMS DUMP CONTENT STATUS    ------------------------
COMMAND ===>

  Enter the IMS CTL/BATCH or DL/I jobname to cause the IMS symbols to
  be set for this dump. Request subsystem list for possible IMS names.

 IMS SUBSYSTEM LIST DESIRED? (Y or N)===>  N

          JOBNAME        ID         ASID        DUMPED?
-------------------------------------------------------------------------------
  CTL     DTSIMSGA      SYS3       0019          YES
  DL/I    NA                       0019          N/A
  DBRC    DTSDBRCA                 001A          YES
  IRLM    N/A           N/A        N/A           N/A



  ABEND CODE =  SYS   0C4         USER    0
  MODULE     =   DFSSCBT0

  IMS SDWA ADDRESS -   007BC680    IMS RELEASE -   320
  IMS SCD  ADDRESS -   00BA1E30
  ABENDED ASID     -   0019
```

Figure 41. IMS Dump Formatting Initialization/Content Panel - ActiveIMS Dump Formatting Initialization/Content Panel - Active

5. IMS dump formatting is invoked from the high-level, low-level, and analysis option menus. Each menu contains a list of selectable entries. Place an `S` or `M` next to an entry to request formatting, and press enter to process your selections. Examples of the high-level and low-level options menus are shown in Figure 42 on page 153 and Figure 43 on page 154.

```
-------------------- IMS HIGH LEVEL DUMP FORMATTING OPTIONS -----  ROW 1 OF 23
Command ===>                                                  Scroll ===> PAGE

N <====SPOOL OUTPUT? (Y or N)       N <====REFRESH FORMATTER? (Y or N)
       S = select   M = select,min     select choices and hit enter
                                       to process or UP/DOWN to scroll
Additional IMS format requests===>

Cmd  Option          Description
-------------------------------------------------------------------------------
_    AUTO       Internally determined options (by failing ITASK type)
_    ALL        All high level IMS dump formatting options
_    SUMMARY    PSW, regs, SAP, failing ITASK blocks at time of abend
_    SCD        SCD, SLX, FP ESCD, scheduler sequence queues
_    SAVEAREA   SAP, savearea, ECB prefix, UEHB  (sorted by DSPNO)
_    DISPATCH   Dispatcher work areas, Dispatcher and Latch traces
_    SPST       System PSTs and subordinate blocks
_    RESTART    CHKPT ID table, SIDX, LCRE, RPST, RRE, EQEL, IEEQE, FRB
_    LOG        LCD, log buffer prefixes, log buffers (OLDS and MON)
_    DB         DDIRs, PDIRs, intent list, DLI and LOCK traces, DPSTs
_    DEDB       ALDS, DMCB, DMAC, XCRB, SRB, ESRB
_    MSDB       BHDR, Main storage databases
_    DC         CLB, LLB, VTCB, CTB, CNT, CTT, SMB, SPQB, LGND, USRD
_    EMH        RCTE, BALG, EMHB
_    QM         QPOOL, QSCD, QMGR hash table, QBFPRF, Queue buffers
_    UTIL       Partial reorg blocks
_    SUBS       External subsystem blocks and trace
_    CBT        Control block table
_    SDE        Storage Descriptor Element Blocks and Storage
_    SB         Sequential buffering control block formatting
_    DBRC       DBRC control blocks and trace
_    IRLM       IRLM control block formatting
_    LUM        LUM trace and control blocks
```

Figure 42. IMS High-Level Dump Formatting PanelIMS High-Level Dump Formatting Panel

The IMS high-level formatter request panel allows selection of IMS formatting areas in a quick and easy manner. The MIN qualifier and spooling and terminal outputs can be selected as well.

```
DFSAALL0 ----------  IMS LOW LEVEL DUMP FORMATTING OPTIONS ------  ROW 1 OF 17
COMMAND ===>                                                  Scroll ===> PAGE

N <===== SPOOL OUTPUT? (Y or N)   N <==== REFRESH FORMATTER? (Y or N)
         S or  M at left plus required ARGument value to select option.
         (Items marked *P* will prompt if ARG blank). UP/DOWN to scroll

Additional IMS formatter requests===>

Cmd  Option   Type      ARG           Argument description
v----------------------vvvvvvvv-----------------------------------------------
_   CLB      ADDRESS                CLB/LLB address (hexadecimal)
_   CLB      NODE                   VTAM node name
_   CLB      LTERM                  IMS logical terminal name (CNT)
_   CLB      CID                    VTAM communication ID (hexadecimal)
_   CLB      LINE                   BTAM line number (decimal)
_   LLB      LINK                   MSC link number (decimal)
_   DPST     ADDRESS                Dependent region PST address (hexadecimal)
_   DPST     NUMBER                 Dependent region PST number (hexadecimal)
_   DPST     NAME                   Dependent region PST jobname
_   SYSPST   ADDRESS                System PST address
_   SYSPST   NAME               *P* System PST name
_   TRACE    NAME               *P* Trace table ID  (2 characters)
_   SAP      ADDRESS                Savearea block address (hexadecimal)
_   SAP      ECBADR                 SAP's ECB address (hexadecimal)
_   POOL     NAME               *P* IMS storage pool name
_   CBTE     NAME                   Control Block Table name
_   LUB      NAME                   LU name
```

Figure 43. IMS Low-Level Dump Formatting Selection PanelIMS Low-Level Dump Formatting Selection Panel

```
DFSAALA0 ---------------------  IMS DUMP ANALYSIS -----------------------------
COMMAND ===>

N <=====SPOOL OUTPUT? (Y or N)    N <====REFRESH FORMATTER? (Y or N)

    Put an S left of desired option to select. Additional FMTIMS
    strings may be entered after "ADDITIONAL REQUESTS". Press Enter to
    process.

 Additional formatting requests ====>

     analysis     output
CMD  option       description
v-----------------------------------------------------------------------------
_     SAPS        savearea set overview analysis
```

Figure 44. IMS Analysis Selection PanelIMS Analysis Selection Panel

Using the ″Other IMS Components″ Formatting Panels

Some IMS components (for example, the Common Queue Server (CQS)) run under the Base Primitive Environment (BPE) system services, rather than the IMS system services. These components use the BPE formatter, and their format options are selected separately from the main IMS dump formatter.

Select ″Other IMS Components″ formatting from the IMS dump formatting primary menu panel, option 6. This choice will allow you to further select the specific component formatting to be done (for example, BPE or CQS). Dump initialization for these components is done via the BPE initialization and status panel under option 6, and not by option 0 on the primary menu.

Using the ″Other IMS-Related Products″ Formatting Panels

IMS provides a selection for calling the dump formatters for products that are separate from IMS, but are still related to IMS.

Select ″Other IMS-Related Products″ formatting from the IMS dump formatting primary menu panel, option 7. You are then presented with a list of all possible products. However, you can only use the formatters of those products that are installed on your system. Each product's formatter will provide a dump initialization panel; you should not use the panel from option 0 on the primary menu.

IMS IPCS Symbols

IMS offline dump formatting creates IPCS symbols for selected key IMS control blocks. The Interactive Dump Formatter helps create these symbols and then uses them to make Offline Dump Formatter requests easier by providing known starting points, including starting points for CLISTs.

IMS creates and lists the IPCS symbols before the SUMMARY option output for basic IMS formatting and in response to the selection of a BPE or CQS jobname in the BPE initialization panel.

Using IMS Enhanced Dump Analysis

If you select option E from the IMS dump formatting primary menu, you see the IMS Enhanced Dump Formatting Menu, shown in Figure 45.

```
--------------- IMS ENHANCED DUMP FORMATTING MENU -------
Option ===>

       1  BROWSE    - Browse dump dataset (IPCS norm)
       2  DB        - Full Function Data Base
       3  FP        - Fast Path Data Base
       4  TM        - Transaction Management and DC
       5  SYS       - Systems
       T  TUTORIAL  - IMS Dump Formatter Tutorial
       X  EXIT      - Exit EDA dump formatting menu
```

Figure 45. IMS Enhanced Dump Formatting MenuIMS Enhanced Dump Formatting Menu

In this panel, the control blocks are organized by function for ease of use. For example, EPST (the extended partition specification table) would be located under option 3 for Fast Path. To review tutorial information about the dump formatter and about how to use the filtering tool, select option T. When you select options 2, 3, 4, or 5, you can use a filtering tool to identify filtering criteria. An example of a filtering panel is shown in Figure 46.

```
 ------------------ Generic Filtering Panel ---------------------
Explanation of the fields:
    Offset  (required)     - Offset of the field in the block.
                             (hex)
    Length  (default = 1)  - Length of field in the control
                             block. (decimal)
    Cond    (default = EQ) - Type of compare to be done. (EQ,NE,
                             GT,GE,LT,LE)
    Bit     (default = N)  - Should comparison be a bit mask?
                             (Y or N)
    Type    (default = X)  - Is the value type decimal, hex, or
                             char (D,X,A)?
    Value   (required)     - Value of the field to be compared
                             at given offset.
    Qual                   - Qualify filter to search in
                             sub-blocks.
    AND/OR                 - How to combine multiple conditions.
                             If blank, only the first condition
                             will be executed.
                                       (up to four conditions allowed).
```

Figure 46. Sample Filtering PanelSample Filtering Panel

When you open the generic filtering panel, default values are automatically filled in, as shown in Figure 46; however, you can overwrite them. For example, you can select criteria that presents two separate conditions:

- You want all the blocks starting at OFFSET 1C that have a value of X'08.'
- You want all the blocks starting at OFFSET A4 that have a non-zero value.

By selecting AND, you indicate that both conditions must be true. These values are shown in Figure 47.

```
<=====  AND/OR  (A/0)        QUAL =====>
```

Figure 47. Sample Filtering Criteria

Formatting IMS Dumps Online

One of the tools available for problem diagnosis is the IMS formatted dump, which formats the control blocks and data areas in an IMS region.

When an abnormal termination occurs and dumping is to be performed, CSECT DFSABND0 gets control from the SCP and gives control to IMS routines to do the dumping. To assist you in rapidly locating areas that are dumped, eye-catchers are supplied in the formatted dump. See “Eye-catchers” on page 146 for eye-catcher examples.

Exception: The BPE and CQS address space does not provide any online dump formatting output.

Formatted Dump for the CTL Address Space

The following is a list of the control address space areas that are dumped (in the order in which they are dumped) and, where applicable, the DSECT mapping macros that are most useful in analyzing them. For a list of the areas dumped when LSO=S, see “Formatted Dump for the DL/I Address Space” on page 159. Descriptive information has been added for some control blocks where it would be useful.

Diagnostic Area
Contains the PSW, system and user completion codes, save area ID of the module that was executing, and registers in use when abnormal termination occurred.

Instruction Area
Contains the area of storage from 128 bytes before to 128 bytes after the address of the failing instruction in the PSW.

System Diagnostic Work Area
The mapping DSECT is IHASDWA.

U0113 Area
Present when an abend caused the dump.

Referenced Sap
The mapping DSECT is ISAP.

System Contents Directory
The mapping DSECT is ISCD.

SCD Extension
The mapping DSECT is DBFESCD.

SCD Latch Extension
The mapping DSECT is ISCD.

Scheduler Sequence Queues
Controls the status of each region. The mapping DSECT is ISCD.

FP ESCD
The mapping DSECT is DBFESCD.

Control Block Table
Contains entries of control blocks that macro DFSCBTS uses for tracking. The mapping DSECT is DFSCBTS.

Save Area Prefix
All SAPs are SNAPed except those owned by the DL/I address space. Each SAP is followed by its save area set. At the end of this section, all of the SAP IPAGES are dumped.

IMS Task Dispatch Work Area
The mapping DSECT is IDSPWRK.

DBRC Task Dispatch Work Area
If present in the system, it is mapped.

IMS Control Task Dispatch Work Area
Contains the same information as the IMS log task dispatch work area.

Dependent Region Dispatch Work Area
For every dependent region in IMS, the dispatcher work area is mapped.

Dispatcher Trace Data
DSECT IDSPWRK contains the function codes associated with the dispatcher and an explanation of each code.

Scheduler Trace Data
Scheduler trace data is mapped by DFSSCHED. The trace entries contain scheduler function codes.

Latch Trace Data
The trace entries contain latch and unlatch function codes. The mapping DSECT is IDLIVSAM TRACENT.

System PSTs
These are system work areas for any online or batch region. The mapping DSECT is IPST.

Checkpoint ID Table
The mapping DSECT is BCPT.

LCRE
The mapping DSECT is DFSLCRE.

SIDX
The mapping DSECT is DFSSSIE.

RRE
The mapping DSECT is DFSRRE.

Log Control Directory
Contains information about the IMS log, for example:

DCB1—the primary log DCB

DCB2—the secondary log DCB (if dual logs were specified)

Log ITASK—the status information

The mapping DSECT is LCDSECT.

Log Buffers
Each log buffer contains buffer information and the log control DECB. The mapping DSECT is LCDSECT.

Log Trace
Contains entries which show IMS internal logging activity if the log trace is active. The trace entries are described by the “IDLIVSAM TRACENT” macro.

Open Record
Contains the type 06 log record. The mapping DSECT is ILOGREC.

Control Record
Contains the type 42 log record. The mapping DSECT is ILOGREC.

Monitor Log Directory
Contains the same information as the log control directory and is used for logging data to the IMS Monitor data set.

DLOG Trace Data
Trace table used to show IMS logging activity. The mapping DSECT is ILOGREC (67FA).

SUBS Trace Data
Trace table used by IMS to show IMS activity in attaching or detaching subsystems. The mapping DSECT is ILOGREC (67FA).

Global ESET Block
The mapping DSECT is DFSGESE.

PSB Directory
A SNAP of the PSB directory. The mapping DSECT is PDIR.

DMB Directory
A SNAP of the DMB directory. The mapping DSECT is DDIR.

Fast Path Trace

Dependent Region PST
See Dependent Region PST Formatting on page 149 for a list of the areas formatted here.

OSAM I/O Control Blocks
The system attempts to dump the IOSB and IOMA blocks.

Sequential Buffering Blocks
Sequential Buffering information is grouped into the following three sections. (See the explanation of the (SB) FMTIMS option on page 142 for a complete list of the blocks dumped in each section.)

1. Subsystem Overview for Sequential Buffering
2. PST Overview of Sequential Buffering control blocks
3. Formatted Sequential Buffering control blocks

DEDB Formatting

Fast Path EMH Formatting

Fast Path MDSB Formatting

Data Communication Control Blocks [3]
For each CLB (line), all the control blocks associated with that line are formatted.

CLB [3]
The mapping DSECT is ICLI CLBBASE=0.

CTB [3]
The mapping DSECT is ICLI CTBBASE=0.

Input Buffer [3]
A SNAP of the input buffer, if input is active.

Output Buffer [3]
A SNAP of the output buffer, if output is active.

CCB [3]
Present if a conversation is active or held. The mapping DSECT is ICLI CCBBASE=0.

CIB [3]
Present if MFS is in use. The mapping DSECT is ICLI CIBBASE=0.

Communication Terminal Table [3]
Defines terminal characteristics. The mapping DSECT is ICLI CTTBASE=0.

SPQB Entries [3]
Entries on the subpool queue block chain. Unallocated CNTs are also formatted here.

SMB Table [3]
This table defines transaction characteristics in the IMS system. The mapping DSECT is IAPS SMBBASE=0.

Queue Manager Pool Prefix and Buffers [3]
The mapping DSECTs are ICLI POOLBASE=0 and ICLI BFRBASE=0.

Buffer Prefix List [3]
Contains the address of each buffer's prefix, status byte, and first and last pending and current DRRN.

QPOOL Prefix [3]
Contains the main QPOOL prefix formatted. The mapping DSECT is QPOOL.

IRLM Control Blocks
The IRLM Subsystem RLMCB block are formatted here if the IMS system is running with IRLM.

Format/Dump/Delete List
Contains module names, module IDs, and module dump data that are not in the storage dump listing.

Formatted Dump for the DL/I Address Space

The following is a list of the areas within the DL/I address space that are dumped when the LSO=S option is active. Descriptive information has been added for some control blocks where it would be useful.

System Contents Directory
The mapping DSECT is ISCD.

SCD Latch Extension
The mapping DSECT is ISCD.

Scheduler Sequence Queues
Controls the status of each region. The mapping DSECT is ISCD.

Save Area Trace

Save Area Prefix
All SAPs belonging to the DL/I address space are SNAPed. A SAP is marked "ACTIVE" if the ITASK associated with it is active. Each SAP is followed by its save area set. At the end of this section, all of the SAP IPAGES are dumped.

DLS Task Dispatch Work Areas
The mapping DSECT is IDSPWRK.

DBRC Task Dispatch Work Area
If present in the system, it is mapped.

Dependent Region Dispatch Work Area
For every dependent region in IMS, the dispatcher work area is mapped.

Dispatcher Trace Data
DSECT IDSPWRK contains the function codes associated with the dispatcher and an explanation of each code.

3. These areas are not dumped in a DBCTL environment.

Latch Trace Data
The trace entries contain latch and unlatch function codes. The mapping DSECT is IDLIVSAM TRACENT.

System PSTs
These are system work areas for any online or batch region. The mapping DSECT is IPST.

PSB Directory
A SNAP of the PSB directory. The mapping DSECT is PDIR.

DMB Directory
A SNAP of the DMB directory. The mapping DSECT is DDIR.

Intent List
This is a SNAP of the intent list.

Partition Specification Table
Formats the PST. The mapping DSECT is IPST.

PDIR
Formats the PDIR, whose address is in the PST. The mapping DSECT for PDIR is PDIR.

PSB Prefix
A SNAP of the PSB prefix, which contains the following:

Index Maintenance Work Area

Index I/O Work Area

Segment Work Area

I/O Work Area

SSA Work Area

User PARMS Area

Buffer Handler Pool
The system attempts to format buffer handler blocks in the order in which they are chained on the queue. However, if an error is encountered during the formatting, the entire pool is dumped as is (unchained).

The pool contains the following:

BFSP	Formats the buffer pool prefix. The mapping DSECT is BFSP.
BFUS	Formats the subpool prefix. The mapping DSECT is BFUS.
RPLI	Formats the DL/I RPL block. The mapping DSECT is RPLI.
DL/I Data	A dump of the DL/I, lock activity and program isolation trace table. The mapping DSECT is IDLIVSAM TRACENT.
Lock Activity Trace Data	See DL/I DATA.
Program Isolation Data	Includes the QEL, QCB, and REQ areas. The mapping DSECT is XC00.

OSAM Control Blocks
The system attempts to follow the main pool, the subpool header, and the buffer prefix, and to dump the buffer. However, if an error is encountered during formatting, the entire buffer pool is SNAPed from the last valid subpool address.

The pool contains the following:

MAINPOOL	Formats the main pool header. The mapping DSECT is IBPOOL.
SUBPOOL	Formats the subpool header. The mapping DSECT is ISUBPL.
Buffer Prefix	Formats the buffer prefix. The mapping DSECT is IBFPRF.

Buffer Physical data not mapped.

OSAM I/O Control Blocks
The system attempts to dump the IOSB and IOMA control blocks. The mapping DSECT is QPOOL.

Sequential Buffering Blocks
Sequential Buffering information is grouped into the following three sections. (See the explanation of the (SB) FMTIMS option on page 142 for a complete list of the blocks dumped in each section.)

1. Subsystem Overview for Sequential Buffering
2. PST Overview of Sequential Buffering control blocks
3. Formatted Sequential Buffering control blocks

Fast Path DEDB Formatting

Fast Path EMH Formatting

Fast Path MDSB Formatting

IRLM Control Blocks
The IRLM Subsystem RLMCB block is formatted here if the IMS system is running with IRLM.

Format/Dump/Delete List
Contains module names, module IDs, and module dump data that are not in the storage dump listing.

SNAP Call Facility

The SNAP call facility (DFSERA20) produces SNAPs of DL/I control blocks for:

- External DL/I SNAP calls. The DL/I test program, DFSDDLT0, issues SNAP calls when it detects unequal conditions based on compare statements.
- Exceptional conditions, such as:
  Pseudoabends in DL/I modules.
  Message or batch-message region abends.
- Internal SNAP requests from DL/I modules.
- SNAP specific requests from other IMS modules.

GSAM modules issue SNAP calls for GSAM databases. See "GSAM Control Block Dump—DFSZD510" on page 246 for a description of the GSAM SNAP.

When a SNAP call is performed for a Fast Path region abend, DFSERA20 bypasses some dumps.

For a Fast Path database (an MSDB or DEDB), DFSERA20 bypasses the DMB dump.

For a DB-PCB that refers to a Fast Path database, DFSERA20 bypasses the DMB, DB-PCB, JCB, and SDB dumps.

SNAP Output

SNAP output consists of buffer pools and all PSB-related control blocks. Optionally, you can request subpools 0-127 in addition to the buffers and blocks.

SNAP output for exceptional conditions is always directed to the IMS log. In all other cases, IMS sends SNAP output to a data set identified on the PRINTDD DD statement. If this data set is not already open, it is opened and closed for each SNAP request. If you do not supply a PRINTDD statement, IMS sends the SNAP output to the IMS log as X'67FD' log records. When neither a SNAP data set nor the IMS log can be used for SNAPs, all SNAP actions are bypassed.

The File Select and Formatting Print utility (DFSERA10) extracts X'67FD' log records, and the exit routine (DFSERA30) formats them. For information about the File Select and Formatting Print utility, see *IMS Version 7 Utilities Reference: System*.

Status codes are not set for SNAP calls.

Common Trace Table Interface

The common trace table interface consists of the traces shown in Table 25. For each trace, Table 25 shows the trace identifier, the events traced, and, if the trace is documented in this manual, the page where you can find more information. You use the trace identifier as an eye-catcher to locate a trace in a dump.

Table 25. Trace Tables in the Common Trace Interface

Trace	ID	What Is Traced	Where Described
DASD log trace	DG	DASD logging	on page 166
Dispatcher trace (online only)	DS	Dispatcher activities	“Dispatcher Trace” on page 167
DL/I and lock	DL	DL/I calls, DL/I buffer handler, DL/I OPEN/CLOSE, Delete/Replace, HD space management, lock activity using PI or IRLM, OSAM, DFP interface, ABENDU0427	“DL/I Trace” on page 214
External subsystem trace (online only)	SU	Subsystem activities	“External Subsystem Trace” on page 176
Fast Path	FP	Fast Path activity	Not documented
Force trace	FO	Internal trace for IMS initialization	Not documented
Intercommunications trace	IC	VTAM exit activity	“Starting the Trace” on page 253
Latch trace (online only)	LA	Latch activities	“Latch Trace” on page 193
Log router trace	LR	Log router activity	“Log Router Trace Data” on page 412
LU trace	LU	LU 6.2 activity	“LU Manager Trace” on page 301
| Online Recovery System (ORS) trace	OR	ORS activity	Not documented
| OTMA trace	OA	OTMA activity	“OTMA Trace” on page 326
Queue manager trace	QM	Queue manager activity	“Queue Manager Trace” on page 197
Scheduler trace (online only)	SC	Scheduler activities	“Scheduler Trace” on page 189
Shared queues interface trace	SQ	Shared queues interface activities.	“Shared Queues Interface Trace” on page 202
Storage Manager trace	SM	Storage Manager activities	“Storage Manager Trace” on page 192

Finding the Trace Tables in a Dump

If you do not choose to write the trace to the log data set, IMS formats trace tables as part of an IMS dump.

Figure 48 on page 163 explains how to find the location of each of the traces in a dump.

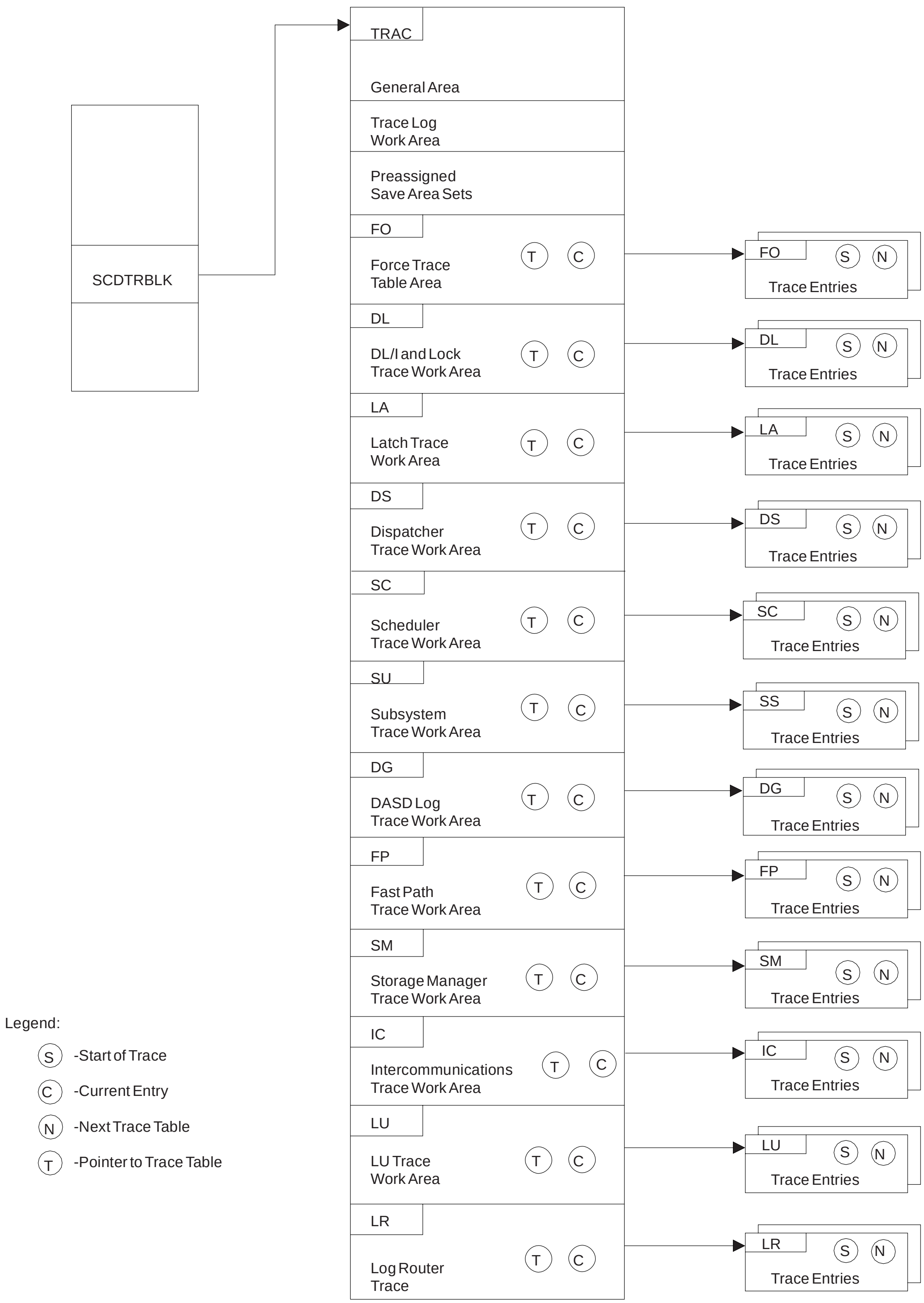


Figure 48. How to Locate Trace Tables

Format of Trace Records

By examining the trace records, you can determine the function that was being traced as well as the order in which a series of system operations took place. In the example trace record in Figure 49 below, the number in the trace sequence field in each entry identifies where that trace entry fits in the sequence of system operations. In addition, each trace entry provides pertinent information about that function.

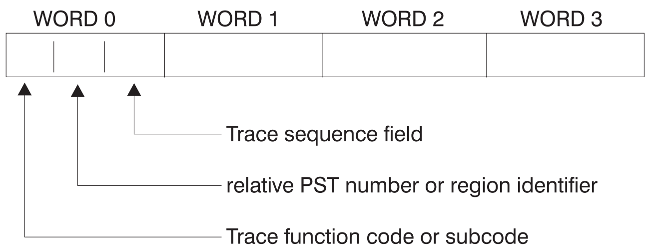


Figure 49. General Trace Record Format

You can find the format of the trace entries by assembling macro IDLIVSAM TRACENT. Assembling IDLIVSAM after each system definition ensures that you have a current mapping of the trace record formats.

IMS Trace Function Codes

The common trace interface captures information for a given trace function code. Table 26 lists some of the important functions traced and their location in trace tables. These function codes are a subset of codes and are listed here only for you to use with the trace examples given in “Dispatcher Trace” on page 167, “Scheduler Trace” on page 189, and “External Subsystem Trace” on page 176. You can find a one-line description of each trace code in macro DFSTRAE0.

Table 26. Trace Function Codes

Trace Table	Function Code	Description
DL/I and lock	X'0C'	DL/I OPEN/CLOSE for each data set
	X'30'	IWAIT called with IXCTL=YES option
	X'31'	Get space for the segment
	X'32'	Free space for the segment
	X'34'	Get space close to root anchor
	X'35'	HD space management GET /ERE local serialization lock
	X'36'	HD space management release local serialization lock /ERE
	X'60'	(OSAM) I/O operation initiated
	X'61'	(OSAM) I/O operation posted
	X'62'	(OSAM) OPEN/CLOSE/EOV complete
	X'69'	Sequential buffering: invalidate SB buffers
	X'6A'	Sequential buffering: buffering evaluation
	X'6B'	Sequential buffering: description why SB was/was not used
	X'6C'	Sequential buffering: refresh SB buffers after a write
	X'6F'	Sequential buffering: search/read call issued by OSAM Buffer Handler
	X'80'	Database authorization request
	X'81'	Database change authorization request
	X'82'	Database re-authorization request
	X'AA'	DL/I call analyzer entry for each database call
	X'AB'	(VSAM) ABEND U0427
	X'B1'	Demand space set by backout or DELETE/REPLACE
	X'B2'	Free space for backout
	X'C4'	DELETE/REPLACE
	X'C7'	(PI) Exclusive control deadlock detection
	X'C8'	Lock request manager (DFSLMGR0) entry
	X'C9'	Lock request manager (DFSLMGR0) exit
	X'CA'	(PI) request trace entry

Table 26. Trace Function Codes (continued)

Trace Table	Function Code	Description
	X'CA'—X'08'	(PI) DL/I call trace entry
	X'CB'	(PI) lock elapsed time entry
	X'CC'	Lock request handler (DFSLRH00)
	X'CF'	I/O Toleration (DFSTOPR0)
	X'D0'	IRLM NOTIFY sent
	X'D1'	IRLM NOTIFY received
	X'D2'	IRLM status exit
	X'D3'	IRLM deadlock exit
	X'D5'	Sysplex data sharing
	X'DA'	VSAM JRNAD or UPAD exit
	X'DB'	Search pool for record in range (buffer handler)
	X'DD'	Release record ownership (buffer handler)
	X'DE'	Retrieve buffer pool statistics (buffer handler)
	X'DF'	VSAM verify
	X'E0'	VSAM PUT
	X'E1'	Block locate (buffer handler)
	X'E2'	Byte locate (buffer handler)
	X'E4'	Create new ESDS/OSAM LRECL (buffer handler)
	X'E5'	Write LRECLs for user (purge) (buffer handler)
	X'E6'	Mark record altered (buffer handler)
	X'E9'	Free space in buffer pool (BFPL) (buffer handler)
	X'EA'	Perform background write function (buffer handler)
	X'EB'	Byte locate and mark altered (buffer handler)
	X'EC'	Mark buffers empty (BFPL) (buffer handler)
	X'ED'	Checkpoint (buffer handler)
	X'EE'	Batch STAE purge at ABEND (buffer handler)
	X'EF'	OSAM buffer forced write (buffer handler)
	X'F0'	Retrieve first LRECL by key (buffer handler)
	X'F1'	Erase logical record (buffer handler)
	X'F2'	Retrieve by key EQ or GT (buffer handler)
	X'F3'	Retrieve key EQ or GT—Repair CI (buffer handler)
	X'F4'	Retrieve by key record to chain from insert logical record (KSDS) (buffer handler)
	X'F8'	Retrieve next sequential root by key (buffer handler)
	X'F9'	Position by key for image copy (buffer handler)
	X'FA'	Get next record for image copy (buffer handler)
Dispatcher	X'01'	FRR driven attempting to SCHEDULE a RESUME SRB in IPOST common (DFSIPOTC)
	X'02'	ITASK started (created)
	X'03'	ITASK terminated
	X'04'	IWAIT called
	X'05'	ITASK reinstated
	X'06'	IPOST called
	X'07'	IXCTL called
	X'08'	ISWITCH 'TO' invoked
	X'09'	Un-initialize ECB called
	X'0A'	Dependent region dispatch reattach
	X'0B'	Process IMS TCB signoff
	X'0C'	Reserved — used by DL/I Open Close
	X'0D'	INITECB called
	X'0E'	Memory change done via PC/PT
	X'0F'	Dispatcher abend issued

Table 26. Trace Function Codes (continued)

Trace Table	Function Code	Description
	X'10'	Cross memory ISWITCH TO=XM or TO=HOME
	X'11'	Cross memory state change
	X'12'	DFSKPXT store POST code in ECB
	X'13'	DFSKPXT called (MVS branch-entry local POST)
	X'14'	DFSCIR called to create an ITASK
	X'15'	DFSKPXT issued MVS branch-entry local POST
	X'16'	Post exit posted ECB enqueue
	X'17'	Post exit resume target IMS TCB
	X'18'	IPOST common store post code in ECB
	X'19'	IPOST common posted ECB enqueue
	X'1A'	IPOST common resume target IMS TCB
	X'1B'	INITECB ECB store results
	X'1C'	INITECB posted ECB enqueue
	X'1D'	Suspend back out resume issued
	X'1E'	SRB scheduled for alternate IPOST
	X'1F'	IPOST called ('SAP=')
	X'20'	Dependent region shutdown ISWITCH
	X'21'	Entry to POST-Exit routine
	X'22'	Reserved
	X'23'	ISERWAIT called
	X'24'	ISWITCH 'TO' with stack invoked
	X'25'	Reserved
	X'26'	Branch entry SCP post
	X'27'	Suspend IMS TCB
	X'28'	Dependent region open dispatcher — sign on
	X'29'	ISWITCH TO=UNSTACK
	X'2A'	IMS list post called
	X'2B'	SCP WAIT issued
	X'2C'	SCP WAIT completed
	X'2D'	ISWITCH 'RET' invoked
	X'2E'	Shutdown ISWITCH reinstated
	X'2F'	Dependent region open dispatcher — TCB switch
Scheduler	X'41'	Scheduling starts
	X'42'	Block mover
	X'43'	Scheduling ends
	X'44'	IRC started
	X'45'	TMS00 started
	X'46'	TMS00 finished
	X'47'	APPC extract call made
	X'48'	Scheduling failed
Queue Manager	X'4E'	Information related to the queue manager
DASD log [1]	X'50'	Logical logger trace entry
	X'51'	Physical logger master ITASK trace entry
	X'52'	Physical logger buffer ITASK trace entry
	X'53'	Physical logger setup ITASK trace entry
	X'54'	Physical logger WADS ITASK trace entry
	X'55'	Physical logger READ ITASK trace entry
External subsystem	X'57'	Created by the module that operates in the IMS control region
	X'58'	Created by the module that operates in the IMS dependent region

Table 26. Trace Function Codes (continued)

Trace Table	Function Code	Description
Storage Manager	X'5F'	Storage Manager trace entry written on pool allocation Buffer Get and Buffer release (CESS, CIOP, EMHB, FPWP, HIOP, SPAP, LUMC, LUMP)
Latch	X'70'	Information related to the latch manager and the use manager
	X'76'	Reserved
Log Router	X'38'	Created by various log router functions

Note:

1. For a detailed description of the log trace entries, refer to a listing of the IDLIVSAM TRACENT macro.

For examples of these trace tables, see "Dispatcher Trace," "Scheduler Trace" on page 189, "External Subsystem Trace" on page 176, and "Storage Manager Trace" on page 192.

Dispatcher Trace

When you use the `/TRACE SET ON TABLE DISP` command, IMS enables the dispatcher trace to an internal table. This internal table is formatted in any IMS-formatted dump. When you use OPTION LOG, IMS sends the entries to the log as type X'67FA' records. You can select and format these log entries by using the utility DFSERA10 with exit DFSERA30.

The following figure shows the general format of a dispatcher trace entry:

WORD 0	WORD 1	WORD 2	WORD 3	WORD 4	WORD 5	WORD 6	WORD 7
I | T | SEQ NUM							TIME STAMP

Figure 50. Dispatcher Trace Record Format

where	represents
I	One-byte trace ID field. This byte indicates the type of the trace entry.
T	One-byte TCB ID. This byte indicates the IMS TCB type which made the trace entry.
SEQ NUM	Two-byte trace sequence number assigned by the IMS trace component.
TIME STAMP	Bytes 3 through 6 of the system clock (STCK) at the time the trace entry was created.

Words 1 through 6 contain data specific to each trace entry, as described below: The letter `A` followed by parentheses () indicates "address of" in all dispatcher trace entries listed below.

```
TRACE ID   = X'01'
 DESC = FRR driven attempting to schedule a RESUME SRB
             in IPOST common (DFSIPOTC)
      word 1 – A(Target ECB being IPOSTed). If high X'80'  on,
                 this indicates recursive FRR entry
      word 2 – SAPCNTRL field from target ECB's SAP
      word 3 – Abend code
      word 4 – A(target dispatcher work area)
      word 5 – IPOST common caller's return address
      word 6 – IPOST common caller's R13

TRACE ID   = X'02'
 DESC      = ECB dispatch – ITASK started (created)

      word 1 – A(ITASK ECB)
```

word 2 – ECB contents
word 3 – A(ITASK SAP)
word 4 – EPFFLAGS field from ECB prefix
word 5 – A(CULE) if present in ECB prefix
word 6 – A(Routine to get control)

TRACE ID = X'03'
DESC = ECB dispatch – ITASK terminated

word 1 – A(ITASK ECB)
word 2 – ECB contents
word 3 – A(ITASK SAP)
word 4 – EPFFLAGS field from ECB prefix
word 5 – A(CULE) if present in ECB prefix
word 6 – 0

TRACE ID = X'04'
DESC = IWAIT called

word 1 – A(ITASK ECB)
word 2 – ECB contents prior to IWAIT
word 3 – IWAIT return address
word 4 – 0
word 5 – 0
word 6 – SAPCNTRL contents

TRACE ID = X'05'
DESC = ECB dispatch – ITASK reinstated

word 1 – A(ITASK ECB)
word 2 – ECB contents
word 3 – SAPCNTRL field from ITASK's SAP
word 4 – EPFFLAGS field from ECB prefix
word 5 – Reinstate address (return address)
word 6 – 0

TRACE ID = X'06'
DESC = IPOST called

word 1 – A(POSTer's ECB) (A(TCB) if ITASK=NO)
word 2 – IPOST return
word 3 – A(ECB to be POSTed)
word 4 – Contents of ECB before IPOST
word 5 – POST code at entry to IPOST (may be complimented)
word 6 – 0

TRACE ID = X'07'
DESC = IXCTL called

word 1 – A(Current ITASK ECB)
word 2 – A(IXCTL target ECB)
word 3 – IXCTL return address
word 4 – A(CULE) from current ECB prefix
word 5 – 0
word 6 – 0

TRACE ID = X'08'
DESC = ISWITCH TO= invoked

word 1 – A(Current ECB)
word 2 – ISWITCH return address
word 3 – A(target dispatcher work area)
word 4 – SAPCNTRL field from ECB's SAP
word 5 – SAPXFLAG contents
word 6 – 0

TRACE ID = X'09'
DESC = UN-INITIALIZE ECB called

word 1 – A(Target ECB)
word 2 – UNINIT return address

```
      word 3 – UNINIT return code
      word 4 – EPFFLAGS from ECB prefix
      word 5 – ECB contents
      word 6 – 0
 TRACE ID   = X'0A'
  DESC      = Dependent region reattach

      word 1 – A(Related PST)
      word 2 – A(Dependent region dispatcher work area)
      word 3 – SAPCNTRL field from PST's SAP
      word 4 – 0
      word 5 – 0
      word 6 – 0
 TRACE ID   = X'0B'
  DESC      = Process IMS TCB signoff

      word 1 – A(Related PST)
      word 2 – A(Released dispatcher work area)
      word 3 – Signoff return address
      word 4 – 0
      word 5 – 0
      word 6 – 0
 TRACE ID   = X'0D'
  DESC      = INITECB called

      word 1 – A(Current ECB)
      word 2 – INITECB return address
      word 3 – A(ECB being initialized)
      word 4 – Contents of ECB before being initialized
      word 5 – 0
      word 6 – 0
 TRACE ID   = X'0E'
  DESC      = Memory change done via PC/PT
      word 1 – A(Current ECB) (X'80' on=PC; off=PT)
      word 2 – Old primary ASID | Secondary ASID
      word 3 – If Word 1 indicates PT: PKM ASID for PT
               If Word 1 indicates PC: PC # issued
      word 4 – A(Current dispatcher work area)
      word 5 – 0
      word 6 – 0
 TRACE ID   = X'0F'
  DESC      = Dispatcher ABEND issued ("other diagnostics"
              dependent on ABEND issuer)
      word 1 – A(Current ECB)
      word 2 – Other diagnostics
      word 3 – ABEND code | reason code
      word 4 – Other diagnostics (usually the dispatcher work area
               address of the abending TCB)
      word 5 – Other diagnostics
      word 6 – Other diagnostics
 TRACE ID   = X'10'
  DESC      = Cross memory ISWITCH TO=XM or TO=HOME

      word 1 – A(Current ECB)
      word 2 – ISWITCH return address
      word 3 – Target code (00=HOME, 01=CTL, 02=DLI)
      word 4 – SAPCNTRL field from ECB's SAP
      word 5 – Home ASID of target | Primary ASID of target
      word 6 – SAPXFLAG contents
 TRACE ID   = X'11'
  DESC      = Cross memory state change

      word 1 – A(Current ECB)
      word 2 – Old primary ASID | Secondary ASID
```

```
      word 3 – New primary ASID | Secondary ASID
      word 4 – A(current dispatcher work area)
      word 5 – 0
      word 6 – 0
```

TRACE ID = X'12'
DESC = DFSKPXT–POST code stored in ECB (ECB was not waiting)

```
      word 1 – A(ECB) to be POSTed
      word 2 – POST code
      word 3 – Contents of ECB on prior to store
      word 4 – 0
      word 5 – 0
      word 6 – 0
```

TRACE ID = X'13'
DESC = DFSKPXT–Special MVS branch-entry POST call

```
      word 1 – A(Caller's TCB) (0 if SRB)
      word 2 – Caller's return address
      word 3 – A(ECB) to be POSTed
      word 4 – Caller's home ASID
      word 5 – 0
      word 6 – 0
```

TRACE ID = X'14'
DESC = DFSCIR called to create an ITASK

```
      word 1 – A(ECB) or -A(ECB list)
      word 2 – ITASK type code
      word 3 – DFSCIR return address
      word 4 – A(ITASK main program)
      word 5 – 0
      word 6 – 0
```

TRACE ID = X'15'
DESC = DFSKPXT issued branch-entry MVS POST (local)

```
      word 1 – A(ECB) to be POSTed
      word 2 – ECB POST code
      word 3 – ECB contents prior to the POST
      word 4 – 0
      word 5 – 0
      word 6 – 0
```

TRACE ID = X'16'
DESC = POST exit POSTed ECB enqueue

```
      word 1 – A(ECB) being POSTed
      word 2 – ECB POST code
      word 3 – Previous POST queue header contents
      word 4 – 0
      word 5 – 0
      word 6 – 0
```

TRACE ID = X'17'
DESC = POST exit RESUME target IMS TCB

```
      word 1 – A(TCB)  (SRB=0)
      word 2 – Home ASID | Primary ASID
      word 3 – Target TCB's ASID
      word 4 – 0
      word 5 – 0
      word 6 – 0
```

TRACE ID = X'18'
DESC = IPOST common store POST code in ECB (ECB was not waiting)

```
      word 1 – A(ECB) being IPOSTed
      word 2 – POST code
```

```
      word 3 – ECB contents prior to the IPOST
      word 4 – A(ECB's dispatcher work area)
      word 5 – IPOST common caller's return address
      word 6 – 0
```

TRACE ID = X'19'
DESC = IPOST common POSTed ECB enqueue

```
      word 1 – A(ECB) being enqueued
      word 2 – ECB POST code
      word 3 – Previous POSTed queue header contents
      word 4 – A(ECB's dispatcher work area)
      word 5 – IPOST common caller's return address
      word 6 – 0
```

TRACE ID = X'1A'
DESC = IPOST common RESUME target IMS TCB

```
      word 1 – A(current TCB) (0=SRB)
      word 2 – Home ASID or Primary ASID
      word 3 – Target TCB's home ASID
      word 4 – A(resumed TCB's dispatcher work area)
      word 5 – 0
      word 6 – 0
```

TRACE ID = X'1B'
DESC = INITECB ECB store results

```
      word 1 – A(ECB) being initialized
      word 2 – WAIT code being stored into ECB
      word 3 – ECB contents prior to INITECB store
      word 4 – 0
      word 5 – 0
      word 6 – 0
```

TRACE ID = X'1C'
DESC = INITECB POSTed ECB enqueue

```
      word 1 – A(ECB) being initialized
      word 2 – ECB POST code
      word 3 – Previous POSTed queue header contents
      word 4 – 0
      word 5 – 0
      word 6 – 0
```

TRACE ID = X'1D'
DESC = SUSPEND back out RESUME issued

```
      word 1 – POSTed queue header contents
      word 2 – Home ASID | Primary ASID
      word 3 – A(SRB) (0 = no SRB)
      word 4 – 0
      word 5 – 0
      word 6 – 0
```

TRACE ID = X'1E'
DESC = SRB scheduled for alternate IPOST

```
      word 1 – A(ECB) to be IPOSTed
      word 2 – Primary ASID | target ASID
      word 3 – A(IPOST SRB) (0 if MVS branch entry XM-POST)
      word 4 – A(current ASCB)
      word 5 – POST code
      word 6 – 0
```

TRACE ID = X'1F'
DESC = IPOST called with TOSAP= option

```
      word 1 – A(Poster's ECB) (A(TCB) if ITASK=NO)
      word 2 – IPOST return address
```

```
      word 3 – A(ECB to be POSTed)
      word 4 – 0
      word 5 – POST code at entry to IPOST (may be complimented)
      word 6 – 0
```

TRACE ID = X'20'
DESC = Dependent region shutdown ISWITCH

```
      word 1 – A(Related PST)
      word 2 – A(Special exit)
      word 3 – SAPCNTRL field from PST's SAP
      word 4 – A(Home dispatcher work area)
      word 5 – 0
      word 6 – 0
```

TRACE ID = X'21'
DESC = Entry to Post-Exit Routine

```
      word 1 – A(ECB) being POSTed
      word 2 – ECB Contents
      word 3 – EPFFLAGS from ECB prefix
      word 4 – 0
      word 5 – 0
      word 6 – 0
```

TRACE ID = X'22'
DESC = ABTERM ISWITCH entered

```
      word 1 – A(ECB) to be switched
      word 2 – ECB contents
      word 3 – SAPCNTRL contents
      word 4 – SAPCNTL2 contents
      word 5 – Posted Q contents
      word 6 – SAPCMEM | SAPCFLGS
```

TRACE ID = X'23'
DESC = ISERWAIT called

```
      word 1 – A(ITASK ECB)
      word 2 – ECB contents prior to ISERWAIT
      word 3 – ISERWAIT return address
      word 4 – 0
      word 5 – 0
      word 6 – SAPCNTRL contents
```

TRACE ID = X'24'
DESC = ISWITCH TO=, STACK=YES called

```
      word 1 – A(Current ECB)
      word 2 – ISWITCH return address
      word 3 – A(Target dispatcher work area)
      word 4 – SAPCNTRL field from ITASK's SAP
      word 5 – SAPXFLAG contents
      word 6 – 0
```

TRACE ID = X'25'
DESC = POST ABTERM ISWITCH

```
      word 1 – A(ECB) to be switched
      word 2 – ECB POST code
      word 3 – previous posted Q contents
      word 4 – A(Target dispatcher work area)
      word 5 – IPOTC/IPEXT caller's return
      word 6 – 0
```

TRACE ID = X'26'
DESC = Branch entry SCP POST

```
      word 1 – A(ECB) to be POSTed
      word 2 – ECB POST code
```

```
      word 3 – A(ASCB) of ECB's address space
      word 4 – A(Current TCB)
      word 5 – A(Current ASCB)
      word 6 – 0
```

TRACE ID = X'27'
DESC = SUSPEND IMS TCB

```
      word 1 – A(Related PST) (0 if not a dependent region/LSD)
      word 2 – Home ASID | Primary ASID
      word 3 – A(Suspended dispatcher work area)
      word 4 – 0
      word 5 – 0
      word 6 – 0
```

TRACE ID = X'28'
DESC = Dependent region open dispatcher–signon

```
      word 1 – A(Related PST)
      word 2 – Home ASID
      word 3 – A(Current TCB)
      word 4 – 0
      word 5 – 0
      word 6 – 0
```

TRACE ID = X'29'
DESC = ISWITCH TO=UNSTACK

```
      word 1 – A(Current ECB)
      word 2 – ISWITCH return address
      word 3 – X'80000000'
      word 4 – SAPCNTRL field from ECB's SAP
      word 5 – SAPXFLAG contents
      word 6 – 0
```

TRACE ID = X'2A'
DESC = IMS list IPOST called

```
      word 1 – A(ECB) to be IPOSTed
      word 2 – List IPOST return address
      word 3 – A(POST list)
      word 4 – 0
      word 5 – 0
      word 6 – 0
```

TRACE ID = X'2B'
DESC = SCP WAIT issued (SVC WAIT)

```
      word 1 – A(WAIT ECB)
      word 2 – SCP WAIT return address
      word 3 – A(Current TCB)
      word 4 – ECB contents prior to WAIT
      word 5 – 0
      word 6 – 0
```

TRACE ID = X'2C'
DESC = SCP WAIT complete (SVC WAIT)

```
      word 1 – A(WAIT ECB)
      word 2 – ECB POST code
      word 3 – A(Current TCB)
      word 4 – 0
      word 5 – 0
      word 6 – 0
```

TRACE ID = X'2D'
DESC = ISWITCH TO=RET called

```
      word 1 – A(Current ECB)
      word 2 – ISWITCH return address
```

```
     word 3 – 0
     word 4 – SAPCNTRL field from ECB's SAP
     word 5 – SAPXFLAG contents
     word 6 – 0
```

TRACE ID = X'2E'
DESC = Shutdown ISWITCH reinstate

```
     word 1 – A(PST)
     word 2 – A(Return save area)
     word 3 – A(Shutdown ECB)
     word 4 – 0
     word 5 – 0
     word 6 – 0
```

TRACE ID = X'2F'
DESC = Dependent region open dispatcher–TCB switch

```
     word 1 – A(Related PST)
     word 2 – A(Previous TCB)
     word 3 – A(Current TCB)
     word 4 – 0
     word 5 – 0
     word 6 – 0
```

TRACE ID = X'30'
DESC = IWAIT called with IXCTL=YES option

```
     word 1 – A(Current ECB)
     word 2 – ECB Contents prior to IWAIT
     word 3 – IWAIT Return address
     word 4 – A(Target ECB)
     word 5 – Target ECB Contents
     word 6 – 0
```

```
**DTR          DISPATCHER TRACE
****************************************************
***TRACE PRINTED FROM OLDEST TO MOST CURRENT ENTRY**
****************************************************
  FUNCTION        WORD 0    WORD 1    WORD 2    WORD 3    WORD 4    WORD 5    WORD 6    WORD 7
XM ISWITCH STK  10035E11  05B5A060  80BBE2E8  80000002  00800001  001B001B  00000000  9AB7A070  MPP   TO=XMDLI
MEM CHANGE      11035E12  05B5A060  001B001B  0084001B  00B16A40  00000000  00000000  9AB7A1B3  MPP
IPOST(ECB=)     06035E17  05B5A060  80B8F516  00B21140  80B48CD7  40C1E6C5  00000000  9AB7A23D  MPP    AWE
IPC ENQ         19015E18  00B21140  40C1E6C5  FF4B7340  00B48CC0  80BE4208  00000000  9AB7A2CB  LOG    AWE
IPC RESUME      1A015E19  006DEE88  001B0084  00000082  00B48CC0  00000000  00000000  9AB7A3FC  LOG
ISERWAIT        23035E1A  85B5A060  00000000  80B8F602  00000000  00000000  00000000  9AB7A5AC  MPP
IECB STORE      1B035E1B  05B5A060  80B16A57  00000000  00000000  00000000  00000000  9AB7A671  MPP
SUSPEND         27035E1C  05B5A060  001B0084  00B16A40  00000000  00000000  00000000  9AB7A6CE  MPP
XM ISWITCH STK  10035E1E  05B4B060  867851F0  80000001  00000001  00320032  00000000  9AB7A7F1  MPP   TO=XMCTL
MEM CHANGE      11035E1F  05B4B060  00320032  00820032  00B22E00  00000000  00000000  9AB7A92D  MPP
IPOST(ECB=)     06FE5E25  006D77F0  80B91FA6  00BA156C  80B48417  40E3D9C1  00000000  9AB7A93D  N/A    TRA
IPC ENQ         19025E26  00BA156C  40E3D9C1  FF4B7C00  00B48400  80BE4208  00000000  9AB7A9A1  CTL    TRA
IPC RESUME      1A025E27  006D77F0  00820082  00000082  00B48400  00000000  00000000  9AB7A9F2  CTL
RE-DISPATCH     05015E28  00B21140  40C1E6C5  40000000  00000000  801504A6  00000000  9AB7ABA1  LOG
IWAIT           04015E2C  00B21140  00C1E6C5  801504A6  00000000  00000000  00000000  9AB7AC31  LOG    AWE
ISWITCH UNSTK   29035E2E  05B4B060  86785246  80000000  00000041  00000000  00000000  9AB7AD61  MPP
IECB STORE      1B015E2F  00B21140  80B48CD7  00C1E6C5  00000000  00000000  00000000  9AB7AF15  LOG
SUSPEND         27015E30  00000000  00820082  00B48CC0  00000000  00000000  00000000  9AB7AF7C  LOG
RE-DISPATCH     05035E31  05B4B060  00025EE4  00000003  00000000  00B22E00  00000000  9AB7AF8F  MPP
MEM CHANGE      11035E32  05B4B060  00820032  00320032  00B22E00  00000000  00000000  9AB7B04E  MPP
ITASK START     02025E33  00BA156C  40E3D9C1  064BC040  00000000  066C6440  00B7E7E0  9AB7B171  CTL    TRA
IPOST(ECB=)     06FE5E34  00000000  8007EAB8  05B37060  80AF3917  801A1D2C  00000000  9AB7B1C7  N/A    VSM
IPC ENQ         19035E35  05B37060  7FE5E2D4  FF50C700  00AF3900  80BE4208  00000000  9AB7B374  MPP    VSM
IPC RESUME      1A035E36  00000000  00840084  00000052  00AF3900  00000000  00000000  9AB7B4EF  MPP
IPOST(SAP=)     1FFE5E37  006CFE88  80B7E94C  00167060  00000000  00000000  00000000  9AB7B569  N/A
IPC ENQ         19155E39  00167060  40E3D9C1  FF4B7840  00B487C0  80BE4394  00000000  9AB7B5BC  TRA    TRA
IPC RESUME      1A155E3A  006CFE88  00820082  00000082  00B487C0  00000000  00000000  9AB7B692  TRA
ISERWAIT        23025E3D  00BA156C  00E3D9C1  80B7E956  00000000  00000000  00000000  9AB7B843  CTL    TRA
IECB STORE      1B025E3E  00BA156C  80B48417  00E3D9C1  00000000  00000000  00000000  9AB7B88D  CTL
SUSPEND         27025E40  00000000  00820082  00B48400  00000000  00000000  00000000  9AB7B8D7  CTL
XM ISWITCH STK  10035E44  05B4B060  80BBE2E8  80000002  00000001  00320032  00000000  9AB7B90E  MPP   TO=XMDLI
RE-DISPATCH     05155E45  00167060  40E3D9C1  40000000  00000000  8015EC84  00000000  9AB7B9FB  TRA
MEM CHANGE      11035E46  05B4B060  00320032  00840032  00B22E00  00000000  00000000  9AB7BA3B  MPP
RE-DISPATCH     05035E48  05B37060  7FE5E2D4  00000041  00000000  8007E9FA  00000000  9AB7BA87  MPP
KPOST LIST      2A155E4A  00167060  8015EC36  00167064  00000000  00000000  00000000  9AB7BACC  TRA
IPC ENQ         19025E4B  00BA156C  40E3D9C1  FF4B7C00  00B48400  80BE456E  00000000  9AB7BC79  CTL    TRA
IPC RESUME      1A025E4D  006CEE88  00820082  00000082  00B48400  00000000  00000000  9AB7BE28  CTL
IPOST(ECB=)     06035E4F  05B4B060  80B90B8E  00B21140  80B48CD7  40C1E6C5  00000000  9AB7BE86  MPP    AWE
IPC ENQ         19015E50  00B21140  40C1E6C5  FF4B7340  00B48CC0  80BE4208  00000000  9AB7BF72  LOG    AWE
IPC RESUME      1A015E51  006DEE88  00320084  00000082  00B48CC0  00000000  00000000  9AB7C0CB  LOG
IWAIT           04155E52  00167060  00E3D9C1  8015EC84  00000000  00000000  00000000  9AB7C1E7  TRA    TRA
IECB STORE      1B155E54  00167060  80B487D7  00E3D9C1  00000000  00000000  00000000  9AB7C324  TRA
SUSPEND         27155E55  00000000  00820082  00B487C0  00000000  00000000  00000000  9AB7C4B1  TRA
ISERWAIT        23035E56  85B4B060  00000000  80B8F602  00000000  00000000  00000000  9AB7C661  MPP
IECB STORE      1B035E57  05B4B060  80B22E17  00000000  00000000  00000000  00000000  9AB7C7AE  MPP
SUSPEND         27035E58  05B4B060  00320084  00B22E00  00000000  00000000  00000000  9AB7C917  MPP
RE-DISPATCH     05015E5B  00B21140  40C1E6C5  40000000  00000000  801504A6  00000000  9AB7CA0E  LOG
IWAIT           04015E5D  00B21140  00C1E6C5  801504A6  00000000  00000000  00000000  9AB7CBB5  LOG    AWE
```

Figure 51. Example of a Dispatcher Trace

ITASK ECB Posting

The post exit routine and the IMS posting routine add all ECBs to the posted queue.

When an IMS TCB waits for work, IMS issues an MVS SUSPEND. This task is reactivated by a RESUME invoked by the post exit posting routine or the IMS posting routine.

System Post Codes

Table 27 lists only a subset of the possible post codes.

Table 27. System Post Codes

Code	Description
X'40', C'BTR'	PST posted by scheduler as a result of BMP termination (Subqueues 4, 5)
X'40', C'CHK'	PST posted by checkpoint (Subqueues 3, 4, 5, 6)
X'40', C'SMB'	PST posted by SMB enqueue when a message is received that can be processed by the PST (Subqueue 3 or 6)
X'40', C'CMD'	PST posted by command processor when `/START PGM`, `/START TRAN`, or a similar command is entered (Subqueues 3, 6)
X'40', C'ABD'	PST posted by DFSCPY00 as a result of an abend in a dependent region (Subqueues 3, 4, 5, 6)
X'40', C'PRG'	PST posted by scheduler to stop region when checkpoint purge (that is, all messages processed) is complete—this is used if MPP issued last message (Subqueue 3)
X'40', C'STP'	PST posted by DFSSTOP0 when the region is waiting in scheduler and is to be stopped (Subqueues 3, 4, 5)
X'40', C'DLG'	PST posted by DFSRDLG0 when dynamic log is free (Subqueues 3, 4, 5, 6)
X'40', C'CF4'	PST posted by DFSASK00 as a result of an abend in a dependent region (Subqueues 3, 4, 5, 6)
X'40', C'DEQ'	Terminate control processor ECB posted by DFSRST00 at restart completion
X'40', C' TO'	PST posted after ISWITCH to IMS control region TCB
X'40', C'RET'	PST posted after ISWITCH return to dependent region TCB

External Subsystem Trace

The External Subsystem (ESS) Trace entries help you analyze problems for either:

- A connection problem between the IMS control region and the external subsystem (for example, DB2)
- Any problem between the IMS dependent region and the external subsystem

You enable the external subsystem trace by using the `/TRACE SET ON TABLE SUBS` command. When you specify `OPTION LOG`, IMS writes the trace externally as type X'67FA' records.

Figure 52 illustrates the external subsystem (ESS) trace record format. Each of the sixteen words is 4 bytes long. Words 0 and 1 hold the standard ESS trace record prefix. The MODule ID and SUB FUNCtion (WORD 1) determines what information appears in words 2 through 15.

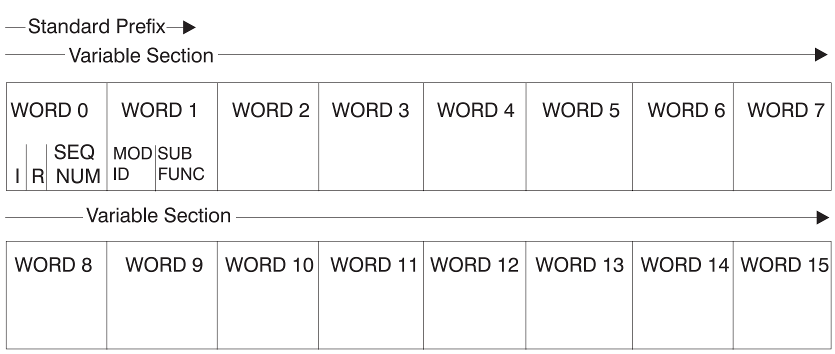


Figure 52. External Subsystem (ESS) Trace Record Format

where represents

I
This 1-byte field contains the hexadecimal trace record ID. Two possible ID values are X'57' and X'58'. The X'57' record ID is created by a module that executes in the IMS control region (for example, the ESS mother task DFSIESI0). The X'58' record ID is created by a module that executes in an IMS dependent region (for example, DFSESCT0).

R
This 1-byte field is reserved.

SEQ NUM
This 2-byte field contains the hexadecimal trace record sequence number assigned by the IMS trace component.

MOD ID
This 2-byte field contains a hexadecimal value that identifies the module that created the trace record. Each ESS module has an associated module ID. Macro DFSESFC contains the complete list of IDs.

SUB FUNC
This 2-byte field contains a hexadecimal value that identifies the subfunction that created the trace record within the module. For example, if a module creates a trace record in each of five internal subroutines, each subroutine has a unique SUB FUNC ID.

Table 28 lists:
- The ID of the module that created the trace record
- The ID of the subfunction (within the module) that created the record
- The name of the module that created the record
- A description of the event being traced

Table 28. Module ID and Subfunction Table

Module ID	Sub Function	Module	Meaning
X'0015'	X'0015'	DFSESS40	ESS message service exit
X'0016'	X'0014'	DFSESS30	ESS logging exit
X'0017'	X'0011' X'0012' X'0040' X'0041'	DFSESS10	IMS control region identify Dependent region identify Control region identify error Identify error subsystem stopped
X'0018'	X'0013'	DFSESS20	ESS termination exit (if X'57') Dependent region ESS term (if X'58')
X'0285'	X'0010'	DFSESD80	Dependent region ESS initialization
X'0288'	X'0001'	DFSESSO0	Dependent region ESS sign on
X'0289'	X'0003'	DFSESD50	Dependent region ESS signoff
X'0290'	X'0005'	DFSESCT0	Dependent region ESS create thread
X'0291'	X'0002' X'0003' X'0004'	DFSESD50	Dependent region ESS term thread Dependent region ESS term thread region ESS signoff Dependent region ESS term identify
X'0292'	X'0004'	DFSESD50	Dependent region ESS term identify
X'0293'	X'0007'	DFSESAB0	Dependent region ESS ABORT
X'0294'	X'0008'	DFSESP10	Dependent region ESS commit prep
X'0295'	X'0009'	DFSESP20	Dependent region ESS commit cont
X'0307'	X'0016' X'0017' X'0018'	DFSFESP0	ESS commit processor entered ESS commit processor exited ESS commit processor R-I-D request

Table 28. Module ID and Subfunction Table (continued)

Module ID	Sub Function	Module	Meaning
X'0402'	X'0020'	DFSESI30	IMS control region daughter identify
	X'0021'		IMS control region resolve-in-doubt
	X'0022'		IMS control region ESS CMD
	X'0023'		IMS control region ESS RRE
	X'0024'		IMS control region ESS ECHO
	X'0025'		IMS control region terminate identify
	X'0026'		IMS control region terminate subsystem
	X'0027'		IMS control region /STOP CMD
	X'0028'		IMS control region ESS term record
	X'0029'		IMS control region ESS shutdown
	X'0030'		IMS control region ESS termination
	X'0031'		IMS control region ESS AWE error
X'0403'	X'0019'	DFSESI50	Control region ESS initialization
X'0404'	X'0042'	DFSESI60	Control region ESS R-I-D exit
X'0405'	X'0032'	DFSESI70	Control region ESS /CHANGE
X'0409'	X'0001'	DFSIESI0	Mother ITASK request
	X'0002'		Control region ESS attach
X'0506'	X'0006'	DFSESPR0	Dependent region ESS program request handler
	X'0019'		Dependent region ESS program request recursive call
	X'0020'		Dependent region ESS Subsystem Not Operational (SNOX)

Layout of the X'57' Variable Section

```
MOD ID   = X'0015'
SUB FUNC = X'0015' DFSESS40 External SubSys MESSAGE service request
                   record

     word   2 -- External SubSystem name
     words  3 through 15   not used
```

```
MOD ID   = X'0016'
SUB FUNC = X'0014' DFSESS30 External SubSys LOGGING service request
                   record

     word   2 -- External SubSystem name
     words  3 through 15   not used
```

```
MOD ID   = X'0017'
SUB FUNC = X'0011' DFSESS10 control region External SubSys IDENTIFY record


      word   2 -- External SubSystem name
      word   3 -- bytes 0-1 not used
                  byte  2   GESEGF1  (DFSGESE macro global flag1)
                  byte  3   GESEGF2  (DFSGESE macro global flag2)
      word   4 -- byte  0   GESEGF3  (DFSGESE macro global flag3)
                  byte  1   not used
                  byte  2   SSIDFLG1 (DFSSSIE subsys status flag1)
                  byte  3   SSIDFLG2 (DFSSSIE subsys status flag2)
      word   5 -- bytes 0-1 not used
                  bytes 2-3 AWQRC    (DFSAWE DFSESI30 identify return code)

      words  6 through 15   not used
```

```
SUB FUNC = X'0040' DFSESS10 External SubSys GLOBAL identify error record

     word   2 -- External SubSystem name
     word   3 -- bytes 0-1 not used
                 byte  2   GESEGF1  (DFSGESE macro global flag1)
                 byte  3   GESEGF2  (DFSGESE macro global flag2)
     word   4 -- byte  0   GESEGF3  (DFSGESE macro global flag3)
                 byte  1   not used
                 byte  2   SSIDFLG1 (DFSSSIE subsys status flag1)
                 byte  3   SSIDFLG2 (DFSSSIE subsys status flag2)
     words  5 through 15   not used
SUB FUNC = X'0041' DFSESS10 External SubSys identify with External SubSystem

                   stopped or stopping record

     word   2 -- External SubSystem name
     word   3 -- bytes 0-1 not used
                 byte  2   GESEGF1  (DFSGESE macro global flag1)
                 byte  3   GESEGF2  (DFSGESE macro global flag2)
     word   4 -- byte  0   GESEGF3  (DFSGESE macro global flag3)
                 byte  1   not used
                 byte  2   SSIDFLG1 (DFSSSIE subsys status flag1)
                 byte  3   SSIDFLG2 (DFSSSIE subsys status flag2)
     words  5 through 15   not used
MOD ID   = X'0018'
SUB FUNC = X'0013' DFSESS20 External SubSys termination record

     word   2 -- External SubSystem name
     word   3 -- bytes 0-1 not used
                 byte  2   GESEGF1  (DFSGESE macro global flag1)
                 byte  3   GESEGF2  (DFSGESE macro global flag2)
     word   4 -- byte  0   GESEGF3  (DFSGESE macro global flag3)
                 byte  1   not used
                 byte  2   SSIDFLG1 (DFSSSIE subsys status flag1)
                 byte  3   SSIDFLG2 (DFSSSIE subsys status flag2)
     words  5 through 15   not used
MOD ID   = X'0402'
SUB FUNC = X'0020' DFSESI30 External SubSys IDENTIFY exit record

     word   2 -- External SubSystem name
     word   3 -- bytes 0-1 not used
                 byte  2   GESEGF1  (DFSGESE macro global flag1)
                 byte  3   GESEGF2  (DFSGESE macro global flag2)
     word   4 -- byte  0   GESEGF3  (DFSGESE macro global flag3)
                 byte  1   ESSTERRC (External SubSys termination reason)
                 byte  2   SSIDFLG1 (DFSSSIE subsys status flag1)
                 byte  3   SSIDFLG2 (DFSSSIE subsys status flag2)
     word   5 -- bytes 0-1 not used
                 bytes 2-3 External SubSys exit routine return code
     words  6 through 15   not used
SUB FUNC = X'0021' DFSESI30 External SubSys RESOLVE IN DOUBT record

     word   2 -- External SubSystem name
     word   3 -- bytes 0-1 not used
                 byte  2   GESEGF1  (DFSGESE macro global flag1)
                 byte  3   GESEGF2  (DFSGESE macro global flag2)
     word   4 -- byte  0   GESEGF3  (DFSGESE macro global flag3)
                 byte  1   ESSTERRC (External SubSys termination reason)
                 byte  2   SSIDFLG1 (DFSSSIE subsys status flag1)
                 byte  3   SSIDFLG2 (DFSSSIE subsys status flag2)
     word   5 -- bytes 0-1 not used
                 bytes 2-3 AWQRC    (DFSAWE return code, see DFSESSEC)
     words  6 through  7   not used
```

```
      words  8 through 11   RRETOKEN (DFSRRE UOW recovery token)
      word  12 -- bytes 0-1 RRECI    (DFSRRE commit indicator)
                  bytes 2-3 not used
      words 13 through 15   not used
```

SUB FUNC = X'0022' DFSESI30 External SubSys /SSR COMMAND exit record

```
      word   2 -- External SubSystem name
      word   3 -- bytes 0-1 not used
                  byte  2   GESEGF1  (DFSGESE macro global flag1)
                  byte  3   GESEGF2  (DFSGESE macro global flag2)
      word   4 -- byte  0   GESEGF3  (DFSGESE macro global flag3)
                  byte  1   ESSTERRC (External SubSys termination reason)
                  byte  2   SSIDFLG1 (DFSSSIE subsys status flag1)
                  byte  3   SSIDFLG2 (DFSSSIE subsys status flag2)
      word   5 -- bytes 0-1 not used
                  bytes 2-3 External SubSys exit routine return code
      words  6 through 15   not used
```

SUB FUNC = X'0023' DFSESI30 External SubSys specific RRE request record

```
      word   2 -- External SubSystem name
      word   3 -- bytes 0-1 not used
                  byte  2   GESEGF1  (DFSGESE macro global flag1)
                  byte  3   GESEGF2  (DFSGESE macro global flag2)
      word   4 -- byte  0   GESEGF3  (DFSGESE macro global flag3)
                  byte  1   ESSTERRC (External SubSys termination reason)
                  byte  2   SSIDFLG1 (DFSSSIE subsys status flag1)
                  byte  3   SSIDFLG2 (DFSSSIE subsys status flag2)
      words  5 through  7   not used
      words  8 through 11   RRETOKEN (DFSRRE UOW recovery token)
      word  12 -- bytes 0-1 RRECI    (DFSRRE commit indicator)
                  bytes 2-3 not used
      words 13 through 15   not used
```

SUB FUNC = X'0024' DFSESI30 External SubSys ECHO exit record

```
      word   2 -- External SubSystem name
      word   3 -- bytes 0-1 not used
                  byte  2   GESEGF1  (DFSGESE macro global flag1)
                  byte  3   GESEGF2  (DFSGESE macro global flag2)
      word   4 -- byte  0   GESEGF3  (DFSGESE macro global flag3)
                  byte  1   ESSTERRC (External SubSys termination reason)
                  byte  2   SSIDFLG1 (DFSSSIE subsys status flag1)
                  byte  3   SSIDFLG2 (DFSSSIE subsys status flag2)
      word   5 -- bytes 0-1 not used
                  bytes 2-3 External SubSys exit routine return code
      words  6 through  7   not used
      words  8 through 11   RRETOKEN (DFSRRE UOW recovery token)
      word  12 -- bytes 0-1 RRECI    (DFSRRE commit indicator)
                  bytes 2-3 not used
      words 13 through 15   not used
```

SUB FUNC = X'0025' DFSESI30 External SubSys TERMINATE IDENTIFY exit record

```
      word   2 -- External SubSystem name
      word   3 -- bytes 0-1 not used
                  byte  2   GESEGF1  (DFSGESE macro global flag1)
                  byte  3   GESEGF2  (DFSGESE macro global flag2)
      word   4 -- byte  0   GESEGF3  (DFSGESE macro global flag3)
                  byte  1   ESSTERRC (External SubSys termination reason)
                  byte  2   SSIDFLG1 (DFSSSIE subsys status flag1)
                  byte  3   SSIDFLG2 (DFSSSIE subsys status flag2)
      word   5 -- bytes 0-1 not used
                  bytes 2-3 External SubSys exit routine return code
      words  6 through 15   not used
```

SUB FUNC = X'0026' DFSESI30 External SubSys TERMINATE SUBSYSTEM record

```
word   2 -- External SubSystem name
word   3 -- bytes 0-1 not used
            byte  2   GESEGF1  (DFSGESE macro global flag1)
            byte  3   GESEGF2  (DFSGESE macro global flag2)
word   4 -- byte  0   GESEGF3  (DFSGESE macro global flag3)
            byte  1   ESSTERRC (External SubSys termination reason)
            byte  2   SSIDFLG1 (DFSSSIE subsys status flag1)
            byte  3   SSIDFLG2 (DFSSSIE subsys status flag2)
words  5 through 15   not used
```

SUB FUNC = X'0027' DFSESI30 External SubSys /STOP command record

```
word   2 -- External SubSystem name
word   3 -- bytes 0-1 not used
            byte  2   GESEGF1  (DFSGESE macro global flag1)
            byte  3   GESEGF2  (DFSGESE macro global flag2)
word   4 -- byte  0   GESEGF3  (DFSGESE macro global flag3)
            byte  1   ESSTERRC (External SubSys termination reason)
            byte  2   SSIDFLG1 (DFSSSIE subsys status flag1)
            byte  3   SSIDFLG2 (DFSSSIE subsys status flag2)
words  5 through 15   not used
```

SUB FUNC = X'0028' DFSESI30 External SubSys IMS termination record

```
word   2 -- External SubSystem name
word   3 -- bytes 0-1 not used
            byte  2   GESEGF1  (DFSGESE macro global flag1)
            byte  3   GESEGF2  (DFSGESE macro global flag2)
word   4 -- byte  0   GESEGF3  (DFSGESE macro global flag3)
            byte  1   ESSTERRC (External SubSys termination reason)
            byte  2   SSIDFLG1 (DFSSSIE subsys status flag1)
            byte  3   SSIDFLG2 (DFSSSIE subsys status flag2)
words  5 through 15   not used
```

SUB FUNC = X'0029' DFSESI30 External SubSys IMS shutdown record

```
word   2 -- External SubSystem name
word   3 -- bytes 0-1 not used
            byte  2   GESEGF1  (DFSGESE macro global flag1)
            byte  3   GESEGF2  (DFSGESE macro global flag2)
word   4 -- byte  0   GESEGF3  (DFSGESE macro global flag3)
            byte  1   ESSTERRC (External SubSys termination reason)
            byte  2   SSIDFLG1 (DFSSSIE subsys status flag1)
            byte  3   SSIDFLG2 (DFSSSIE subsys status flag2)
words  5 through 15   not used
```

SUB FUNC = X'0030' DFSESI30 External SubSys TERMINATION exit record

```
word   2 -- External SubSystem name
word   3 -- bytes 0-1 not used
            byte  2   GESEGF1  (DFSGESE macro global flag1)
            byte  3   GESEGF2  (DFSGESE macro global flag2)
word   4 -- byte  0   GESEGF3  (DFSGESE macro global flag3)
            byte  1   ESSTERRC (External SubSys termination reason)
            byte  2   SSIDFLG1 (DFSSSIE subsys status flag1)
            byte  3   SSIDFLG2 (DFSSSIE subsys status flag2)
word   5 -- bytes 0-1 not used
            bytes 2-3 External SubSys exit routine return code
words  6 through 15   not used
```

SUB FUNC = X'0031' DFSESI30 AWE error record

```
word   2 -- External SubSystem name
word   3 -- bytes 0-1 not used
            byte  2   GESEGF1  (DFSGESE macro global flag1)
            byte  3   GESEGF2  (DFSGESE macro global flag2)
word   4 -- byte  0   GESEGF3  (DFSGESE macro global flag3)
            byte  1   ESSTERRC (External SubSys termination reason)
```

```
                byte  2   SSIDFLG1 (DFSSSIE subsys status flag1)
                byte  3   SSIDFLG2 (DFSSSIE subsys status flag2)
    word   5 -- bytes 0-1 not used
                bytes 2-3 AWQRC    (DFSAWE return code)
    words  6 through 15   not used
```

MOD ID   = X'0403'
SUB FUNC = X'0019' DFSESI50 External SubSys INITIALIZATION exit record

```
    word   2 -- External SubSystem name
    word   3 -- bytes 0-1 not used
                byte  2   GESEGF1  (DFSGESE macro global flag1)
                byte  3   GESEGF2  (DFSGESE macro global flag2)
    word   4 -- byte  0   GESEGF3  (DFSGESE macro global flag3)
                byte  1   not used
                byte  2   SSIDFLG1 (DFSSSIE subsys status flag1)
                byte  3   SSIDFLG2 (DFSSSIE subsys status flag2)
    word   5 -- bytes 0-1 not used
                bytes 2-3 External SubSys exit routine return code
    words  6 through 15   not used
```

MOD ID   = X'0404'
SUB FUNC = X'0042' DFSESI60 External SubSys RESOLVE IN DOUBT exit record

```
    word   2 -- External SubSystem name
    word   3 -- bytes 0-1 not used
                byte  2   GESEGF1  (DFSGESE macro global flag1)
                byte  3   GESEGF2  (DFSGESE macro global flag2)
    word   4 -- byte  0   GESEGF3  (DFSGESE macro global flag3)
                byte  1   not used
                byte  2   SSIDFLG1 (DFSSSIE subsys status flag1)
                byte  3   SSIDFLG2 (DFSSSIE subsys status flag2)
    word   5 -- bytes 0-1 not used
                bytes 2-3 External SubSys exit routine return code
    words  6 through  7   not used
    words  8 through 11   RRETOKEN (DFSRRE UOW recovery token)
    word  12 -- bytes 0-1 RRECI    (DFSRRE commit indicator)
                bytes 2-3 not used
    words 13 through 15   not used
```

MOD ID   = X'0405'
SUB FUNC = X'0032' DFSESI70 External SubSys /CHANGE command record

```
    word   2 -- External SubSystem name
    word   3 -- bytes 0-1 not used
                byte  2   GESEGF1  (DFSGESE macro global flag1)
                byte  3   GESEGF2  (DFSGESE macro global flag2)
    word   4 -- byte  0   GESEGF3  (DFSGESE macro global flag3)
                byte  1   not used
                byte  2   SSIDFLG1 (DFSSSIE subsys status flag1)
                byte  3   SSIDFLG2 (DFSSSIE subsys status flag2)
    words  5 through 15   not used
```

MOD ID   = X'0409'
SUB FUNC = X'0001' DFSIESI0 mother ITASK request record

```
    word   2 -- not used
    word   3 -- bytes 0-1 function requested
                  Function requested:
                  X'0002' terminate the mother ITASK TCB
                  X'0003' build / merge subsystem definitions
                  X'0004' SSM JCL parameter
                  X'0005' attach external subsystem ITASK TCB
                  X'0007' /START command
                  X'0008' sync request
                bytes 2-3 not used
    word   4 -- not used
    word   5 -- bytes 0-1 not used
                bytes 2-3 AWQRC    (DFSAWE DFSIESI0 return code)
    words  6 through 15   not used
```

```
SUB FUNC = X'0002' DFSIESI0 External Subsys ATTACH record

     word   2 -- External SubSystem name
     word   3 -- bytes 0-1 function requested
                  Function requested:
                  X'0005' attach external subsystem ITASK TCB
                  X'0007' /START command
                byte  2   GESEGF1  (DFSGESE macro global flag1)
                byte  3   GESEGF2  (DFSGESE macro global flag2)
     word   4 -- byte  0   GESEGF3  (DFSGESE macro global flag3)
                byte  1   not used
                byte  2   SSIDFLG1 (DFSSSIE subsys status flag1)
                byte  3   SSIDFLG2 (DFSSSIE subsys status flag2)
     word   5 -- bytes 0-1 not used
                bytes 2-3 AWQRC    (DFSAWE attach process return code)
     words  6 through 15   not used
```

Layout of the X'58' Variable Section

```
MOD ID   = X'0015'
SUB FUNC = X'0015' DFSESS40 External SubSys MESSAGE service request
                  record

     word   2 -- External SubSystem name
     words  3 through 15   not used
```

```
MOD ID   = X'0016'
SUB FUNC = X'0014' DFSESS30 External SubSys LOGGING service request
                  record

     word   2 -- External SubSystem name
     words  3 through 15   not used
```

```
MOD ID   = X'0017'
SUB FUNC = X'0011' DFSESS10 control region External SubSys IDENTIFY record

     word   2 -- External SubSystem name
     word   3 -- bytes 0-1 PSTID    (IMS dependent region ID)
                byte  2   GESEGF1  (DFSGESE macro global flag1)
                byte  3   GESEGF2  (DFSGESE macro global flag2)
     word   4 -- byte  0   GESEGF3  (DFSGESE macro global flag3)
                byte  1   not used
                byte  2   SSIDFLG1 (DFSSSIE subsys status flag1)
                byte  3   SSIDFLG2 (DFSSSIE subsys status flag2)
     word   5 -- bytes 0-1 not used
                bytes 2-3 AWQRC    (DFSAWE DFSESI30 identify return code)
     words  6 through  7   not used
     words  8 through 11   LCRETOKN (DFSLCRE UOW recovery token)
     words 12 through 15   not used
```

```
SUB FUNC = X'0012' DFSESS10 dependent region External SubSys IDENTIFY
                  record

     word   2 -- External SubSystem name
     word   3 -- bytes 0-1 PSTID    (IMS dependent region ID)
                byte  2   EZSGFL   (DFSEZS connection status byte1)
                byte  3   EZSLFL   (DFSEZS connection status byte2)
     word   4 -- byte  0   EZSEFL1  (DFSEZS thread startup status)
                byte  1   EZSEFL2  (DFSEZS thread commit status)
                byte  2   EZSEFL3  (DFSEZS thread termination status)
                byte  3   EZSEFL4  (DFSEZS termination flag)
     word   5 -- bytes 0-1 not used
                bytes 2-3 AWQRC    (DFSAWE DFSESI30 identify return code)
     words  6 through  7   not used
     words  8 through 11   LCRETOKN (DFSLCRE UOW recovery token)
     words 12 through 15   not used
```

```
SUB FUNC = X'0040' DFSESS10 IMS detected External SubSys IDENTIFY error
                   record

     word   2 -- External SubSystem name
     word   3 -- bytes 0-1 PSTID    (IMS dependent region ID)
                 byte  2   GESEGF1  (DFSGESE macro global flag1)
                 byte  3   GESEGF2  (DFSGESE macro global flag2)
     word   4 -- byte  0   GESEGF3  (DFSGESE macro global flag3)
                 byte  1   not used
                 byte  2   SSIDFLG1 (SSIDX subsys status flag1)
                 byte  3   SSIDFLG2 (SSIDX subsys status flag2)
     words  5 through  7   not used
     words  8 through 11   LCRETOKN (DFSLCRE UOW recovery token)
     words 12 through 15   not used
SUB FUNC = X'0041' DFSESS10 IMS detected External SubSys IDENTIFY with
                   External SubSystem stopped or stopping record

     word   2 -- External SubSystem name
     word   3 -- bytes 0-1 PSTID    (IMS dependent region ID)
                 byte  2   GESEGF1  (DFSGESE macro global flag1)
                 byte  3   GESEGF2  (DFSGESE macro global flag2)
     word   4 -- byte  0   GESEGF3  (DFSGESE macro global flag3)
                 byte  1   not used
                 byte  2   SSIDFLG1 (SSIDX subsys status flag1)
                 byte  3   SSIDFLG2 (SSIDX subsys status flag2)
     words  5 through  7   not used
     words  8 through 11   LCRETOKN (DFSLCRE UOW recovery token)
     words 12 through 15   not used
MOD ID   = X'0018'
SUB FUNC = X'0013' DFSESS20 External SubSys termination record

     word   2 -- External SubSystem name
     word   3 -- bytes 0-1 not used
                 byte  2   GESEGF1  (DFSGESE macro global flag1)
                 byte  3   GESEGF2  (DFSGESE macro global flag2)
     word   4 -- byte  0   GESEGF3  (DFSGESE macro global flag3)
                 byte  1   not used
                 byte  2   SSIDFLG1 (DFSSSIE subsys status flag1)
                 byte  3   SSIDFLG2 (DFSSSIE subsys status flag2)
     words  5 through 15   not used
MOD ID   = X'0285'
SUB FUNC = X'0010' DFSESD80 dep region External SubSys INITIALIZATION exit
                   record

     word   2 -- External SubSystem name
     word   3 -- bytes 0-1 PSTID    (IMS dependent region ID)
                 byte  2   EZSGFL   (DFSEZS connection status byte1)
                 byte  3   EZSLFL   (DFSEZS connection status byte2)
     word   4 -- byte  0   EZSEFL1  (DFSEZS thread startup status)
                 byte  1   EZSEFL2  (DFSEZS thread commit status)
                 byte  2   EZSEFL3  (DFSEZS thread termination status)
                 byte  3   EZSEFL4  (DFSEZS termination flag)
     word   5 -- bytes 0-1 not used
                 bytes 2-3 External SubSys exit routine return code
     words  6 through  7   not used
     words  8 through 11   LCRETOKN (DFSLCRE UOW recovery token)
     words 12 through 15   not used
MOD ID   = X'0288'
SUB FUNC = X'0001' DFSESS00 External SubSys SIGNON exit record

     word   2 -- External SubSystem name
     word   3 -- bytes 0-1 PSTID    (IMS dependent region ID)
                 byte  2   EZSGFL   (DFSEZS connection status byte1)
                 byte  3   EZSLFL   (DFSEZS connection status byte2)
     word   4 -- byte  0   EZSEFL1  (DFSEZS thread startup status)
                 byte  1   EZSEFL2  (DFSEZS thread commit status)
```

```
                 byte  2   EZSEFL3  (DFSEZS thread termination status)
                 byte  3   EZSEFL4  (DFSEZS termination flag)
     word   5 -- bytes 0-1 not used
                 bytes 2-3 External SubSys exit routine return code
     words  6 through  7   not used
     words  8 through 11   LCRETOKN (DFSLCRE UOW recovery token)
     words 12 through 15   not used
```

MOD ID   = X'0289'
SUB FUNC = X'0003' DFSESD50 External SubSys SIGNOFF exit record

```
     word   2 -- External SubSystem name
     word   3 -- bytes 0-1 PSTID    (IMS dependent region ID)
                 byte  2   EZSGFL   (DFSEZS connection status byte1)
                 byte  3   EZSLFL   (DFSEZS connection status byte2)
     word   4 -- byte  0   EZSEFL1  (DFSEZS thread startup status)
                 byte  1   EZSEFL2  (DFSEZS thread commit status)
                 byte  2   EZSEFL3  (DFSEZS thread termination status)
                 byte  3   EZSEFL4  (DFSEZS termination flag)
     word   5 -- bytes 0-1 not used
                 bytes 2-3 External SubSys exit routine return code
     words  6 through  7   not used
     words  8 through 11   LCRETOKN (DFSLCRE UOW recovery token)
     words 12 through 15   not used
```

MOD ID   = X'0290'
SUB FUNC = X'0005' DFSESCT0 External SubSys CREATE THREAD exit record

```
     word   2 -- External SubSystem name
     word   3 -- bytes 0-1 PSTID    (IMS dependent region ID)
                 byte  2   EZSGFL   (DFSEZS connection status byte1)
                 byte  3   EZSLFL   (DFSEZS connection status byte2)
     word   4 -- byte  0   EZSEFL1  (DFSEZS thread startup status)
                 byte  1   EZSEFL2  (DFSEZS thread commit status)
                 byte  2   EZSEFL3  (DFSEZS thread termination status)
                 byte  3   EZSEFL4  (DFSEZS termination flag)
     word   5 -- bytes 0-1 not used
                 bytes 2-3 External SubSys exit routine return code
     words  6 through  7   not used
     words  8 through 11   LCRETOKN (DFSLCRE UOW recovery token)
     words 12 through 15   not used
```

MOD ID   = X'0291'
SUB FUNC = X'0002' DFSESD50 External SubSys TERMINATE THREAD exit record

```
     word   2 -- External SubSystem name
     word   3 -- bytes 0-1 PSTID    (IMS dependent region ID)
                 byte  2   EZSGFL   (DFSEZS connection status byte1)
                 byte  3   EZSLFL   (DFSEZS connection status byte2)
     word   4 -- byte  0   EZSEFL1  (DFSEZS thread startup status)
                 byte  1   EZSEFL2  (DFSEZS thread commit status)
                 byte  2   EZSEFL3  (DFSEZS thread termination status)
                 byte  3   EZSEFL4  (DFSEZS termination flag)
     word   5 -- bytes 0-1 not used
                 bytes 2-3 External SubSys exit routine return code
     words  6 through  7   not used
     words  8 through 11   LCRETOKN (DFSLCRE UOW recovery token)
     words 12 through 15   not used
```

MOD ID   = X'0292'
SUB FUNC = X'0004' DFSESD50 External SubSys TERMINATE IDENTIFY exit
record

```
     word   2 -- External SubSystem name
     word   3 -- bytes 0-1 PSTID    (IMS dependent region ID)
                 byte  2   EZSGFL   (DFSEZS connection status byte1)
                 byte  3   EZSLFL   (DFSEZS connection status byte2)
     word   4 -- byte  0   EZSEFL1  (DFSEZS thread startup status)
                 byte  1   EZSEFL2  (DFSEZS thread commit status)
                 byte  2   EZSEFL3  (DFSEZS thread termination status)
```

```
                  byte  3   EZSEFL4  (DFSEZS termination flag)
      word   5 -- bytes 0-1 not used
                  bytes 2-3 External SubSys exit routine return code
      words  6 through  7   not used
      words  8 through 11   LCRETOKN (DFSLCRE UOW recovery token)
      words 12 through 15   not used
```

MOD ID   = X'0293'
SUB FUNC = X'0007' DFSESAB0 External SubSys ABORT exit record

```
      word   2 -- External SubSystem name
      word   3 -- bytes 0-1 PSTID    (IMS dependent region ID)
                  byte  2   EZSGFL   (DFSEZS connection status byte1)
                  byte  3   EZSLFL   (DFSEZS connection status byte2)
      word   4 -- byte  0   EZSEFL1  (DFSEZS thread startup status)
                  byte  1   EZSEFL2  (DFSEZS thread commit status)
                  byte  2   EZSEFL3  (DFSEZS thread termination status)
                  byte  3   EZSEFL4  (DFSEZS termination flag)
      word   5 -- bytes 0-1 not used
                  bytes 2-3 External SubSys exit routine return code
      words  6 through  7   not used
      words  8 through 11   LCRETOKN (DFSLCRE UOW recovery token)
      words 12 through 15   not used
```

MOD ID   = X'0294'
SUB FUNC = X'0008' DFSESP10 External SubSys COMMIT PREPARE exit record

```
      word   2 -- External SubSystem name
      word   3 -- bytes 0-1 PSTID    (IMS dependent region ID)
                  byte  2   EZSGFL   (DFSEZS connection status byte1)
                  byte  3   EZSLFL   (DFSEZS connection status byte2)
      word   4 -- byte  0   EZSEFL1  (DFSEZS thread startup status)
                  byte  1   EZSEFL2  (DFSEZS thread commit status)
                  byte  2   EZSEFL3  (DFSEZS thread termination status)
                  byte  3   EZSEFL4  (DFSEZS termination flag)
      word   5 -- bytes 0-1 not used
                  bytes 2-3 External SubSys exit routine return code
      words  6 through  7   not used
      words  8 through 11   LCRETOKN (DFSLCRE UOW recovery token)
      words 12 through 15   not used
```

MOD ID   = X'0295'
SUB FUNC = X'0009' DFSESP20 External SubSys COMMIT CONTINUE exit record

```
      word   2 -- External SubSystem name
      word   3 -- bytes 0-1 PSTID    (IMS dependent region ID)
                  byte  2   EZSGFL   (DFSEZS connection status byte1)
                  byte  3   EZSLFL   (DFSEZS connection status byte2)
      word   4 -- byte  0   EZSEFL1  (DFSEZS thread startup status)
                  byte  1   EZSEFL2  (DFSEZS thread commit status)
                  byte  2   EZSEFL3  (DFSEZS thread termination status)
                  byte  3   EZSEFL4  (DFSEZS termination flag)
      word   5 -- bytes 0-1 not used
                  bytes 2-3 External SubSys exit routine return code
      words  6 through  7   not used
      words  8 through 11   LCRETOKN (DFSLCRE UOW recovery token)
      words 12 through 15   not used
```

MOD ID   = X'0297'
SUB FUNC = X'000A' DFSESP30 External SubSys COMMIT VERIFY exit record

```
      word   2 -- External SubSystem name
      word   3 -- bytes 0-1 PSTID    (IMS dependent region ID)
                  byte  2   EZSGFL   (DFSEZS connection status byte1)
                  byte  3   EZSLFL   (DFSEZS connection status byte2)
      word   4 -- byte  0   EZSEFL1  (DFSEZS thread startup status)
                  byte  1   EZSEFL2  (DFSEZS thread commit status)
                  byte  2   EZSEFL3  (DFSEZS thread termination status)
                  byte  3   EZSEFL4  (DFSEZS termination flag)
      word   5 -- bytes 0-1 not used
```

```
                  bytes 2-3 External SubSys exit routine return code
     words  6 through  7   not used
     words  8 through 11   LCRETOKN (DFSLCRE UOW recovery token)
     words 12 through 15   not used
```

MOD ID   = X'0307'
SUB FUNC = X'0016' DFSFESP0 External SubSys commit processor entry record

```
     word   2 -- External SubSystem name
     word   3 -- bytes 0-1 PSTID    (IMS dependent region ID)
                  byte  2   EZSGFL   (DFSEZS connection status byte1)
                  byte  3   EZSLFL   (DFSEZS connection status byte2)
     word   4 -- byte  0   EZSEFL1  (DFSEZS thread startup status)
                  byte  1   EZSEFL2  (DFSEZS thread commit status)
                  byte  2   EZSEFL3  (DFSEZS thread termination status)
                  byte  3   EZSEFL4  (DFSEZS termination flag)
     word   5 -- byte  0   PSTFUNCT (IDLI function code)
                  byte  1   PSTSYNFC (sync function code)
                  byte  2   SSTTFGT1 (DFSSSOB termination flag)
                  byte  3   not used
     word   6 -- bytes 0-1 SSTTCOMP (DFSSSOB user completion bytes 2,3)
                  byte  2   LCREF1   (DFSLCRE status indicators)
                  byte  3   LCREF2   (DFSLCRE region connection status)
     word   7 -- byte  0   LCREF3   (DFSLCRE thread status)
                  byte  1   LCREF4   (DFSLCRE internal resource manager status)
                  byte  2   LCREESST (DFSLCRE ESS resource manager status byte1)
                  byte  3   LCREESF  (DFSLCRE ESS resource manager status byte2)
     words  8 through 11   RRETOKEN (DFSRRE UOW recovery token)
     word  12 -- bytes 0-1 RRECI    (DFSRRE commit indicator)
                  bytes 2-3 not used
     words 13 through 15   not used
```

SUB FUNC = X'0017' DFSFESP0 External SubSys commit processor exit record

```
     word   2 -- External SubSystem name
     word   3 -- bytes 0-1 PSTID    (IMS dependent region ID)
                  byte  2   EZSGFL   (DFSEZS connection status byte1)
                  byte  3   EZSLFL   (DFSEZS connection status byte2)
     word   4 -- byte  0   EZSEFL1  (DFSEZS thread startup status)
                  byte  1   EZSEFL2  (DFSEZS thread commit status)
                  byte  2   EZSEFL3  (DFSEZS thread termination status)
                  byte  3   EZSEFL4  (DFSEZS termination flag)
     word   5 -- byte  0   PSTFUNCT (IDLI function code)
                  byte  1   PSTSYNFC (sync function code)
                  byte  2   SSTTFGT1 (DFSSSOB termination flag)
                  byte  3   not used
     word   6 -- bytes 0-1 SSTTCOMP (DFSSSOB user completion bytes 2,3)
                  byte  2   LCREF1   (DFSLCRE status indicators)
                  byte  3   LCREF2   (DFSLCRE region connection status)
     word   7 -- byte  0   LCREF3   (DFSLCRE thread status)
                  byte  1   LCREF4   (DFSLCRE internal resource manager status)
                  byte  2   LCREESST (DFSLCRE ESS resource manager status byte1)
                  byte  3   LCREESF  (DFSLCRE ESS resource manager status byte2)
     words  8 through 11   RRETOKEN (DFSRRE UOW recovery token)
     word  12 -- bytes 0-1 RRECI    (DFSRRE commit indicator)
                  bytes 2-3 not used
     words 13 through 15   not used
```

SUB FUNC = X'0018' DFSFESP0 External SubSys commit processor Resolve
In Doubt requested record

```
     word   2 -- External SubSystem name
     word   3 -- bytes 0-1 PSTID    (IMS dependent region ID)
                  byte  2   EZSGFL   (DFSEZS connection status byte1)
                  byte  3   EZSLFL   (DFSEZS connection status byte2)
     word   4 -- byte  0   EZSEFL1  (DFSEZS thread startup status)
                  byte  1   EZSEFL2  (DFSEZS thread commit status)
                  byte  2   EZSEFL3  (DFSEZS thread termination status)
                  byte  3   EZSEFL4  (DFSEZS termination flag)
```

```
      word   5 -- byte  0   PSTFUNCT (IDLI function code)
                   byte  1   PSTSYNFC (sync function code)
                   byte  2   SSTTFGT1 (DFSSSOB termination flag)
                   byte  3   not used
      word   6 -- bytes 0-1 SSTTCOMP (DFSSSOB user completion bytes 2,3)
                   byte  2   LCREF1   (DFSLCRE status indicators)
                   byte  3   LCREF2   (DFSLCRE region connection status)
      word   7 -- byte  0   LCREF3   (DFSLCRE thread status)
                   byte  1   LCREF4   (DFSLCRE internal resource manager status)
                   byte  2   LCREESST (DFSLCRE ESS resource manager status byte1)
                   byte  3   LCREESF  (DFSLCRE ESS resource manager status byte2)
      words  8 through 11   RRETOKEN (DFSRRE UOW recovery token)
      word  12 -- bytes 0-1 RRECI    (DFSRRE commit indicator)
                   bytes 2-3 not used
      words 13 through 15   not used
```

MOD ID   = X'0506'
SUB FUNC = X'0006' DFSESPR0 External SubSys PROGRAM REQUEST HANDLER record

```
      word   2 -- External SubSystem name
      word   3 -- bytes 0-1 PSTID    (IMS dependent region ID)
                   byte  2   EZSGFL   (DFSEZS connection status byte1)
                   byte  3   EZSLFL   (DFSEZS connection status byte2)
      word   4 -- byte  0   EZSEFL1  (DFSEZS thread startup status)
                   byte  1   EZSEFL2  (DFSEZS thread commit status)
                   byte  2   EZSEFL3  (DFSEZS thread termination status)
                   byte  3   EZSEFL4  (DFSEZS termination flag)
      word   5 -- bytes 0-1 not used
                   bytes 2-3 External SubSys exit routine return code
      words  6 through  7   not used
      words  8 through 11   LCRETOKN (DFSLCRE UOW recovery token)
      words 12 through 15   not used
```

SUB FUNC = X'0019' DFSESPR0 External SubSys PROGRAM REQUEST recursive call record

```
      word   2 -- External SubSystem name
      word   3 -- bytes 0-1 PSTID    (IMS dependent region ID)
                   byte  2   EZSGFL   (DFSEZS connection status byte1)
                   byte  3   EZSLFL   (DFSEZS connection status byte2)
      word   4 -- byte  0   EZSEFL1  (DFSEZS thread startup status)
                   byte  1   EZSEFL2  (DFSEZS thread commit status)
                   byte  2   EZSEFL3  (DFSEZS thread termination status)
                   byte  3   EZSEFL4  (DFSEZS termination flag)
      word   5 -- bytes 0-1 not used
                   bytes 2-3 External SubSys exit routine return code
      words  6 through  7   not used
      words  8 through 11   LCRETOKN (DFSLCRE UOW recovery token)
      words 12 through 15   not used
```

SUB FUNC = X'0020' DFSESPR0 External SubSys NOT OPERATIONAL (SNOX) exit record

```
      word   2 -- External SubSystem name
      word   3 -- bytes 0-1 PSTID    (IMS dependent region ID)
                   byte  2   EZSGFL   (DFSEZS connection status byte1)
                   byte  3   EZSLFL   (DFSEZS connection status byte2)
      word   4 -- byte  0   EZSEFL1  (DFSEZS thread startup status)
                   byte  1   EZSEFL2  (DFSEZS thread commit status)
                   byte  2   EZSEFL3  (DFSEZS thread termination status)
                   byte  3   EZSEFL4  (DFSEZS termination flag)
      word   5 -- bytes 0-1 not used
                   bytes 2-3 External SubSys exit routine return code
      words  6 through  7   not used
      words  8 through 11   LCRETOKN (DFSLCRE UOW recovery token)
      words 12 through 15   not used
```

Figure 53 shows an example of an external subsystem trace with both X'57' and X'58' record IDs. The ESS trace is called the subsystem (SST) trace in a dump.

```
****************************************************
***TRACE PRINTED FROM OLDEST TO MOST CURRENT ENTRY**
****************************************************
  FUNCTION          WORD 0    WORD 1    WORD 2    WORD 3    WORD 4    WORD 5    WORD 6    WORD 7
ESI5  CTL INIT    5700198F  04030019  F1F0F0F1  00000000  00000000  00000000  00000000  00000000
ESI3  IDENT       570019B8  04020020  F1F0F0F1  00000800  00000000  00000000  00000000  00000000
ESS4  MESSAGE     570019BD  00150015  F1F0F0F1  00000000  00000000  00000000  00000000  00000000
ESI3  R-I-D       570019C6  04020021  F1F0F0F1  00002C00  00000000  00000000  00000000  00000000
ESS3  LOGGING     570019CF  00160014  F1F0F0F1  00000000  00000000  00000000  00000000  00000000
ESCT  CRT THRD    58003165  02900005  F1F0F0F1  0001CC0C  81000000  00000000  00000000  00000000
FESP  SYNC STA    580035D0  03070016  F1F0F0F1  0001CC0C  8C100000  42048000  03F00000  00000000
ESI3  RRE REQ     570035EC  04020023  F1F0F0F1  00008C00  00000000  00000000  00000000  00000000
ESI3  XS ECHO     570035F1  04020024  F1F0F0F1  00008C00  00000000  00000000  00000000  00000000
ESI3  R-I-D       570035F6  04020021  F1F0F0F1  00008C00  00000000  00000000  00000000  00000000
ESS3  LOGGING     57003608  00160014  F1F0F0F1  00000000  00000000  00000000  00000000  00000000
ESCT  CRT THRD    58003A8F  02900005  F1F0F0F1  0001CC0C  81000000  00000000  00000000  00000000
FESP  SYNC STA    58003AA1  03070016  F1F0F0F1  0001CC0C  8C100000  01080000  00000000  00000000
ESP1  COM PREP    58003AC8  02940008  F1F0F0F1  0001CC0C  8C500000  00000000  00000000  00000000
FESP  SYNC END    58003ACB  03070017  F1F0F0F1  0001CC0C  8CD00000  01080000  00000080  02940000
FESP  SYNC STA    58003B1A  03070016  F1F0F0F1  0001CC0C  8CD00000  010C0000  00002080  00000000
ESP2  COM CONT    58003B3D  02950009  F1F0F0F1  0001CC0C  8CD40000  00000000  00000000  00000000
FESP  SYNC END    58003B44  03070017  F1F0F0F1  0001CC0C  9CCC0000  010C0000  000020C0  02950000
FESP  SYNC STA    58003BA3  03070016  F1F0F0F1  0001CC0C  9CCC0000  42080000  00000000  00000000
FESP  SYNC END    58003BA4  03070017  F1F0F0F1  0001CC0C  9CCC0000  42080000  00000080  02950000
FESP  SYNC STA    58003BDF  03070016  F1F0F0F1  0001CC0C  9CCC0000  420C0000  00002080  00000000
ESD5  TRM THRD    58003BE7  02910002  F1F0F0F1  0001CC0C  9CCC0000  00000000  00000000  00000000
FESP  SYNC END    58003BF1  03070017  F1F0F0F1  0001CC0C  95000C00  420C0000  00002080  00000000

GOBAL ESET PREFIX
BLOCK AT 00BED480
                  PGES   00BED4A4  PLES   00000000  SCDAD  00BEA2B0  PCPE   00000000  ESGL
                  PICT   00000001  POCT   00000001         00000000

*** GLOBAL ESET BLOCK ***
00BED4A4  00000000 0059E9C0 00BED480 F1F0F0F1  40404040 E2E8E2F1 C4E2D5D4 C9D5F1F0
00BED4C4  40404040 40404040 D9F14040 0FC4E2D7  00B4DB40 001547C0 00A0C4C0 80B4DB57
00BED4E4  0FC4E2D7 00B4DB40 00153868 80A0C550  80B4DB57 108021DE 00000022 0059F9C8
00BED504  00000000 00000000 00000000 00000000  00005628 005B85A0 FF412B0C 00000000
00BED524  8C000000 009DC078 0059F998
```

Figure 53. Example of an External Subsystem Trace (SST)

Scheduler Trace

When you use the /TRACE SET ON TABLE SCHD command, IMS enables the scheduler trace. When you specify OPTION LOG, IMS sends these entries to the log as type X'67FA' records.

The diagrams in Figure 54 through Figure 59 show the formats of the scheduler trace records for function codes X'41' through X'44', X'47', and X'48' listed in Table 26 on page 164.

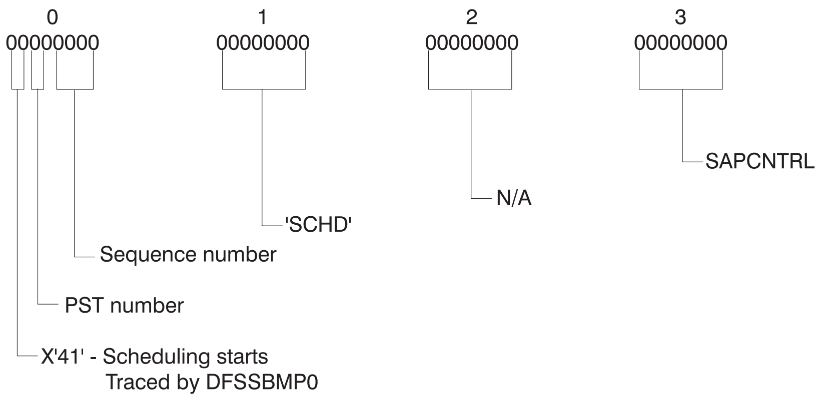


Figure 54. Scheduler Trace Record Format for Function Code X'41'

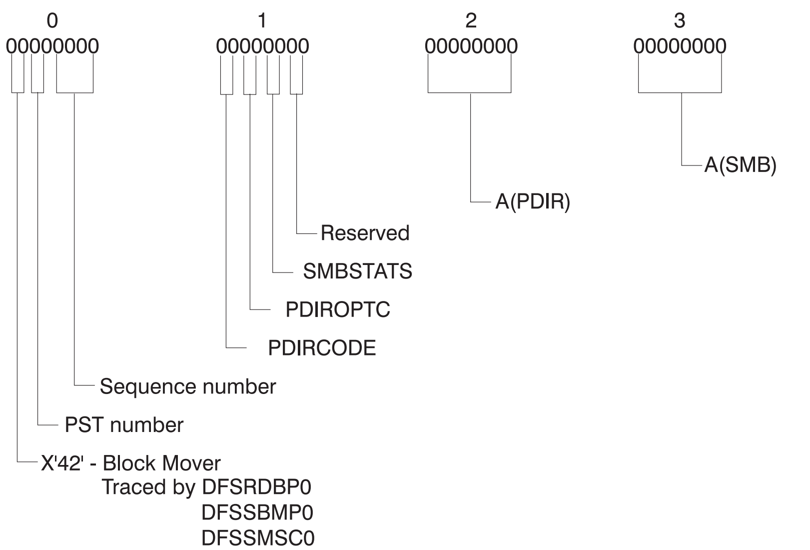


Figure 55. Scheduler Trace Record Format for Function Code X'42'

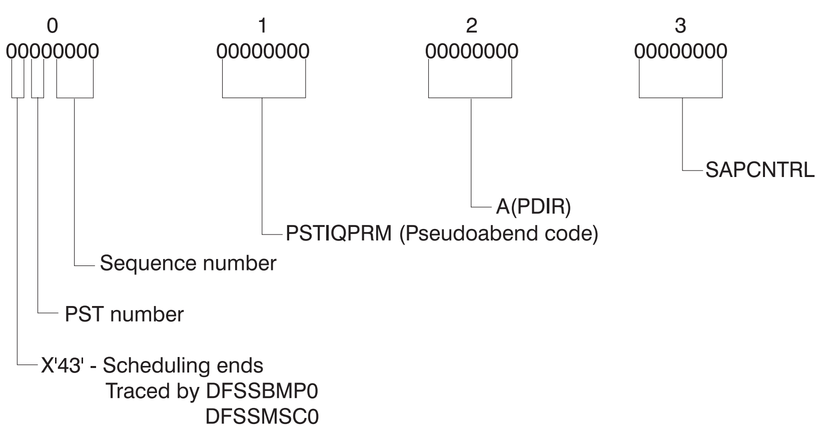


Figure 56. Scheduler Trace Record Format for Function Code X'43'

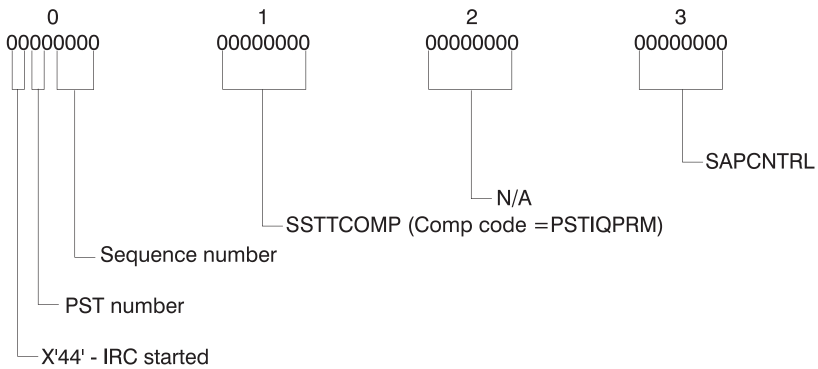


Figure 57. Scheduler Trace Record Format for Function Code X'44'

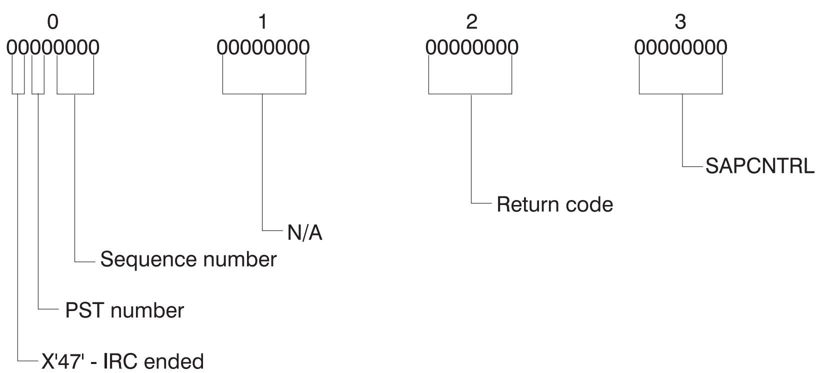


Figure 58. Scheduler Trace Record Format for Function Code X'47'

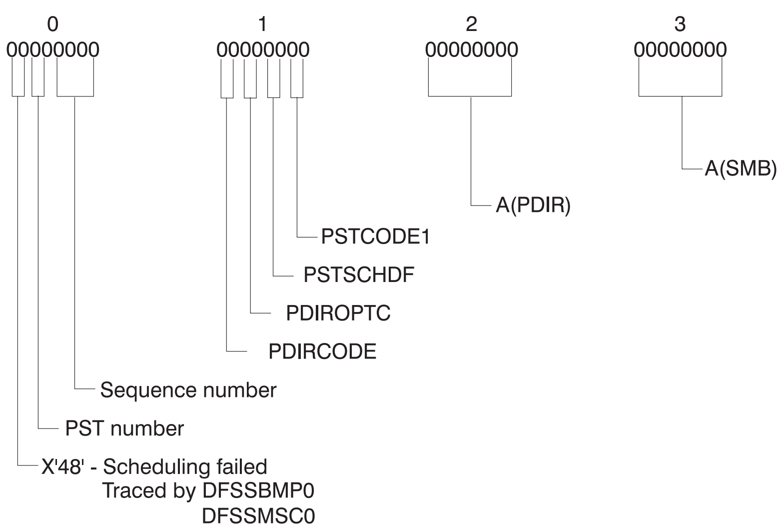


Figure 59. Scheduler Trace Record Format for Function Code X'48'

Figure 60 on page 192 shows an example of a scheduler trace.

```
**STR         SCHEDULER TRACE
****************************************************
***TRACE PRINTED FROM OLDEST TO MOST CURRENT ENTRY**
****************************************************
  FUNCTION          WORD 0    WORD 1    WORD 2    WORD 3
IRC START         4408C28B  00000000  00000000  00000001
SCHED START       4108C2B3  E2C3C8C4  0098E050  40008001
SCHED START       410DC30E  E2C3C8C4  0082F050  40028001
BLOCK MOVER       420DC315  44040549  00ACA058  00A95688
SCHED END         430DC316  00000000  00A26AE4  40028001
IRC START         440DC3BE  00000000  00000000  00020001
BLOCK MOVER       4208C470  8402854B  00AC98E8  00A93190
SCHED END         4308C471  0098E050  00A27F24  40008001
IRC START         4408C4EB  00000000  00000000  00000001
SCHED START       4108C513  E2C3C8C4  0098E050  40008001
BLOCK MOVER       4208C51A  8402854B  00AC98E8  00A930C0
SCHED END         4308C51B  0098E050  00A27F24  40008001
IRC START         4408C668  00000000  00000000  00000001
SCHED START       4108C690  E2C3C8C4  0098E050  40008001
SCHED START       410DC853  E2C3C8C4  0082F050  40028001
BLOCK MOVER       420DC875  8402B050  00AC76C8  00000000
SCHED END         430DC876  00000000  00A29304  40028001
BLOCK MOVER       4208C92C  8402854B  00AC98E8  00A931F8
SCHED END         4308C92D  0098E050  00A27F24  40008001
IRC START         4408C9E3  00000000  00000000  00000001
SCHED START       4108CA0B  E2C3C8C4  0098E050  40008001
BLOCK MOVER       4208CA83  8402854B  00AC98E8  00A93190
SCHED END         4308CA84  0098E050  00A27F24  40008001
IRC START         4408CAC3  00000000  00000000  00000001
SCHED START       4108CAEB  E2C3C8C4  0098E050  40008001
BLOCK MOVER       4207CC47  8402C54B  00AB92D8  00A5F260
SCHED END         4307CC48  00988050  00A2BDC4  40008001
IRC START         4407CDDA  00000000  00000000  00000001
SCHED START       4107CE02  E2C3C8C4  00988050  40008001
BLOCK MOVER       4208CE86  8402854B  00AC98E8  00A931F8
SCHED END         4308CE87  0098E050  00A27F24  40008001
IRC START         4408CECA  00000000  00000000  00000001
SCHED START       4108CEF2  E2C3C8C4  0098E050  40008001
BLOCK MOVER       4208CF7C  8402854B  00AC98E8  00A930C0
SCHED END         4308CF7D  0098E050  00A27F24  40008001
IRC START         4408D017  00000000  00000000  00000001
SCHED START       4108D03F  E2C3C8C4  0098E050  40008001
BLOCK MOVER       4208D046  8402854B  00AC98E8  00A949F0
SCHED END         4308D047  0098E050  00A27F24  40008001
IRC START         4408D0A9  00000000  00000000  00000001
SCHED START       4108D0D1  E2C3C8C4  0098E050  40008001
BLOCK MOVER       4208D1A6  8402854B  00AC98B0  00A90B60
SCHED END         4308D1A7  0098E050  00A25C44  40008001
IRC START         4408D227  00000000  00000000  00000001
SCHED START       4108D24F  E2C3C8C4  0098E050  40008001
BLOCK MOVER       4208D331  8402854B  00AC98E8  00A931F8
SCHED END         4308D332  0098E050  00A27F24  40008001
IRC START         4408D36E  00000000  00000000  00000001
```

Figure 60. Example of a Scheduler Trace

Storage Manager Trace

The storage manager trace writes a record each time it is called to allocate a pool, get a buffer, or release a buffer. The storage manager traces requests from the following pools: HIOP, CIOP, CESS, SPAP, EMHB, FPWP, LUMP, LUMC.

You can enable the storage manager trace during IMS initialization with the STRG= option in the DFSVSMxx PROCLIB member, or online using the /TRACE command. The /TRACE SET ON TABLE STRG command activates the trace and sends the output to an internal trace table. When you specify OPTION LOG on the /TRACE command, IMS sends the output to the system log or external trace data set. For information about using the /TRACE command, see *IMS Version 7 Command Reference*.

You can format the internal trace table using the Offline Dump Formatter under IPCS with either the VERBX command or the Interactive Dump Formatter panels. To format the trace records, any storage manager control blocks, and pool storage, you can specify ALL as the poolid as shown in the following example.

FMTIMS ...(POOL,NAME,ALL),...or you can specify FMTIMS (TRACE, NAME, SM).

For detailed information on formatting the trace table, see the Offline Dump Formatter section in this chapter or in *IMS Version 7 Utilities Reference: System*.

To locate the storage manager trace in a formatted dump, look for eye-catcher **SMTR.

To locate the trace tables in an unformatted dump, look for the trace identifier SM in the trace table header record.

The following diagrams show the format of each storage manager trace record.

WORD Ø	WORD 1	WORD 2	WORD 3	WORD 4	WORD 5	WORD 6	WORD 7
Control Information	Pool name	Variable pool size	Variable pool address fixed pool upper limit	Ø	Caller's return address	Return code	Ø

Figure 61. TRACE ID = X'5F03' (Allocate trace record)

WORD Ø	WORD 1	WORD 2	WORD 3	WORD 4	WORD 5	WORD 6	WORD 7
Control Information	Pool name	Buffer request size	Buffer address	Address of caller's ECB	Caller's return address	Return code	Current pool size

Figure 62. TRACE ID = X'5F04' (Get trace record)

WORD Ø	WORD 1	WORD 2	WORD 3	WORD 4	WORD 5	WORD 6	WORD 7
Control Information	Pool name	Ø	Buffer address	Address of caller's ECB	Caller's return address	Return code	Current pool size

Figure 63. TRACE ID = X'5F05' (Release trace record)

Latch Trace

When you use the /TRACE SET ON TABLE LATC command, IMS traces events related to its internal serialization services (latch manager, use manager, and system locate control function) to an internal table. Figure 64 on page 194 shows the general format of a latch trace entry:

WORD 0			WORD 1	WORD 2	WORD 3	WORD 4	WORD 5	WORD 6	WORD 7
I	S	SEQ NUM	ENTRY TYPE						

Figure 64. Format of a Latch Trace Entry

where

I
: One-byte trace ID field. This byte indicates the type of the trace entry. It is always X'70' for latch trace entries.

S
: One-byte trace subtype field. Not used for latch trace entries.

SEQ NUM
: Two-byte trace sequence number assigned by the IMS trace component.

ENTRY TYPE
: Four-byte printable character string, indicating the type of latch trace entry. The entry types are documented in detail below.

Words 2 through 6 contain data specific to each trace entry, as described in the following sections.

Latch Manager Trace Entries

```
Sub Function: X'01' Get latch (GET)
Description:  Get a latch
     word   1  -- Caller's SAP address
     word   2  -- Latch name
     word   3  -- Caller's return address
     word   4  -- Resource header address
     word   5  -- 1st halfword = latch level;
                  2nd halfword = flags from latch manager parmlist
     word  6/7 -- 8-byte STCK value
```

```
Sub Function: X'02' - Upgrade latch (GETU)Description:  Upgrade a latch from shared to exclusive
     word   1  -- Caller's SAP address
     word   2  -- Latch name
     word   3  -- Caller's return address
     word   4  -- Resource header address
     word   5  -- 1st halfword = latch level;
                  2nd halfword = flags from latch manager parmlist
     word  6/7 -- 8-byte STCK value
```

```
Sub Function: X'03' - Release latch (REL)
Description:  Release a latch
     word   1  -- Caller's SAP address
     word   2  -- Latch name
     word   3  -- Caller's return address
     word   4  -- Resource header address
     word   5  -- 1st halfword = latch level;
                  2nd halfword = flags from latch manager parmlist
     word  6/7 -- 8-byte STCK value
```

```
Sub Function: X'04' - Recover latch (RCOV)
Description:  Recover a latch
     word   1  -- SAP, TCB, or ASCB address
     word   2  -- Latch name
     word   3  -- Caller's return address
     word   4  -- 0
     word   5  -- 1st halfword = latch level;
                  2nd halfword = flags from latch manager parmlist
     word  6/7 -- 8-byte STCK value
```

Use Manager Trace Entries

Latch Manager Trace Entries:

WORD Ø			WORD 1	WORD 2	WORD 3	WORD 4	WORD 5	WORD 6	WORD 7
I X'7Ø'	S	SEQ NUM	Entry type: 'USE'	Block type	Call ID	Work ID	Block address	SAP address	Caller's return address

Figure 65. USE — Inuse request trace entry

WORD Ø			WORD 1	WORD 2	WORD 3	WORD 4	WORD 5	WORD 6	WORD 7
I X'7Ø'	S	SEQ NUM	Entry type: 'LOK'	Block type	Call ID	Work ID	Block address	SAP address	Caller's return address

Figure 66. LOK — Lock request trace entry

WORD Ø			WORD 1	WORD 2	WORD 3	WORD 4	WORD 5	WORD 6	WORD 7
I X'7Ø'	S	SEQ NUM	Entry type: 'CON'	Block type	Call ID	Work ID	Block address	SAP address	Caller's return address

Figure 67. CON — Connect request trace entry

WORD Ø			WORD 1	WORD 2	WORD 3	WORD 4	WORD 5	WORD 6	WORD 7
I X'7Ø'	S	SEQ NUM	Entry type: 'MRG'	Block type	Call ID	Work ID	Block address	SAP address	Caller's return address

Figure 68. MRG — Merge request trace entry

WORD Ø			WORD 1	WORD 2	WORD 3	WORD 4	WORD 5	WORD 6	WORD 7
I X'7Ø'	S	SEQ NUM	Entry type: 'INQ'	Block type	Call ID	Work ID	Block address	SAP address	Caller's return address

Figure 69. INQ — Inquiry request trace entry

WORD Ø			WORD 1	WORD 2	WORD 3	WORD 4	WORD 5	WORD 6	WORD 7
I X'7Ø'	S	SEQ NUM	Entry type: 'NUSE'	Block type	Call ID	Work ID	Block address	SAP address	Caller's return address

Figure 70. NUSE — Nouse request trace entry

WORD Ø			WORD 1	WORD 2	WORD 3	WORD 4	WORD 5	WORD 6	WORD 7
I X'7Ø'	S	SEQ NUM	Entry type: 'NLOK'	Block type	Call ID	Work ID	Block address	SAP address	Caller's return address

Figure 71. NLOK — Unlock request trace entry

WORD Ø			WORD 1	WORD 2	WORD 3	WORD 4	WORD 5	WORD 6	WORD 7
I X'7Ø'	S	SEQ NUM	Entry type: 'NCON'	Block type	Call ID	Work ID	Block address	SAP address	Caller's return address

Figure 72. NCON — Disconnect request trace entry

WORD Ø			WORD 1	WORD 2	WORD 3	WORD 4	WORD 5	WORD 6	WORD 7
I X'7Ø'	S	SEQ NUM	Entry type: 'RCOV'	'SAP'	Block type	SAP address	Ø	Ø	Caller's return address

Figure 73. RCOV (SAP level) — Use recovery performed at the SAP (ITASK) level trace entry

WORD Ø			WORD 1	WORD 2	WORD 3	WORD 4	WORD 5	WORD 6	WORD 7
I X'7Ø'	S	SEQ NUM	Entry type: 'RCOV'	'TCB'	Block type	Ø	TCB address	Ø	Caller's return address

Figure 74. RCOV (TCB level) — Use recovery performed at the TCB level trace entry

WORD Ø			WORD 1	WORD 2	WORD 3	WORD 4	WORD 5	WORD 6	WORD 7
I X'7Ø'	S	SEQ NUM	Entry type: 'RCOV'	'MEM'	Block type	Ø	ASCB address	Ø	Caller's return address

Figure 75. RCOV (address space level) — Use recovery performed at the address space level trace entry

System Locate Control Function Entries

WORD Ø			WORD 1	WORD 2	WORD 3	WORD 4	WORD 5	WORD 6	WORD 7
I X'7Ø'	S	SEQ NUM	Entry type: 'SLC0'	Block type	Work ID	Call ID	' '	SAP address	Caller's return address

Figure 76. SLC0 — Locate a block and issue a use manager inuse call against it

WORD Ø			WORD 1	WORD 2	WORD 3	WORD 4	WORD 5	WORD 6	WORD 7
I X'7Ø'	S	SEQ NUM	Entry type: 'SLC1'	Block type	Work ID	Call ID	' '	SAP address	Caller's return address

Figure 77. SLC1 — Locate a block and issue a use manager nouse call against it

```
**LTR                                        LATCH TRACE
****************************************************
***TRACE PRINTED FROM OLDEST TO MOST CURRENT ENTRY**
****************************************************
  FUNCTION      WORD 0     WORD 1  WORD 2  WORD 3  WORD 4    WORD 5    WORD 6    WORD 7
COMMON LATCH    70006A98   GET     QMGR    SHR     00005F28  00290000  065975F0  8004BABE
COMMON LATCH    70006A99   REL     QMGR    ANY     00005F28  00290000  065975F0  800EAA62
COMMON LATCH    70006A9A   GET     QMGR    SHR     00005F28  00290000  065975F0  8004BABE
COMMON LATCH    70006A9B   REL     QMGR    ANY     00005F28  00290000  065975F0  800EAA62
COMMON LATCH    70006A9C   GET     DCSL    SHR     05B581B0  00030000  065975F0  8004F2C4
COMMON LATCH    70006A9E   GET     LOGL    EXCL    05B58F70  002F0000  065975F0  85B0EED4
COMMON LATCH    70006A9F   REL     LOGL    EXCL    05B58F70  002F0000  065975F0  85B0E53C
COMMON LATCH    70006AA1   GET     QMGR    SHR     00005F28  00290000  065975F0  8004BABE
COMMON LATCH    70006AA2   REL     QMGR    ANY     00005F28  00290000  065975F0  800EAA62
COMMON LATCH    70006AA3   REL     DCSL    SHR     05B581B0  00030000  065975F0  80046012
COMMON LATCH    70006AA4   NUSE    ALLW    ....    05F66060  00000000  06 75F0  06D2CCC2
COMMON LATCH    70006AA6   GET     LOGL    EXCL    05B58F70  002F0000  065975F0  85B0EED4
COMMON LATCH    70006AA7   REL     LOGL    EXCL    05B58F70  002F0000  065975F0  85B0E53C
COMMON LATCH    70006AAD   GET     LOGL    EXCL    05B58F70  002F0000  065975F0  85B0EED4
COMMON LATCH    70006AB2   REL     LOGL    EXCL    05B58F70  002F0000  065975F0  85B0E53C
COMMON LATCH    70006AB4   GET     TCTB    EXCL    05B71858  00130000  065975F0  85B5CB3A
COMMON LATCH    70006AB5   REL     TCTB    EXCL    05B71858  00130000  065975F0  85B5CD78
COMMON LATCH    70006AB6   GET     SMGT    EXCL    05C47288  002B0000  065975F0  85B0BAEA
COMMON LATCH    70006AB7   REL     SMGT    EXCL    05C47288  002B0000  065975F0  85B0BBB6
COMMON LATCH    70006AB8   GET     PDRB    EXCL    05BA9E90  00150000  065975F0  85B5AB26
COMMON LATCH    70006AB9   GET     PSBP    SHR     05B587A0  00160000  065975F0  85B5ABE6
COMMON LATCH    70006ABA   REL     PDRB    EXCL    05BA9E90  00150000  065975F0  85B5AED4
COMMON LATCH    70006ABB   REL     PSBP    ANY     05B587A0  00160000  065975F0  85B5AF90
COMMON LATCH    70006ABC   GET     SUBQ    SHR     05B71418  00200000  065975F0  85B4291E
COMMON LATCH    70006ABD   REL     SUBQ    SHR     05B71418  00200000  065975F0  85B42A60
COMMON LATCH    70006ABE   GET     SUBQ    SHR     05B71430  00200000  065975F0  85B4291E
COMMON LATCH    70006ABF   REL     SUBQ    SHR     05B71430  00200000  065975F0  85B42A60
COMMON LATCH    70006AC7   GET     QMGR    SHR     00005F28  00290000  06597790  8004BABE
COMMON LATCH    70006AC8   REL     QMGR    ANY     00005F28  00290000  06597790  800EAA62
COMMON LATCH    70006ACA   SLC0    LNBQ    .. -    C4D3C1F3  40404040  06597790  05B7BD2A
COMMON LATCH    70006ACB   GET     VLQB    SHR     00BD2230  00260000  06597790  800511A4
COMMON LATCH    70016ACC   USE     CNT     DLA3    05FB4060  07926568  06597790  05B312AE
COMMON LATCH    70006ACD   REL     VLQB    ANY     00BD2230  00260000  06597790  800511A4
COMMON LATCH    70006ACE   REL     SCHD    ANY     05B58660  00120000  06597790  85B60CB4
```

Figure 78. Example of a Latch Trace

Queue Manager Trace

The queue manager trace provides information about relevant queue manager functional and exceptional events. Use the trace under the direction of IBM support personnel when problems are suspected in the queue manager area.

You can turn on the queue manager trace in two ways:

- During IMS online initialization with the QMGR parameter in the DFSVSMxx IMS.PROCLIB member
- During online operation, with the `/TRACE` command.

You can specify trace output destination and tracing volume on both the QMGR parameter and the /TRACE command.

If you send output to the common trace table, you can format the table using the Offline Dump Formatter under IPCS, using either the VERBX command or the Interactive Dump Formatter panels. If you send the output to an external data set, you can use the File Select and Formatting Print utility (DFSERA10) with exit routine DFSERA60 to format the trace entries.

To locate the queue manager trace in a formatted dump, look for eye catcher **QMGR. To locate the trace table in an unformatted dump, look for the trace identifier QM in the trace table header record.

Related Reading: For information about:
- The QMGR parameter, see *IMS Version 7 Installation Volume 2: System Definition and Tailoring*.
- The /TRACE command, see *IMS Version 7 Command Reference*.
- The common trace table interface, see “Common Trace Table Interface” on page 162.
- The Offline Dump Formatter, see “Formatting IMS Dumps Offline” on page 129.
- The File Select and Formatting Print utility, see *IMS Version 7 Utilities Reference: System*.

Format of Trace Records

The following diagrams show the format of the trace records. Each trace record has a trace function code of X'4E' and is X'20' bytes long.

This figure depicts the trace (low level) record format of the following functions with these subfunction codes (SC):

SC	FUNCTION
X'00'	GET PREFIX
X'01'	CANCEL INPUT
X'02'	GET UNIQUE
X'03'	GET NEXT
X'04'	DEQUEUE
X'05'	SAVE
X'06'	REJECT
X'07'	DELETE
X'08'	CANCEL OUTPUT (LOG)
X'09'	CANCEL OUTPUT (NOLOG)
X'0C'	ENQUEUE (FIFO)
X'0D'	ENQUEUE (LIFO)
X'0E'	REENQUEUE (FIFO)
X'0F'	REENQUEUE (LIFO)
X'10'	REPOSITION
X'11'	AOI COMMAND INPUT
X'12'	AOI MESSAGE TO MASTER
X'13'	AOI CANCEL UEHB
X'14'	AOI TERMINATION

| **X'17'** UNUSED OP CODE

| **X'18'** UNUSED OP CODE

| **X'19'** UNUSED OP CODE

| **X'1A'** INSERT PREFIX

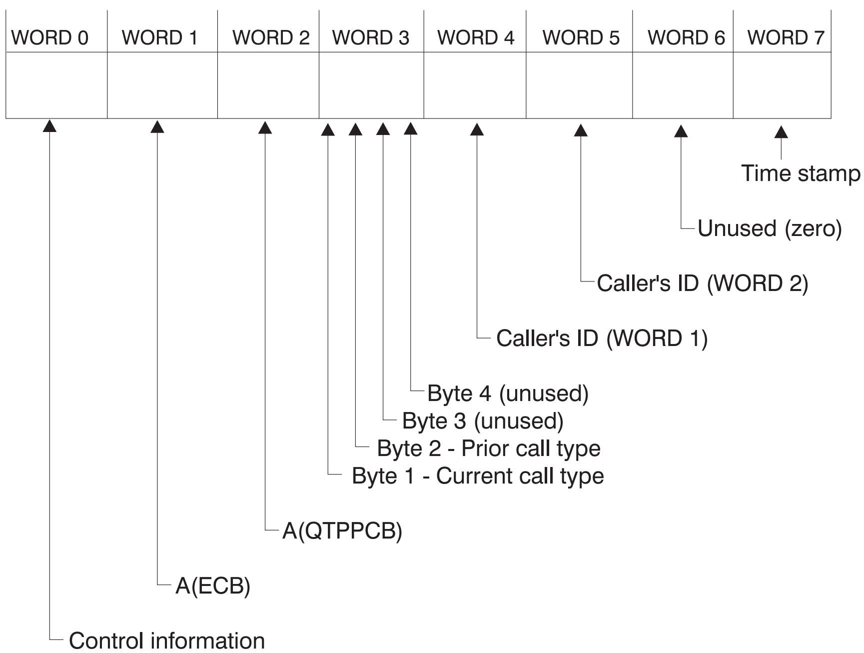
| **X'1C'** CONDITIONAL ENQUEUE (FIFO)

| **X'1D'** CONDITIONAL ENQUEUE (LIFO)

| **X'1E'** TRANSFER

| **X'1F'** NOTE/POINT

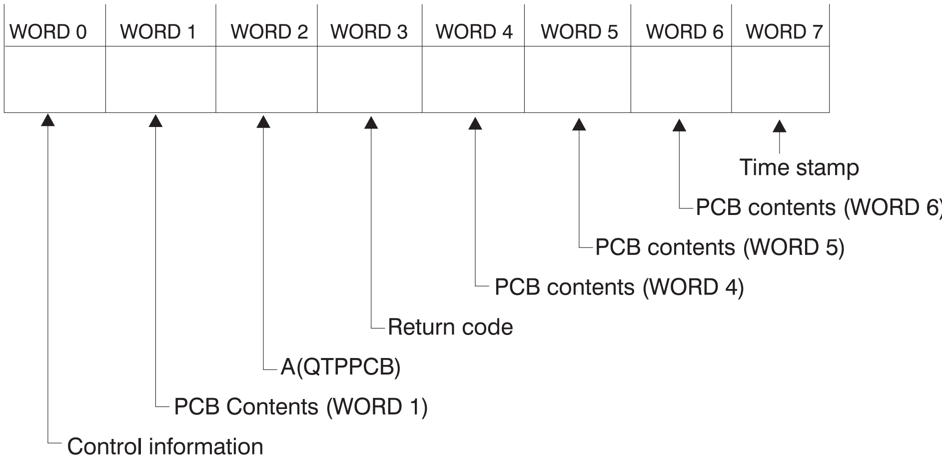
|



|

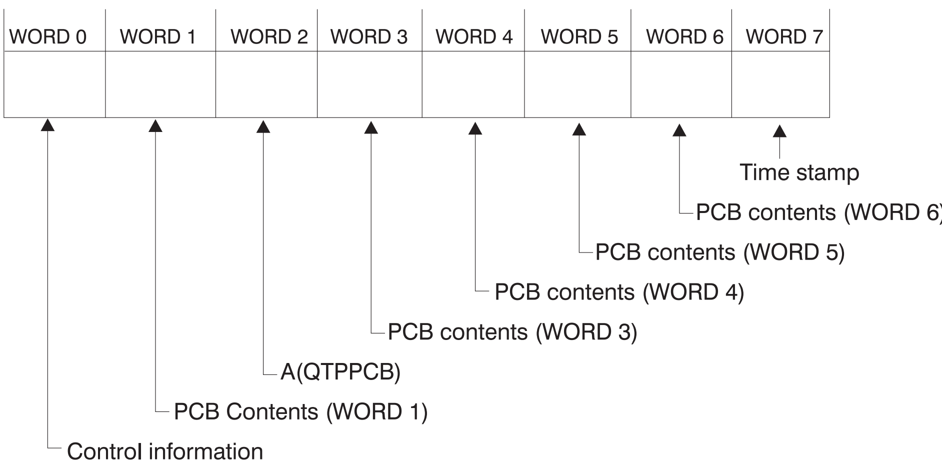
| This figure depicts the trace (medium level) record format of the following function with this subfunction
| code:

FUNCTION	**Subfunction Code**
EXIT FROM QUEUE MANAGER	X'21'



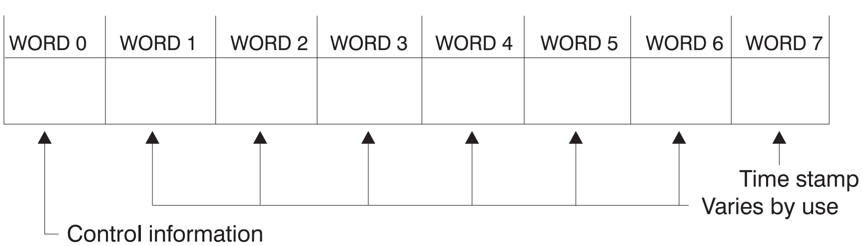
This figure depicts the trace (medium level) record format of the following function with this subfunction code:

FUNCTION	**Subfunction Code**
ENTRY TO QUEUE MANAGER	X'20'



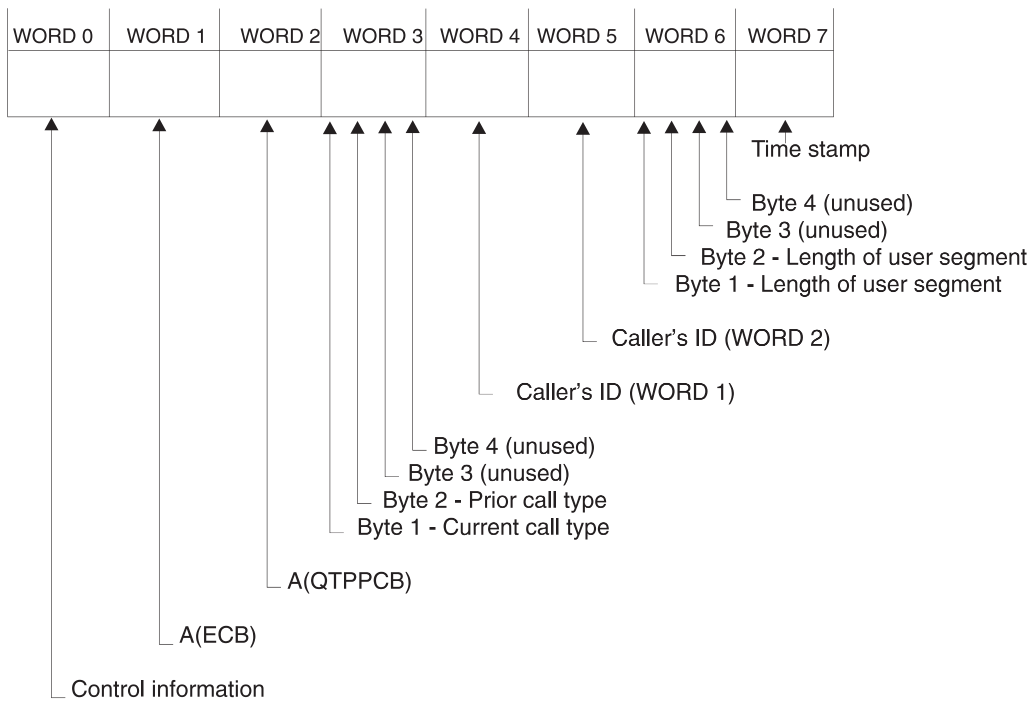
This figure depicts the trace (medium level) record format of the following function with this subfunction code:

FUNCTION	**Subfunction Code**
Special- Not Applicable	X'22'



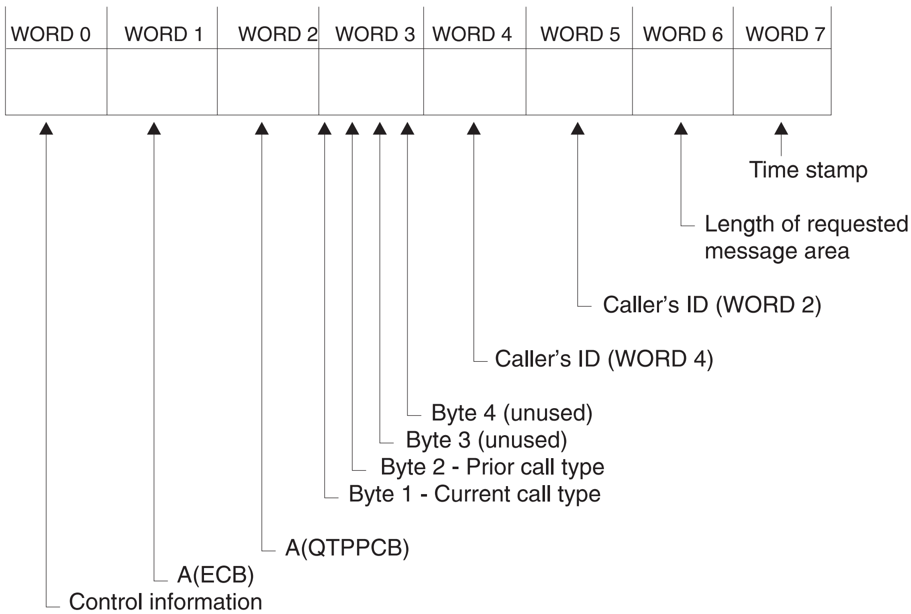
This figure depicts the trace (low level) record format of the following functions with these subfunction codes:

FUNCTION	Subfunction Code
INSERT MOVE	X'08'
MESSAGE REROUTE	X'15'
INSERT MOVE SPANNABLE	X'1B'



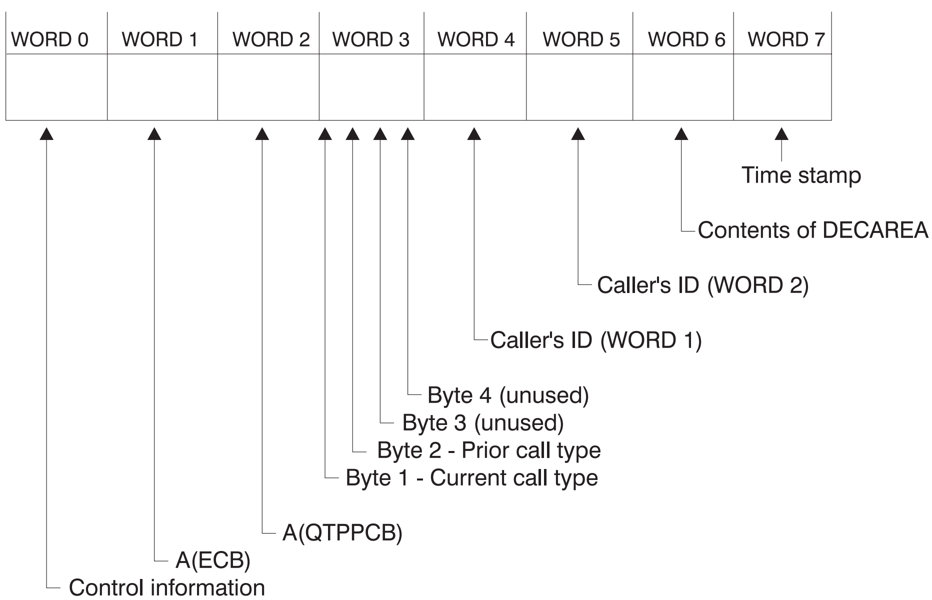
This figure depicts the trace (low level) record format of the following function with this subfunction code:

FUNCTION	Subfunction Code
INSERT LOCATE	X'0A'



This figure depicts the trace (low level) record format of the following function with this subfunction code:

FUNCTION	**Subfunction Code**
RELEASE	X'16'



Shared Queues Interface Trace

The shared queues interface trace provides information about errors associated with the interface between IMS and CQS. Examples of errors that are traced are:

- CQS Request errors
- CQS Inform errors
- Service errors
- Storage errors

| Use this trace under the direction of IBM support personnel when problems are suspected in the interface
| between IMS and CQS.

You can turn on the shared queues interface trace in two ways:
- During IMS online initialization, with the SQTT parameter in the DFSVSMxx IMS.PROCLIB member
- During online operation, with the /TRACE command.

Each trace entry is X'20' bytes long.

You can specify trace output destination and tracing volume on both the SQTT parameter and the /TRACE command.

The /TRACE SET ON TABLE SQTT command activates the trace and sends the output to an internal trace table that consists of 126 entries. If you specify OPTION LOG on the /TRACE command, IMS sends the output to the system log or an external trace data set in groups of 126. Other parameters control the volume of output.

You can format trace table entries with the Offline Dump Formatter under IPCS, using either the VERBX parameter or the Interactive Dump Formatter panels. You can use the File Select and Formatting Print utility (DFSERA10) with exit routine DFSERA60 to format the trace entries written to an external data set.

To locate the shared queues interface trace in a dump, look for eyecatcher **SQTT.

To display the status of the trace, use the /DISPLAY TRACE command

Related Reading: For information about:
- The SQTT parameter, see *IMS Version 7 Installation Volume 2: System Definition and Tailoring*.
- The /TRACE command, see *IMS Version 7 Command Reference*.
- The common trace table interface, see “Common Trace Table Interface” on page 162.
- The Offline Dump Formatter, see “Formatting IMS Dumps Offline” on page 129.
- The File Select and Formatting Print utility, see *IMS Version 7 Utilities Reference: System*.

Chapter 8. DB—Database Service Aids

The information contained in this chapter addresses service aids and diagnostic techniques used to analyze IMS database problems. This chapter specifically addresses the following items:

- The job control block (JCB) trace that traces the last few DL/I calls and related status codes for a specific logical database
- The DL/I test program that is used to test DL/I calls against a given database [a]
- The COMPARE statement SNAP [a]
- Output from SNAP calls [a]
- SNAPs on exceptional conditions [a]
- The DL/I call image capture service aid that traces database application activity and generates DL/I test program control statements to simulate that activity
- A technique for approaching DL/I analysis in a batch environment
- Locating database related traces
- A description of the DL/I trace record formats
- A retrieve trace that records the control flow between the retrieve module and other database routines
- Program isolation-related problem analysis
- A few additional problem determination tools for specific sequential buffering problems
- GSAM control blocks dump [a]

Note: [a] In a Database Control (DBCTL) environment, this information applies only to Batch Message Processing (BMP) programs, not Coordinator Controller (CCTL) programs.

The Job Control Block (JCB) Trace

The job control block (JCB) trace is one of most useful diagnosis tools for any application problem that may occur. It is an easy way to determine the last five calls that were issued, and what their return codes were.

Analyzing the JCB trace is a good way to identify application problems. For example, sometimes the application programmer forgets to handle a certain status code, even though it identifies an error situation. Seeing the call and its return code draws attention to this application error and makes it much easier to resolve.

The JCB trace is always on (you don't need to do anything explicit to turn it on), and it is included in every IMS dump. The job control block portion of the dump is formatted under the heading, `JCB`. The JCB trace is a wrap-around area that consists of six 2-byte entries. The first entry begins at offset X'20' in the JCB portion of the dump and is followed immediately by the remaining five entries. As the entries are inserted into the trace area, previous entries are shifted left.

In the first through fifth entries, the first byte identifies the DL/I call (see the "Code" column of Table 29 on page 206). The second byte in these entries contains the second character of the DL/I I/O status code (return code). The sixth entry contains information about the call that immediately preceded the call that was being processed at the time of the abend; this is sometimes useful in determining what had been going on prior to the failure. The function of that prior call is identified in field JCBPREVF at offset X'2A' of the JCB, and the status code of the prior call is in field JCBPREVR at offset X'2B'.

Related Reading: The DL/I status codes and return codes are defined in *IMS Version 7 Application Programming: Database Manager*.

If one of the 2-byte fields in the JCB trace contains X'0000', this means that no call was made.

Example: The JCB trace might contain the following six fields:

`0000 0000 0205 0305 0140 0140`

This trace indicates that only four calls were made, the most recent of which was a get-unique call (either `GU` or `GHU`), as indicated by the first-byte code of X'01'. The status code for the most recent call was X'40'.

Sample JCB Trace

A sample JCB dump is shown in Figure 79.

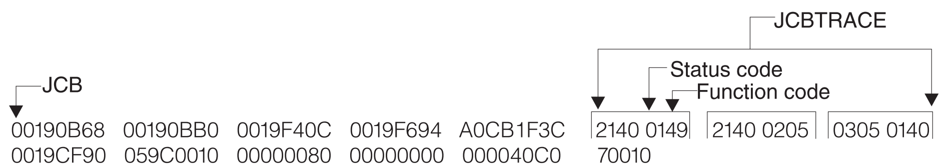


Figure 79. Example of a Job Control Block (JCB) Dump

JCB Trace Call Function Codes

The DL/I user call encoded functions are contained in DFSDLA00, at label FUNCSTRT. They are listed in Table 29.

Table 29. DL/I User Call Encoded Functions

Code	Call	Code	Call
00	GB	65	LOG
00	GBT	70	RELOAD
00	GHB	80	OPEN
00	GHBT	81	CLOSE
00	GHP	82	STOP
00	GL	83	CHANGE
00	GND	84	SNAP
00	GNX	85	CHECK POINT
00	GP	86	STATISTICS REQUEST
01	GHU	87	CMD
01	GU	88	GCMD
03	GHN	89	ROLB
03	GN	90	PURGE
04	GHNP	A0	UNLD
04	GNP	A1	GSCD
20	DLET or REPL	A2	MOVE
21	REPL	B0	SPND
22	DLET	F1	XSET
23	DLET or REPL	F2	XRUN
40	ISRT	F3	XFIN
41	ISRT	F4	XSCD
42	ASRT	F5	XOFF
60	DEQ		

DL/I status codes and return codes are defined in *IMS Version 7 Application Programming: Database Manager*.

Data Language/I Test Program—DFSDDLT0

The DL/I test program is an IMS application that issues calls to DL/I based on control statement information. For diagnostic purposes, this allows you a means of separating the application logic from DL/I logic to resolve problems.

Optionally, the DL/I test program compares the results of the calls with expected results provided in control statements. If the returned results do not match the expected results, the program can provide a SNAP of any combination of DL/I blocks, I/O buffer pool, subpools 0-127, and the entire region. The test program can also invoke the IMS SNAP call, by means of its control statements, during normal execution to provide diagnostic information on the DL/I calls that are executing correctly.

Related Reading: For details on the functions of this program and instructions for using it, refer to the chapter on testing an application program in *IMS Version 7 Application Programming: Database Manager*.

COMPARE Statement SNAPs

When a DL/I call does not produce the results you expect, you can use the `COMPARE` statement to compare the actual results of a call with the expected results. The normal output of this statement usually provides enough information to determine what is causing the problem.

When the output from a `COMPARE` statement does not provide enough information, you can use the `SNAP` option of the `COMPARE` statement to obtain additional diagnostic information. Specifically, the I/O buffer pool and the DL/I blocks are dumped. You can use the generated diagnostic output, in conjunction with *IMS Version 7 Failure Analysis Structure Tables (FAST) for Dump Analysis* in order to determine the cause of the user abend you are diagnosing.

Attention: The `COMPARE SNAP` statement is a call to DL/I. Therefore, when a `SNAP` option is issued, some data in the captured area might be changed as a result. To prevent inadvertent change to data that is not involved in the problem, use a `COMPARE SNAP` statement only for the specific data you believe is involved in the problem.

For more information about the `COMPARE` statement `SNAP` option, see *IMS Version 7 Application Programming: Database Manager*.

SNAP Output

Some control blocks are always dumped. Others are dumped only when you request them in the SNAP options.

These control blocks are always dumped:

The SCD

The PST (save areas related to the current DL/I task are a part of the PST)

The retrieve trace area

The following SNAP option requests dump the control blocks or buffers listed:

- A request for the buffer pool dumps:

  OSAM buffer pool prefix and buffer pool, if present

  VSAM subpool prefix, buffer prefix and subpools, and the buffer handler trace table

  Header for the DL/I, dispatcher, scheduler, and latch trace tables

  The DL/I trace table

The dispatcher trace table
The scheduler trace table
The latch trace table
Hierarchical direct (HD) trace table, if present
Sequential buffering control blocks and buffer pools, if present

- A request for the current DB PCB or all PSB-related control block dumps:

  Delete/replace work areas, when allocated
  ENQ/DEQ trace table, if present
  PSB and PSB work areas
  PCB information, including JCB, DSGs, level table, and PRL
  The block of SDBs, SDB expansion blocks, and generated SDBs
  DMB directories
  DMBs for the current PSB
  | PNTs associated with partition DMBs

  If you also requested buffers, a request for the current DB PCB or all PSB-related control block dumps:

  Any HISAM/QSAM buffers
  Any VSAM LRECs for each qualifying DSG

- A request for the entire region, or subpools 0-127, dumps the entire region or the subpools.

  A SNAP of the entire region or subpools is sent to a SNAP data set.

  If the SNAP destination is the IMS log, the request is changed to a SNAP of all control blocks, regardless of other option specifications.

  A region or subpool SNAP, when requested, appears before any additional SNAPs that were requested.

  If the destination of the SNAP is the IMS log, you can select and format these records (type X'67FD') from the log by using the File Select and Formatting Print utility with exit routine, DFSERA30. For information about this utility, see *IMS Version 7 Utilities Reference: System*.

SNAPs on Exceptional Conditions

IMS produces SNAPs of DL/I control blocks on the IMS log (or the CICS system log) in the following exceptional situations:

- A pseudoabend condition is encountered in a DL/I module.
- A system or user abend occurs for either a message region or a batch message region.

Control block SNAPs are produced in the same format as those produced by a DL/I SNAP call specifying ALL or YYY as SNAP options.

The SNAP IMS log records are record type X'67', subrecord type X'FF'. You can select these log records from the IMS log with the File Select and Formatting Print utility (DFSERA10). You can format output selected from the log with the formatting edit routine DFSERA30. For information about this utility, see *IMS Version 7 Utilities Reference: System*.

SNAP Specific

Internal IMS functions can request the snapping of specific virtual storage areas by issuing a SNAP Specific call to DFSERA20.

The following IMS functions request or use the SNAP Specific facility:

- SBSNAP option, on completion of calls from IMS modules to the Sequential Buffering buffer handler
- SBESNAP option, during SB evaluation

- SB COMPARE option, when detecting a mismatch between the buffer content that the SB buffer handler was returning to the OSAM buffer handler and the content of the database block as it is stored on DASD

For IMS online regions and CICS, these SNAPs are written to the IMS log. For IMS batch regions, these SNAPs can be written to either the log or to a data set specified on another DD statement.

When written to the log, the IMS log records have a record type X'67' and a subrecord type X'E'. The value of the low-order half-byte of the subrecord type depends on the IMS function that requests the SNAP. The subrecord types are:

X'ED' SBESNAP option

X'EE' SBSNAP option

X'EF' SB COMPARE option

The formatting edit routine DFSERA30 can format output selected from the log (see “File Select and Formatting Print Utility” on page 127).

DL/I Call Image Capture

DL/I call image capture (module DFSDLTR0) allows you to trace and record all DL/I calls issued by an application program. The trace output is in a format acceptable as input to the DL/I test program DFSDDLT0.

| **Related Reading:** For information about DFSDDLT0, see *IMS Version 7 Application Programming: Design Guide*.

DL/I call image capture is a useful debugging tool because it allows you to rerun an application program and generate the DL/I calls necessary to duplicate the condition that caused the program failure. This run provides you with documentation to assist you in problem determination.

You can run the trace in either a batch or an DB/DC environment.

Batch Environment

In a batch environment, you start DL/I call image capture using the DLITRACE control statement in the DFSVSAMP DD data set. The control statement allows you to trace either all DL/I calls issued by an application program or a range of calls. The traced information can be put in a sequential data set, the IMS log data set, or into both concurrently.

Related Reading: For information about:

- Writing the trace table externally to DASD, a tape data set, or the online log data set (OLDS), see the DFSVSMxx procedure in *IMS Version 7 Installation Volume 2: System Definition and Tailoring*.
- Using a call image capture statement to trace DL/I calls, see *IMS Version 7 Application Programming: Database Manager*.

Online Environment

In a DB/DC, DCCTL, or DBCTL environment, you start and terminate DL/I call image capture by issuing the /TRACE command from the master terminal (DB/DC and DCCTL only) or from the system console. For example, to trace full-function database calls for a named PSB and send the output to an external data set, issue the following command:

```
/TRACE SET ON PSB psbname OPTION LOG
```

Related Reading: For information about:

- The /TRACE command, see *IMS Version 7 Command Reference*.
- Writing the trace table externally to DASD, a tape data set, or the online log data set (OLDS), see “Write Trace Tables Externally” on page 5.
- Allocating the external trace data sets (DFSTRA01 and DFSTRA02) used by the IMS online systems, see *IMS Version 7 Installation Volume 1: Installation and Verification*.

How to Retrieve DL/I Call Image Capture Data from the Log Data Set

If trace data is sent to the IMS log data set, you can retrieve it using the File Select and Formatting Print utility (DFSERA10) and the DL/I call image capture exit DFSERA50.

To use DFSERA50, you need to insert a DD statement defining the output data set in the DFSERA10 input stream. The default ddname for this DD statement is TRCPUNCH. The statement must specify LRECL=80.

Related Reading: For information about the File Select and Formatting Print utility, see *IMS Version 7 Utilities Reference: System*.

DL/I Analysis

These debugging suggestions are useful in a batch environment. The information is valid for DL/I or DBB regions.

Before diagnosing abends in a batch region, review the external conditions. Verify that your environment is correct by asking the following questions:

- Are the JOBLIB/STEPLIB DD statements pointing to the correct libraries?
- Are the PSBLIBs and DBDLIBs at the same level as the JOBLIB/STEPLIB modules?
- If running with an ACBLIB, was the ACBGEN run under the same level of IMS you are currently running on?
- Were the databases correctly allocated and intact before starting the current run?

IMS Abends

In general, there are two causes of abend dumps:

- An abend issued by an IMS module (user abend)
- A program check within an IMS module (system abend)

All IMS abends are issued with the dump option.

User Abends

There are two methods by which an IMS module can issue an abend when an error condition is detected.

- The first method is the standard ABEND macro issued by the code at the point of error detection. With this method, the PSW, at entry to the abend, points at the code within the module that both detected the error and issued the abend.
- With the second method, the module that detects the error does not issue the abend, but instead passes the error indication back to the program request handler, which then issues a real abend. The PSW, at entry to the abend, now points to the program request handler rather than to the module that detected the error. The pseudoabend method is used by DL/I modules that abend an application program in a dependent region but do not abend the IMS control region in a DB/DC environment.

When the DL/I test program is being used as the application program, the pseudoabend is passed back to the test program rather than to the program request handler. This allows the test program to request a formatted SNAP rather than just an abend dump.

Dump Analysis—General

The following represents initial considerations for dump analysis:

- The first request block (RB) on the RB chain represents the IMS batch region controller (DFSRRC00); the second RB on the RB chain represents the batch program controller (DFSPCC30). This module (DFSPCC30) always links to the application program named in the parameter field of the EXEC statement; therefore, the application program must be represented by the third RB. However, if the application program uses an IMS service, and that service abended, then the third RB points to the offending IMS routine.
- The last two SVRBs represent ABEND and ABDUMP. The register contents at the time of abend are usually found in the first abend SVRB. Other areas used to hold the register contents at abend time are the IMS STAE work area (DFSFSWA0) and the RTM work area in MVS.
- There are two PSTs in a batch environment. One is used for all application calls and the second is used for background write whenever it is activated.
- Each PST has a 15-level save area set as part of the PST; at abend time, abdump prints the save areas associated with the active PST.
- At abend time the IMS STAE routine gets control to flush the database buffers and close the log data set. It builds six additional save areas and chains them to the last save area in the active PST. The IMS STAE routine is partially contained within module DFSPCC30 and has an entry ID starting with the characters PCE.
- Most IMS modules use register 12 as a base register.

Dump Analysis—Detailed

To thoroughly analyze a dump, you need to understand the save area, DL/I call sequence, and the buffer handler request sequence. This section discusses each of these elements.

Save Areas

A DL/I call passes from the application program to the DL/I language interface (DFSLI000), to the program request handler (DFSPR000), to the batch nucleus (DFSBNUC0), and then to the DL/I call analyzer (DFSDLA00).

If everything works properly, the save area trace shows the contents of the registers at entry to the application program, the program request handler, and the DL/I analyzer. The DL/I analyzer passes the first save area in the PST to a DL/I module. This PST save area is the first save area below the save area that holds the contents of the registers at entry to the DL/I analyzer.

The contents of register 1 at entry to the DL/I analyzer is a pointer to the PST. This is the only register passed to the analyzer (the user call list pointer is passed to the analyzer in PSTIQPRM).

If the abend is a program check or an inline abend, the save area trace always gives a true indication of the flow of control between DL/I modules and the current depth of save area set usage. Most DL/I modules “or”X'01' with the low-order byte of register 14 on return to a higher-level module.

If the abend is a pseudoabend, the save areas below the analyzer might have been reused and therefore would not reflect the conditions at the time the abend condition was detected; for example, the DB Monitor might have been called by the analyzer.

DL/I Call Sequence

You can determine the current DL/I call and the sequence of calls leading up to the failure by scanning the DL/I trace table. Find the last entry made in the trace table by using the current entry pointer and then scanning backward in the table for the last entry made by the DL/I analyzer (entry code AA). This entry represents the current DL/I call.

You can determine the call sequence by continuing the backward scan, noting each entry made by the analyzer. Along with the call function, the analyzer also records the PCB address that was passed in the user's call list.

Buffer Handler Request Sequence

The buffer handler router traces each request to the buffer handler from a DL/I module. When the router receives the request, it passes the request to the OSAM buffer handler, the VSAM track recovery interface, or the VSAM interface module. When the call is complete, control returns to the router. The router obtains the next available trace table entry and stores information describing the input and output for the buffer handler call.

By looking at all buffer handler entries between two DL/I analyzer DFSDLA00 entries (two specific DL/I calls), you can determine all requests made to the buffer handler to satisfy any specific DL/I call. A typical request to the buffer handler is a GET by relative byte address from the retrieve module. The entry made for this GET by relative byte address has a function code of E2, the RBA requested, and, if the request was satisfied (return code 0), the address of the segment read into the buffer pool.

Generalized DL/I Problem Analysis

The following sequence of steps describes a method of problem analysis. Not all DL/I abends can be diagnosed using this sequence, but you can use it as a guide to DL/I debugging. All numbers are in hexadecimal.

1. The approaches described below are true if the IMS dependent region subtask appears in the dump.

   Look at the user's call list for the current or last call. PSTIQPRM points to the call list. For all dependent region types, if the reentrant DL/I language interface, DFSLI000, is used, the user's call list address can be found in the contents of register 1 in the save area set at entry point to DFSPROX0-115 from the save area trace.

   To find the last call parameters in a MPP or BMP dump, locate module DFSFSWA0 in the dump. Scan this module for ECP. At offset X'104' from ECP is a pointer to the parameters that made the last call to DL/I.

   To find the PCBs in an MPP or BMP dump, find DIRCA in module DFSFSWA0. The word immediately following DIRCA contains the address of an area of storage obtained by the GETMAIN macro instruction. This area contains the PCB list and all non-GSAM PCBs. The format of this area is:

   - At offset X'14' is the beginning of the PCB list passed to the program.
   - Immediately following the end of the PCB list is a copy of the I/O PCB, if one exists.
   - The next PCB (and subsequent PCBs) follow the end of the I/O PCB.

   Because they exist elsewhere in the dump, GSAM PCBs are not copied here. The pointers to the GSAM PCBs can be found in the PCB list at offset X'14'.
2. If the abend occurred after the DL/I analyzer received the call, but before the application program got control back, the last call entry (code AA) in the DL/I trace table matches the current call. Use the technique described in "DL/I Call Sequence" on page 211 to determine the call sequence as far back as possible, noting the PCB address associated with each call.
3. Compare the contents of PSTDBPCB to the PCB address in the last call entry in the trace table. If they are different, index maintenance is probably in control using its PCB within the PSB. Check the save area trace to verify this.
4. Find the current PCB from the address in the trace table, and then find the JCB. Starting 14 bytes into the JCB are six 2-byte trace entries for the last six calls issued against this PCB. The oldest entry is at displacement 14 and the newest entry is at displacement 1E. The first byte of an entry is the encoded call function and the second byte is the last half of the status code for that call. For example, an 0140 is an entry for a GET UNIQUE call that resulted in a blank status code. This trace is maintained by the DL/I analyzer at the completion of the call. (See also Figure 79 on page 206.)
5. Look at the contents of JCBLEVIC. If the call is a get or an insert, the retrieve module zeros this word at entry and then stores a pointer to each level table entry when it completes the call for that particular

level. If the word is zero, retrieve is still trying to satisfy the call at the root level. Generally, JDBLEVIC reflects the lowest level satisfied during the current or last get or insert call.

6. Check each level table entry to see if it holds a valid current position. Valid position is indicated by the absence of the empty bit in FLAG1 (LEVEMPTY in LEVF1, bit 1 byte 1). If this bit is off (valid position), LEVSDB points to the SDB currently in use or the last one used for this level. At the same time, LEVTTR, which contains either a relative byte address (RBA) or a relative record number (RRN), should match the current position saved in the SDB (SDBPOSC). In addition, if the database is HISAM, LEVSEGOF matches SDBPOSN. This is the offset into the current relative record number.
7. Look at the key feedback area—level table position. The key feedback area contains the fully concatenated key of the segment currently positioned on. If a level table entry contains a valid position, the contents of the key feedback area for that level is the key (if any) of the segment whose SDB is pointed to by LEVSDB and whose database position is contained within LEVTTR and LEVSEGOF. The contents of the key feedback area are never cleared or blanked out. Therefore, unless the level table entry indicates it has a valid position, the residue in the key feedback area might not be meaningful.
8. Map the database structure involved in the failure. Starting with the root SDB, which you can find with a pointer in the JCB (JCBSDB1), take each SDB in the sequence it is found in the dump and examine the field SDBPARA at displacement 20. This is a pointer to the parent SDB (the root SDB points at the PCB). (See Figure 34 on page 102 to see how the prefix of a segment is mapped.) Map the structure according to SDBPARA; the result should match the logical structure defined at PSBGEN time. When mapping the structure, note the contents of SDBTARG at displacement 28. If this field is nonzero, the segment is involved in either logical relationships or indexing. The code in the high-order byte indicates which is the case.
9. Use the DL/I trace table to analyze the sequence of buffer handler calls. (See Figure 99 on page 234.) The buffer handler trace is the most useful debugging tool for DL/I. The trace is available in both batch and DB/DC environments, and the entries are identical.

   Get calls are the most common, so this section uses a get call as an example. In an attempt to satisfy a get call, the retrieve module must examine a segment or a series of segments to see if it meets the call requirements. All segments must be requested from the buffer handler and the request must be in the form of an RBA, RRN, or a specific key request.

   The most common request from retrieve to the buffer handler is a byte locate. The parameters passed to the buffer handler are the function (byte locate), the RBA requested, and the data set in which the RBA exists. At exit to the buffer handler router, the next available trace entry is obtained and the code of the function requested is stored in the first byte. The buffer handler function codes are listed in the PST DSECT under PSTFNCTN. The byte locate function code is E2. The second byte of the trace entry is the relative PST number responsible for the request, which in batch is always an 01.

   Along with the function code, the DSG and RBA are placed into the entry at displacements 8 and 1C, respectively. When the call to the buffer handler (OSAM or VSAM) is completed, the results are traced, again by the buffer handler router. The return code is stored in the third byte. The return codes are listed in the PST DSECT under PSTRTCDE. If the call is successful, the address of the segment within the buffer pool is stored at displacement C. This trace now shows each segment (RBA) requested by retrieve; by examining the buffer pools the contents of the segments and their prefixes can be seen. RBAs found in the trace table can be compared to position fields in the SDB and level table to accurately re-create the get call. Figure 34 on page 102 shows the mapping of the prefix of a segment.

Locating Database-Related Traces

The importance of the DL/I-related traces and the information that they convey is discussed in "DL/I Analysis" on page 210. Figure 80 on page 214 shows how to locate the following traces:

- Retrieve trace—records the flow through the retrieve module subroutines.
- JCBTRACE—traces the status of the prior six calls.
- DL/I trace—shows calls made to the call analyzer, buffer handler, and hierarchic direct space management, as well as information on Delete/Replace.
- LOG data set—records database changes, before and after images.

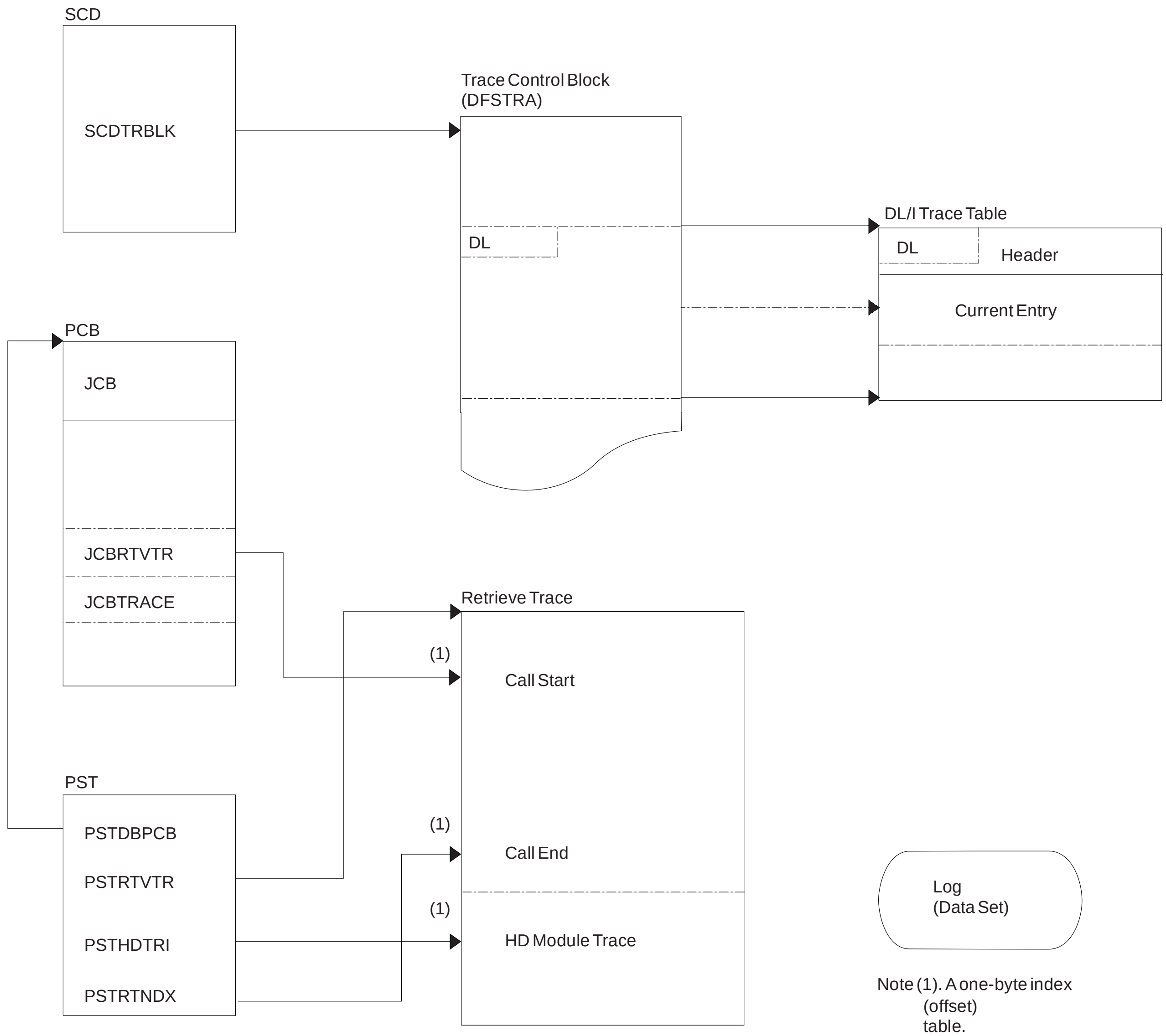


Figure 80. How to Locate the Database Traces

DL/I Trace

The DL/I trace table is a combined trace consisting of entries from DL/I calls, the DL/I buffer handler, DL/I OPEN/CLOSE, HD space management, lock activity (using PI or IRLM), OSAM, DFP interface, and ABENDU0427.

For information about starting and stopping the DL/I trace, writing the trace table to the log, and finding the trace tables in a dump, see “Common Trace Table Interface” on page 162. This section also lists the function codes for the DL/I and lock traces.

Be aware that the DL/I trace and the DL/I Call Image trace are different traces. The DLITRACE statement in IMS.PROCLIB member DFSVSMxx turns on the DL/I Call Image trace, not the DL/I trace.

If the trace was written to the log, you must use the File Select and Formatting Print utility (DFSERA10) with an exit routine (DFSERA40 or DFSERA60) to format and print the trace entries.

The Database Tracking trace entries are described in “Database Tracker Trace Entries” on page 411.

Using the DL/I Trace

The DL/I trace facility is an important diagnostic tool that can help you determine the cause of a problem. Frequently, a problem occurs as a result of the interaction between two separate tasks. Interpreting the DL/I trace entries can be the best way of determining what each task was doing, and when.

Example: An IMS Fast Path application receives an abend 1027, and the user reports the problem to the support staff. Some of the steps the diagnostician might take are:

1. Look up the abend code in *IMS Version 7 Failure Analysis Structure Tables (FAST) for Dump Analysis*. This book indicates that the return code is in register 15.
2. Look at register 15 in the dump; it contains a value of X'0D'.

   IMS Version 7 Failure Analysis Structure Tables (FAST) for Dump Analysis indicates that this return code indicates that an enqueue or dequeue call was issued by module DBFBENQ0, and the return code from DFSLRH00 was X'12', indicating an invalid call.
3. Look at the DL/I trace to determine what resource was involved (if the DL/I trace was on at the time of the abend). If the DL/I trace was not on, it might be necessary to re-create the problem with DL/I trace on.

   The list of trace entry IDs in “DL/I Trace Formats” indicates that one of the trace entries is “Exclusive control ENQ/DEQ PI trace entry” (Figure 94 on page 225). This would probably be a good place to start the DL/I trace analysis.

What you learn from the DL/I trace might help you:

- Identify and resolve an application error
- Review APAR descriptions to see if this problem has occurred previously
- Report the problem to IBM

DL/I Trace Formats

The figures in this section show the formats of the most commonly used DL/I trace entries. They are included to help you understand the DL/I trace entries in order to communicate more effectively with IBM software support representatives and to build a valid search argument.

Exception: Not every trace entry is shown. The entries that are not described can be obtained by assembling IDLIVSAM TRACENT from IMS.SDFSMAC.

Trace ID	Description of Content of Trace Entry
X'0C'	DL/I OPEN/CLOSE for each data set (Figure 81 on page 216).
X'31'	HD space management: Get space for the segment (Figure 82 on page 217).
X'32'	HD space management: Free space for the segment (Figure 82 on page 217).
X'34'	HD space management: Get space close to root anchor point (Figure 82 on page 217).
X'B1'	HD space management: Get space set by backout or DELETE/REPLACE (Figure 82 on page 217).
X'B2'	HD space management: Free space set by backout (Figure 82 on page 217).
X'60'	OSAM I/O initiated trace entry (Figure 83 on page 218).
X'62'	OSAM trace entry for OPEN/CLOSE/EOV trace entries (Figure 84 on page 218).
X'AA'	Analyzer entry (Figure 85 on page 219).
X'AC'	Database call analyzer entry (DBCTL only) (Figure 86 on page 219).
X'C4'	DELETE/REPLACE (Figure 87 on page 220).

X'C7' Exclusive control deadlock detection trace entry (without IRLM, in Figure 88 on page 220; with IRLM in Figure 89 on page 221).

X'C8' Lock request manager entry (DFSLMGR0) (Figure 90 on page 222).

X'C9' Lock request manager exit (DFSLMGR0) (Figure 91 on page 223).

X'CA' Exclusive control ENQ/DEQ (program isolation) entry (for non-Fast Path, Figure 92 on page 224; for Fast Path, Figure 94 on page 225).

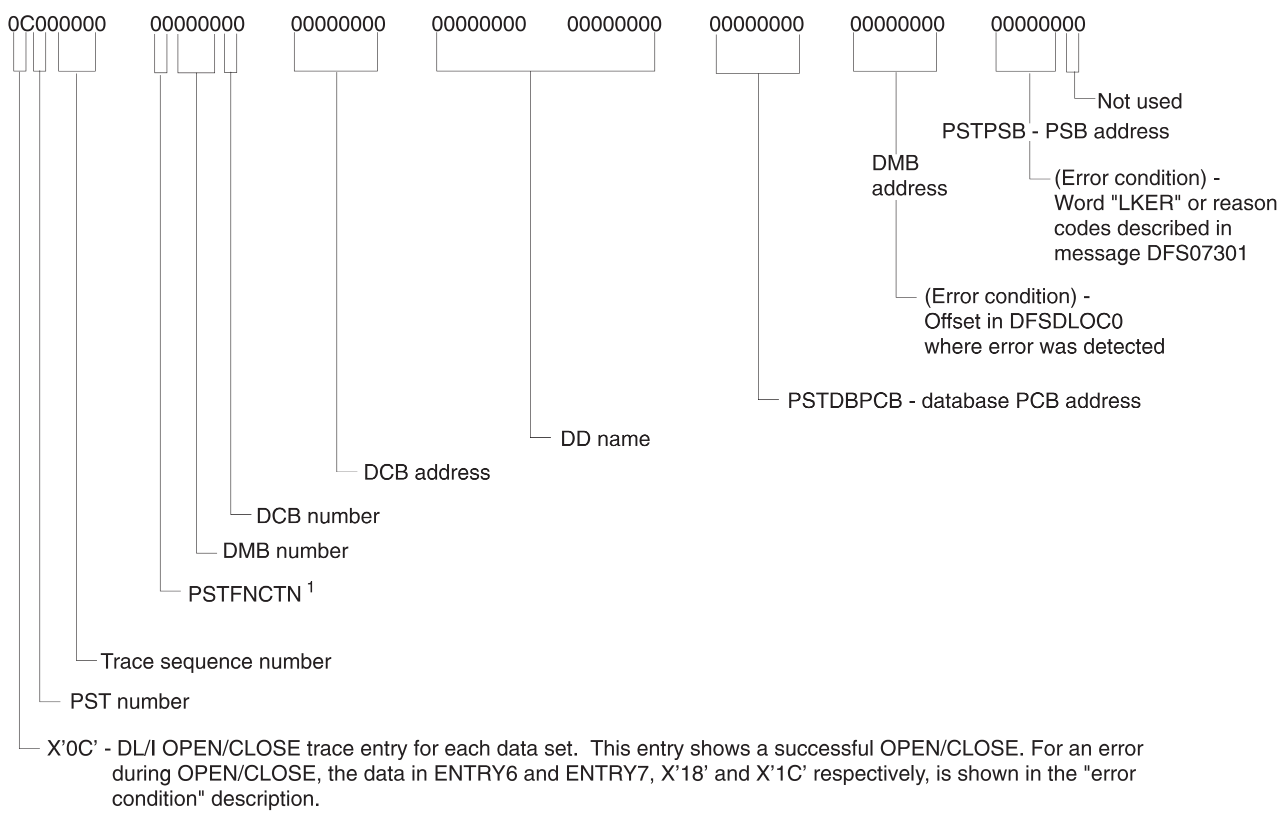
X'CA'X'08' PI DL/I call trace entry (Figure 93 on page 225).

X'CB' PI trace lock elapsed time (Figure 95 on page 226).

X'CC' Lock request handler (DFSLRH00) entry (Figure 96 on page 226).

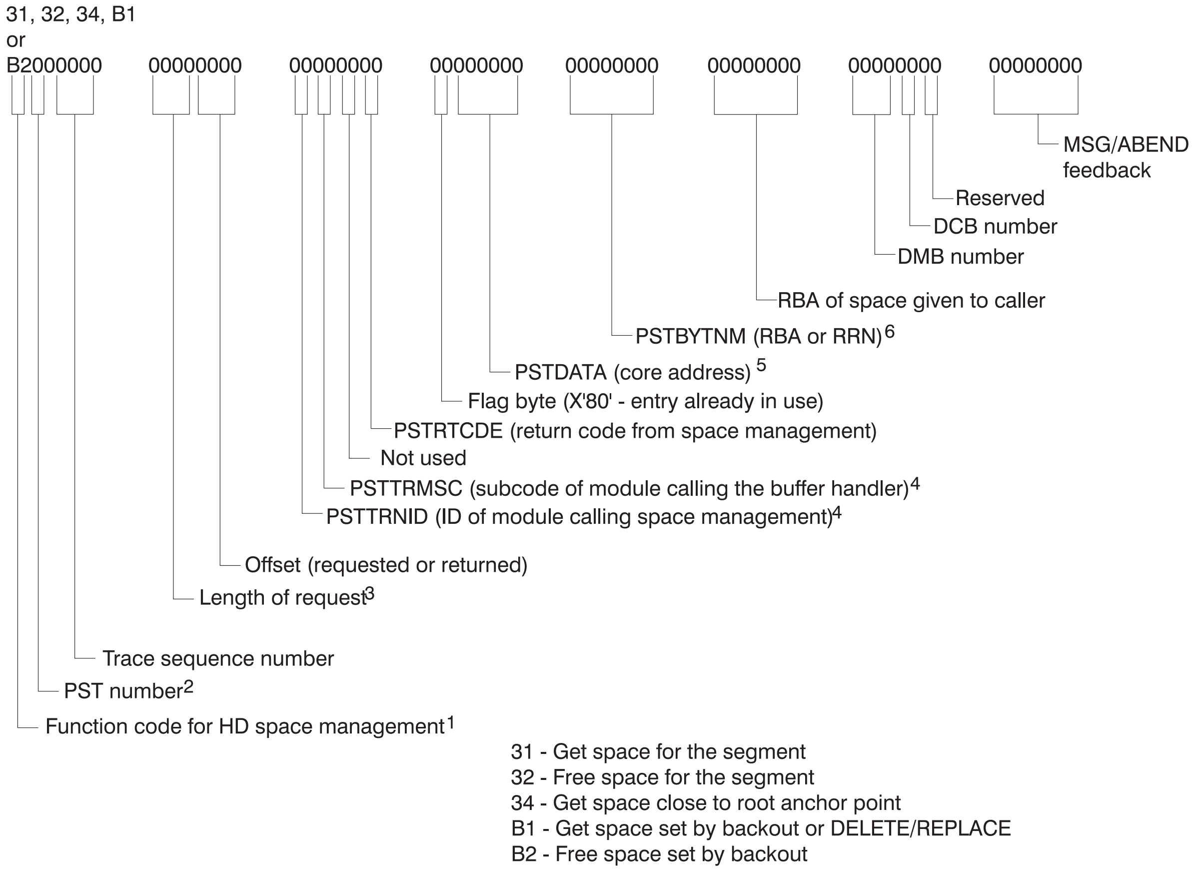
X'DA' VSAM JRNAD or UPAD exit (Figure 97 on page 229).

X'DB'- X'FA' Buffer handler trace (Figure 98 on page 230).



[1] Use the OPEN/CLOSE section of Table 32 on page 276

Figure 81. X'0C' Trace Entry



This trace entry can be helpful when a U0832 abend shows you are pointing to free space. It might also be helpful with U085x abends.

Figure 82. X'31', X'32', X'34', X'B1', and X'B2' Trace Entries

Notes to Figure 82:

1. You need the X'32' entries to resolve this problem.
2. Numbers 3 and 4 are very important. In most cases, the segment was deleted by another task (see PST number), and this task (see PST number) tried to enqueue on the segment that waited while the other PST finished its processing. During the attempt, an FSE was found and abend U0832 resulted. An IMS internal error usually causes this problem.
3. The length of the segment that was freed. (Use the FSE chart in the *IMS Version 7 Administration Guide: Database Manager* for an explanation of FSEs.)
4. See Table 34 on page 233 for the module names that correspond to the module IDs.
5. The real storage address of the segment during the time of deletion.
6. The PSTBYTNM is the key field in the trace table. Look for a X'32' entry with the PSTBYTNM field equal to the PSTBYTNM field found in the buffer trace.

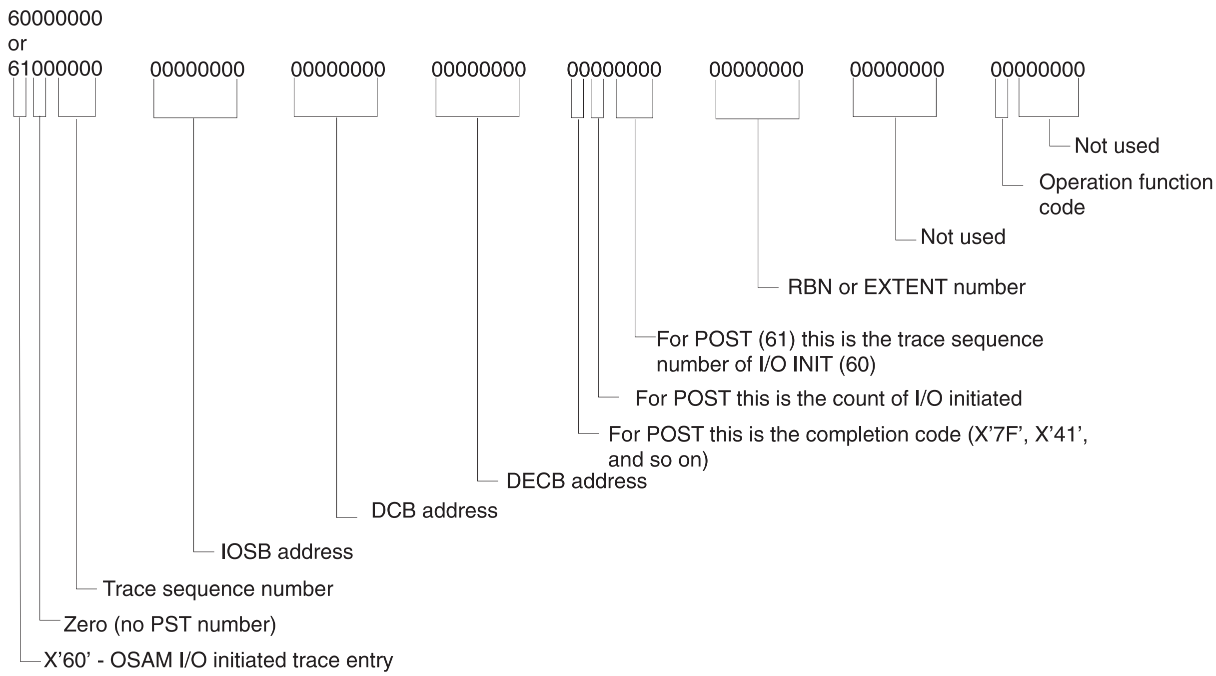


Figure 83. X'60' and X'61' Trace Entries

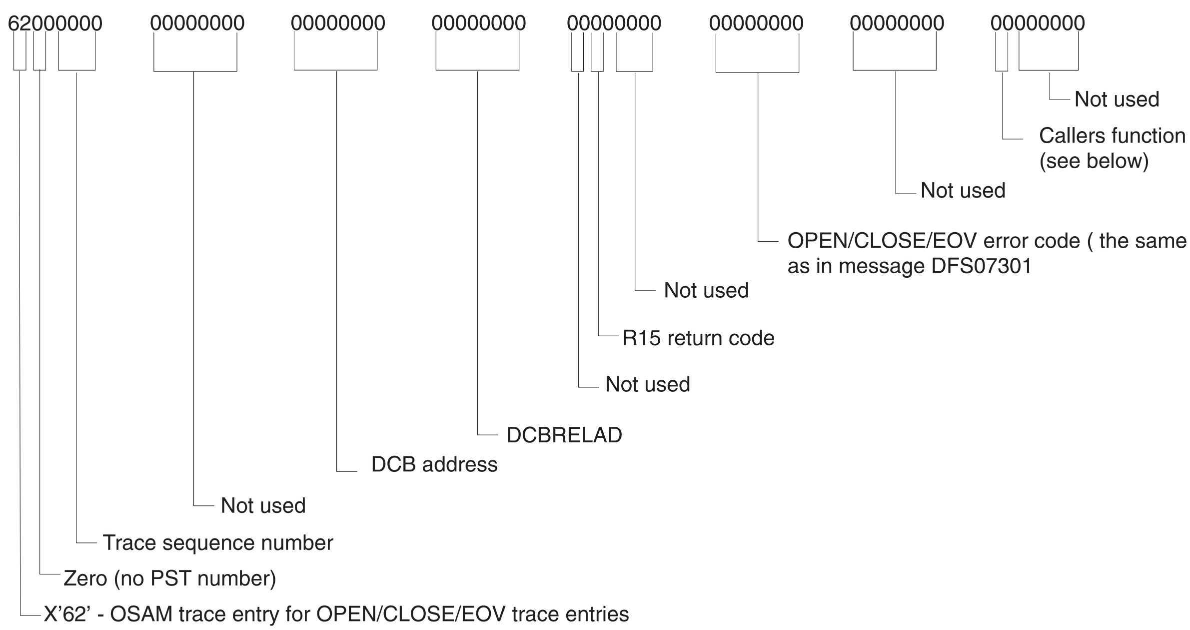


Figure 84. X'62' Trace Entry

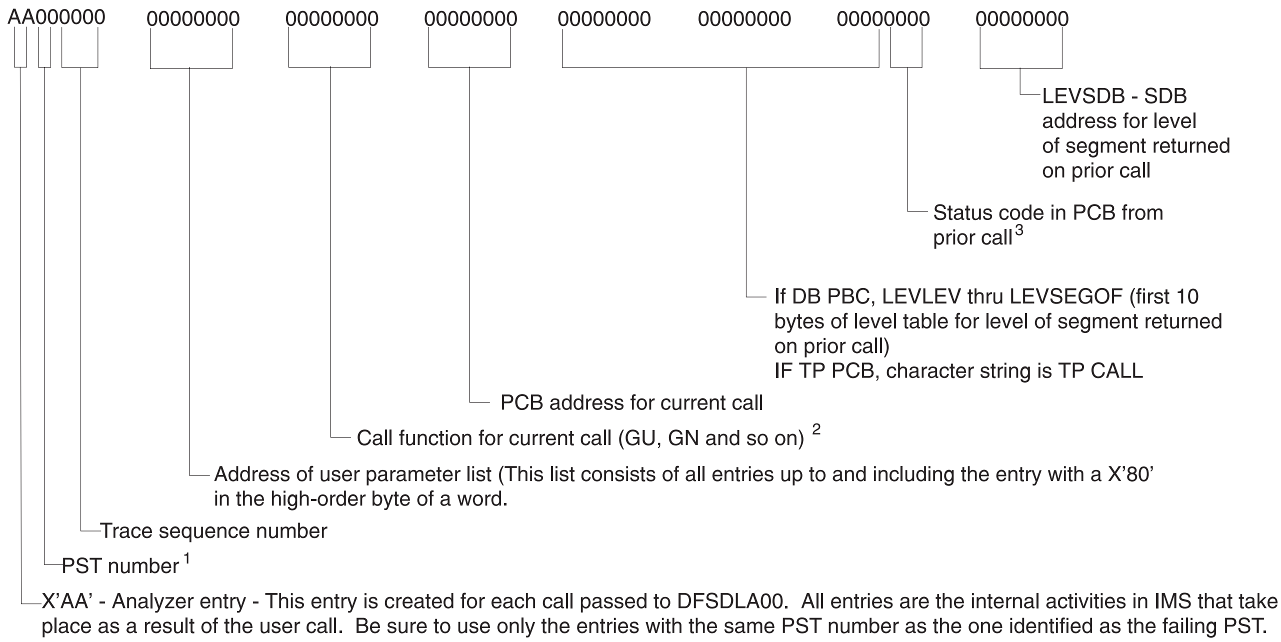


Figure 85. X'AA' Trace Entry

Notes to Figure 85:

1. Use only the trace entries for the PST that had the failure.
2. Determine the current call.
3. Shows how the prior call for this PCB completed.

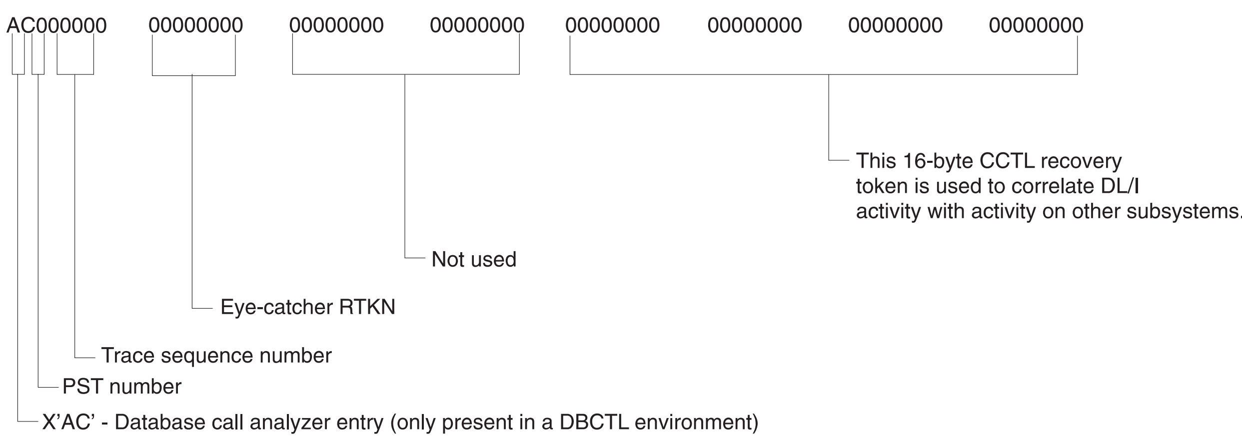


Figure 86. X'AC' Trace Entry

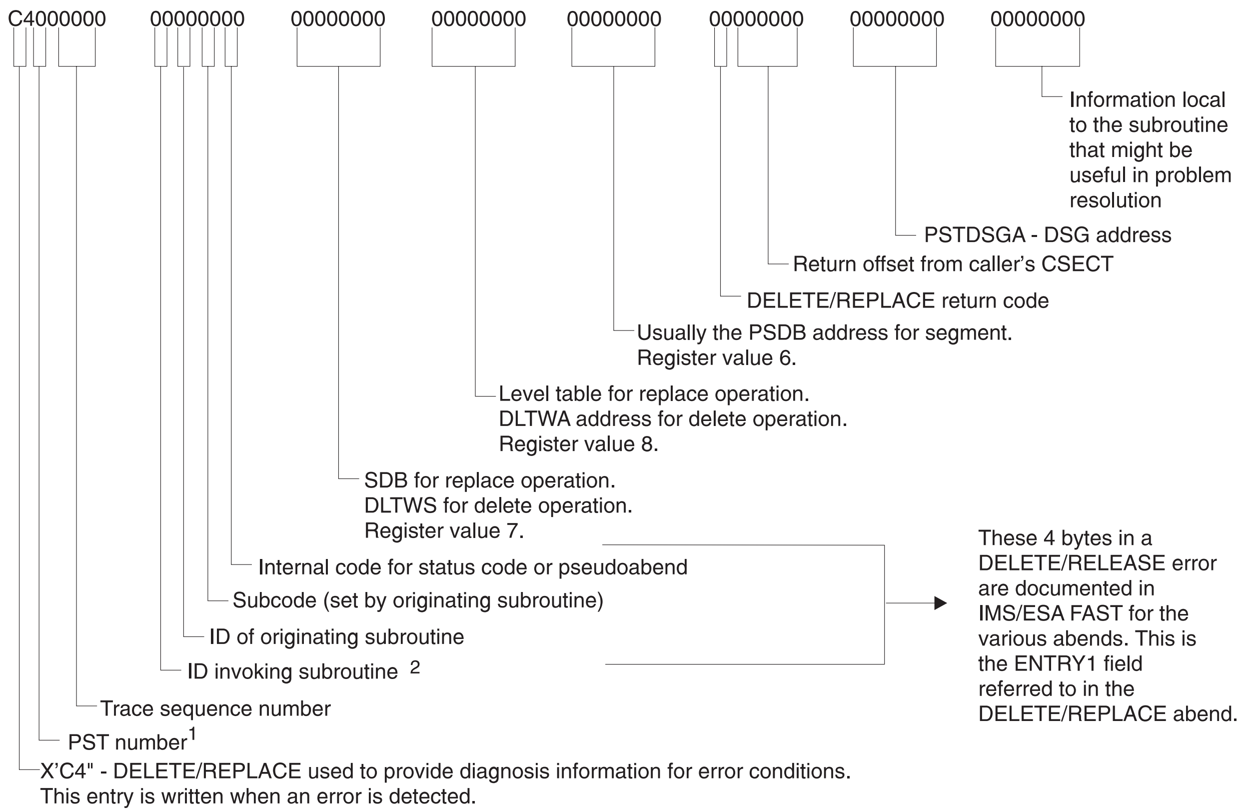


Figure 87. X'C4' Trace Entry

Notes to Figure 87:

1. Use only the entries for the PST that abended.
2. When a DELETE/REPLACE failure occurs, you need the X'C4' entries to solve the problem. You can usually find several X'C4' entries in a row in the trace table. Scan up the trace table to the first (lowest trace sequence number) entry. This entry is usually the key to why the failure occurred. Level 2 needs this information to resolve the problem.

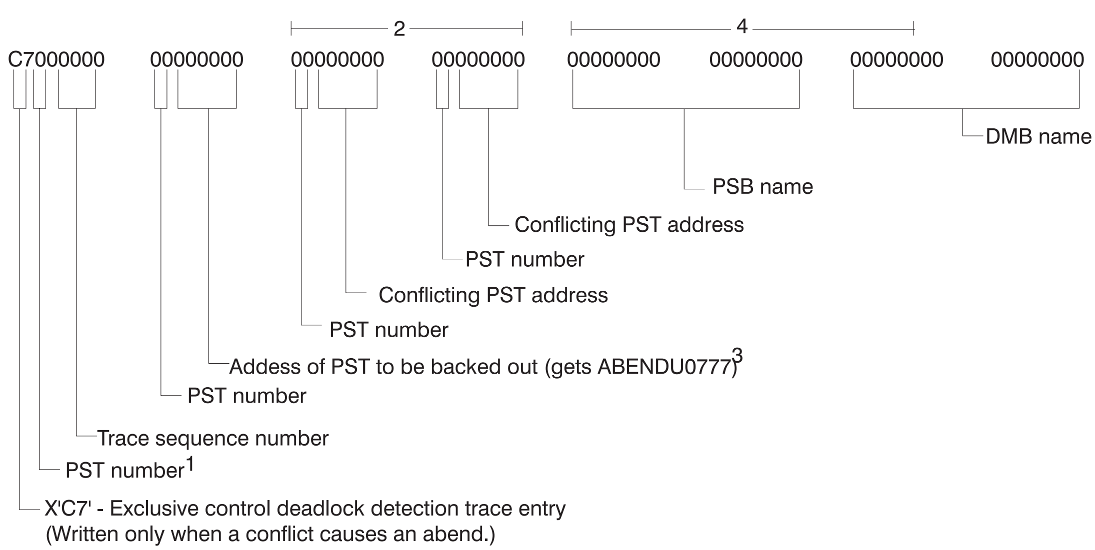


Figure 88. X'C7' Trace Entry (When Not Using the IRLM)

Notes to Figure 88:

1. The entry for the PST number that got the U0777.
2. The addresses of the two conflicting PSTs.
3. The address of the PST that got the U0777.
4. The PSB and DMB name of the cause for the contention.

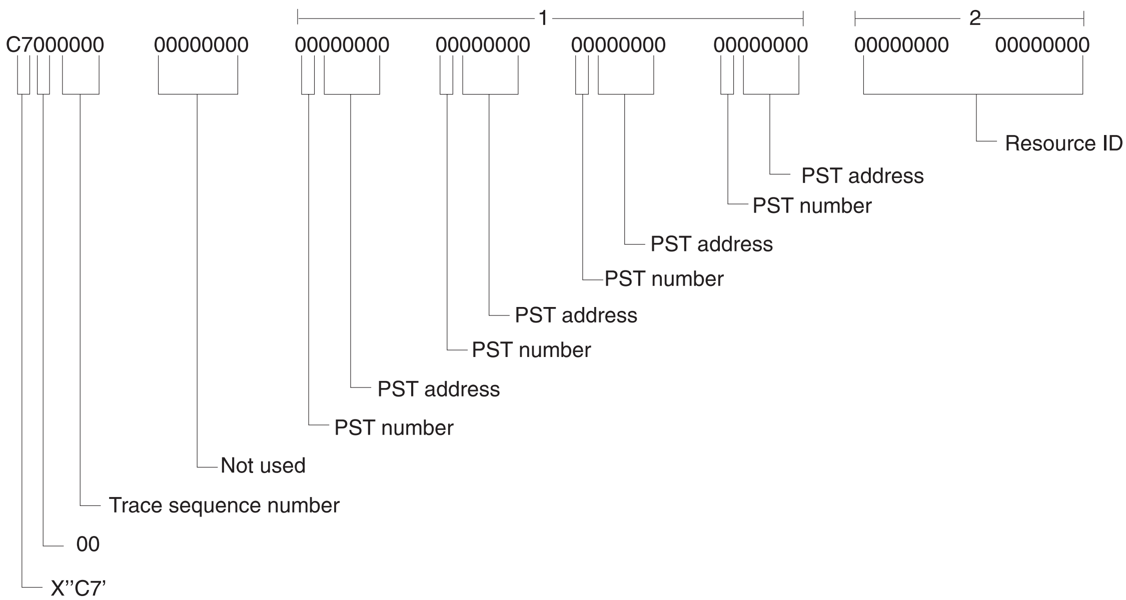


Figure 89. X'C7' Trace Entry (When Using the IRLM)

Notes to Figure 89:

1. PST number and address of PSTs in deadlock net. If number of PSTs in deadlock net is greater than 4, only 4 are shown.
2. Resource ID that is the cause of the deadlock.

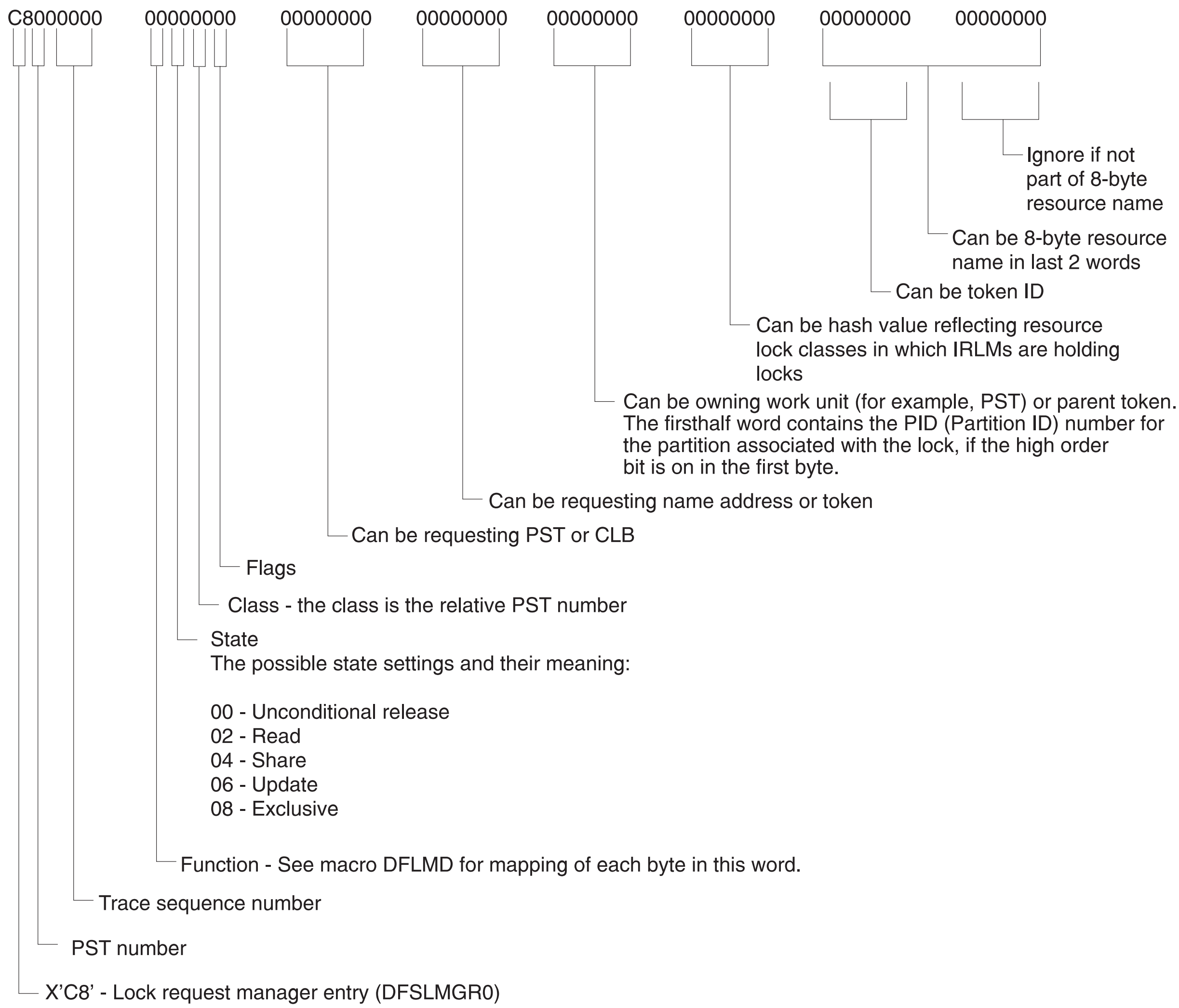


Figure 90. X'C8' Trace Entry

C9000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000

This is a feedback area from the RLPL and is used primarily by the IBM Support Center, if needed.

Lock manager subcode (2 bytes). These bytes along with the return code from IRLM define the problem. (For a description of IRLM error, return, and reason codes, see *IMS/ESA Messages and Codes*.)

Can be resource name address, token, or altered buffer mask

Can be PST, CLB or SRB address

Return code from IRLM

Flags

Class - the class is the relative PST number

State
The possible state settings and their meaning:

00 - Unconditional release
02 - Read
04 - Share
06 - Update
08 - Exclusive

Function - See macro DFLMD for mapping of each byte in this word.

Trace sequence number

PST number

X'C9' - Lock request manager entry (DFSLMGR0) exit

Figure 91. X'C9' Trace Entry

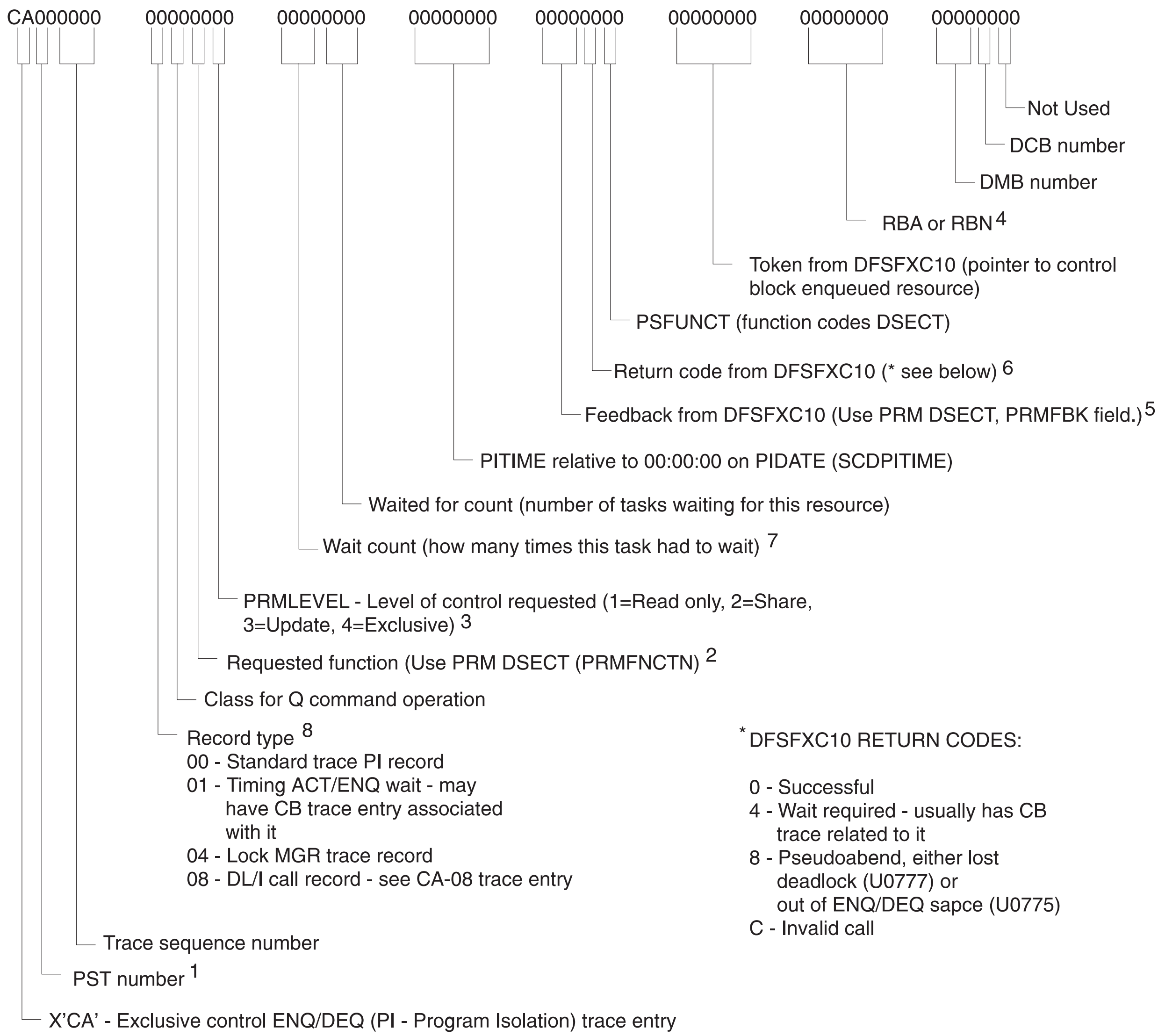


Figure 92. X'CA' Trace Entry

Notes to Figure 92:

1. Use the entries for the PST in question. If you are checking a PI problem, you might have to find this entry and then scan up the trace table using the field in note 4 (below) as a search field to find the other PST that is using the resources.
2. The requested PI function.
3. The level at which the resource was requested.
4. The RBA or RBN of the resource requested by PI (relates to X'04' in the X'CC' trace entry).
5. The 2 bytes of feedback from DFSFXC10 (X'0C' and X'0D' in PRM DSECT).
6. The return code.
7. If a resource (RBA or RBN) is currently owned and the task (PST) must wait, the “wait count” (2 bytes) is incremented in a X'CA' trace entry for the task (PST) that owns the resource. The “waited for count” (2 bytes) is incremented to show that another task is waiting for the resource. This wait should also cause a X'CA', X'CB' pair of trace entries to show the wait occurred. (See the X'CB' trace entry for more details on PI waits.)
8. This shows the type of X'CA' record this is. (X'CA-08' trace entry follows.)

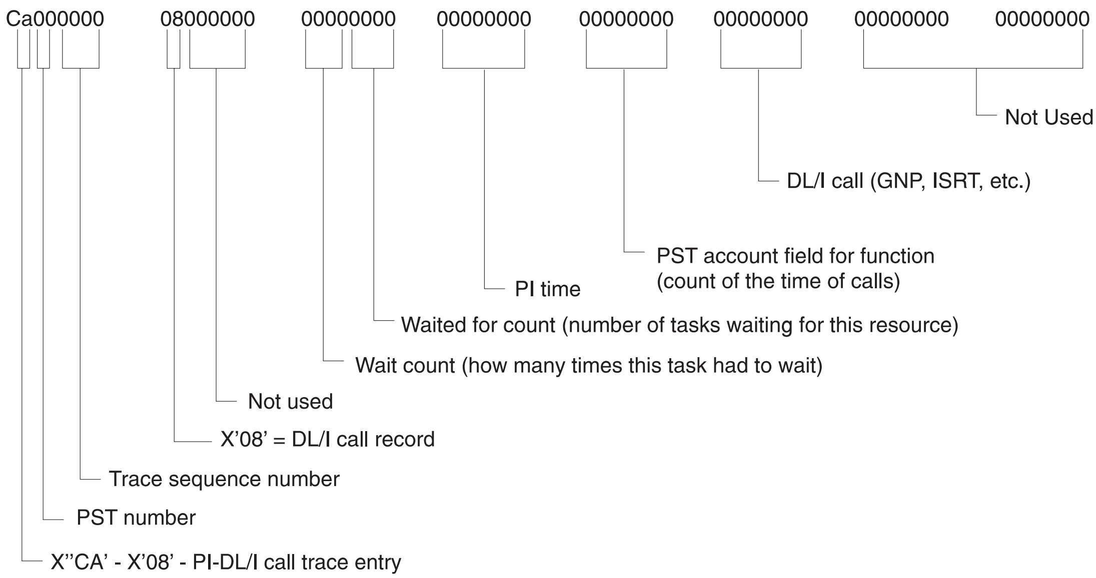


Figure 93. X'CA'—X'08' Trace Entry

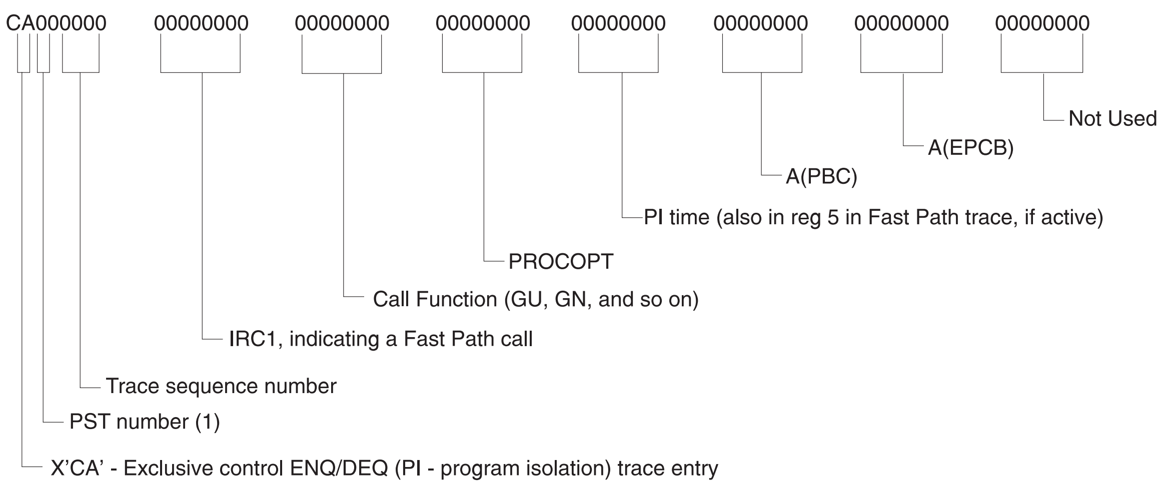


Figure 94. X'CA' Trace Entry for Fast Path Calls

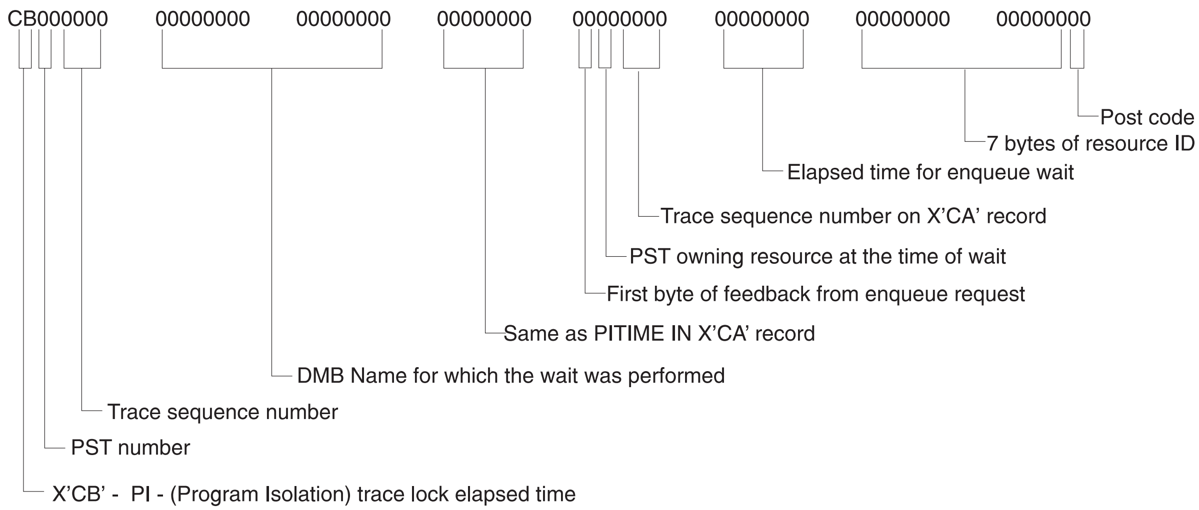


Figure 95. X'CB' Trace Entry

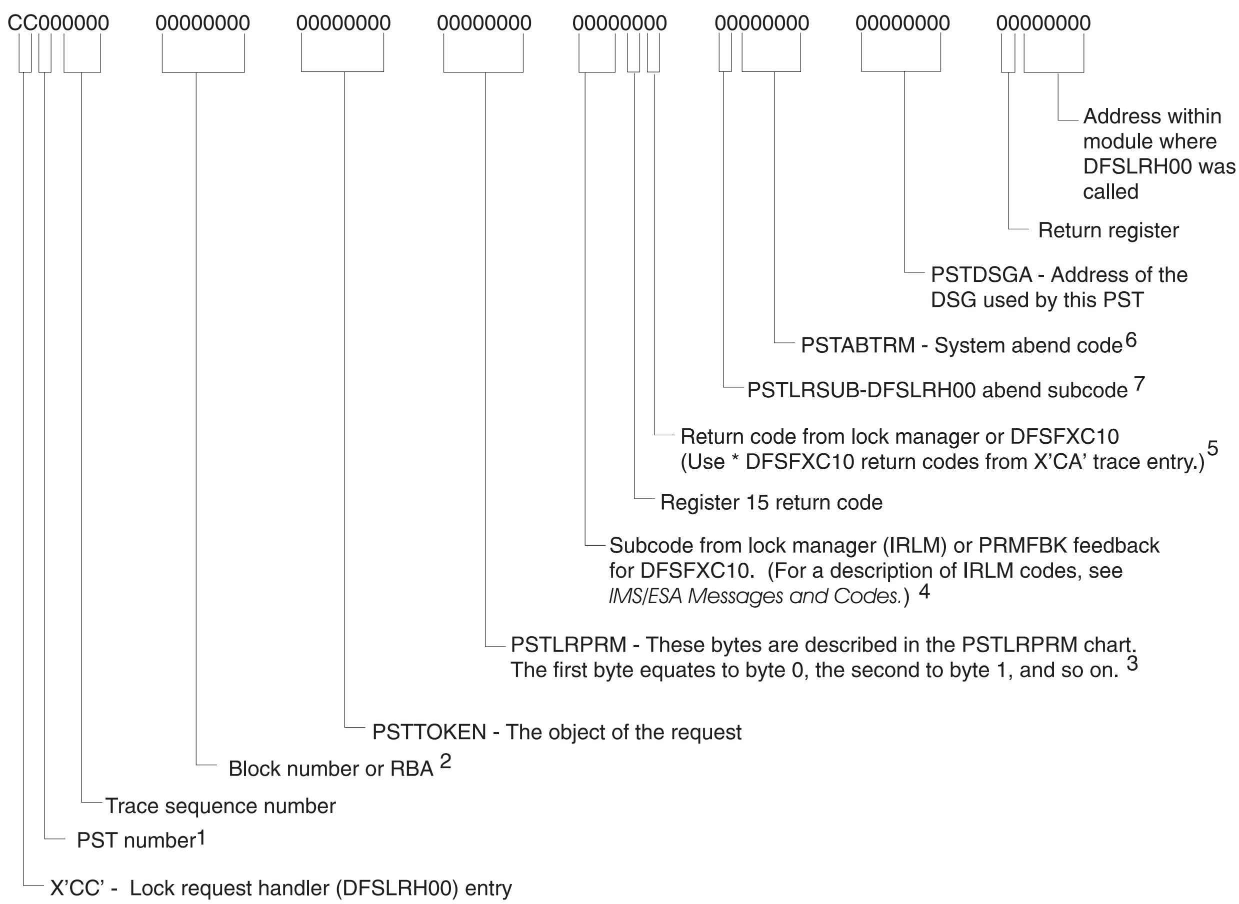


Figure 96. X'CC' Trace Entry

Notes to Figure 96:

1. The PST number for the task (PST).
2. The RBA or RBN of the resource for which a request was issued in a X'CA' trace entry. When some of the problem types occur, you can find the same field or the beginning RBA of the block in the traces for a different PST number.

3. Shows what the request was.
4. For PI, these 2 bytes are in the PRM DSECT at X'0C' and X'0D'.
5. For PI, follow the above. The DFSFXC10 return code is usually also placed in the register 15 return code field.
6. A key field when DFSLRH00 issues an abend (such as U0855, U03301, U03302). The abend is in hexadecimal, not in decimal (for example, 855 = X'0357', 3302 = X'0CE6'). Ignore the field if an abend was not issued from DFSLRH00. For more information about modules issuing abends, find the abend in *IMS Version 7 Failure Analysis Structure Tables (FAST) for Dump Analysis*.
7. For abends issued by DFSLRH00, this field contains the Lock Request Handler abend subcode. For a description of these subcodes, see *IMS Version 7 Failure Analysis Structure Tables (FAST) for Dump Analysis*.

You might need the X'CC' trace entry for several problem types including:
- Task was allowed to process even though a wait was requested.
- DFSLRH00 abends (such as U0855, U03302).
- Request not satisfied. These problems might indicate internal IMS error.

Table 30. PSTLRPRM Chart (Bytes 0 through 3)

Byte 0(Hex)	Meaning
11	Get local segment lock
12	Get local data set busy lock
13	Get local buffer update lock
14	Get local Q command lock
22	Get global buffer update lock
23	Get global data set busy lock
24	Get global data set extend lock
25	Get global data set reference lock
26	Get global command lock
27	Get global command lock (CLB)
30	Get local and global root locks
31	Get local segment and global buffer update locks
32	Get local-global data set busy locks
33	Get local-global buffer update locks
34	Get local Q command and global buffer update locks
41	Release local segment lock
42	Release local data set busy lock
43	Release local buffer update lock
44	Release local Q command lock
52	Release global buffer update lock
53	Release global data set busy lock
54	Release global data set extend lock
55	Release global data set reference lock
56	Release global command lock
57	Release global command lock (CLB)
60	Release local and global root locks
61	Release local and global data set busy locks
62	Release local and global buffer update locks
63	Release local segment and global buffer update locks
70	Test local lock share or update state
71	Test global lock share or update state
72	Test local and global lock share or update
73	Test feedback for local lock
74	Test feedback for global lock

Table 30. PSTLRPRM Chart (Bytes 0 through 3) (continued)

75	Test feedback for local and global locks
80	LRHGIRDX new root, LRHRRIDX old root
81	Release alternate local and global root locks
82	Get local segment and local and global buffer update locks
83	Release all subsystem global busy locks
84	Release all subsystem locks
90	Get Fast Path lock
91	Release Fast Path lock
92	Change ownership of Fast Path lock
93	Force known locks for Fast Path
94	Change locks to retain locks for Fast Path
95	Change ownership of Fast Path UOW lock from release lock ITASK to PST dependent region (HSSP only)
96	Change locks to retain locks for DL/I
97	Invalid call if function is equal to or greater than 97
Byte 1(Hex)	**Meaning**
80	MODE=COND
40	MODE=UNCOND
10	Owning WU given on RRIDX
00	Mode not applicable
Byte 2(Hex)	**Meaning**
01	STATE=READ
02	STATE=SHARE
03	STATE=UPDATE
04	STATE=EXCL
F0	STATE PRESET (Fast Path)
00	STATE not applicable
Byte 3(Hex)	**Meaning**
80	CLB call if LRHPRMFL=X'80'
C0	Fast Path request
68	Root lock request
40	'Single' request
20	'Local' request
10	'Get' request
08	'P-Lock' request
07	'Combined' request if <= X'07'
01	LRHTTLKX, LRHTIBDX
02	LRHGRIDU, LRHRRIDW
03	LRHGSEGX, LRHRSEGX
04	LRHGBIDX, -RBIDX, -GBIDA
05	LRHGZIDX, LRHRZIDX
06	LRHGQCMX
00	LRHRZIDA, LRHRALLX

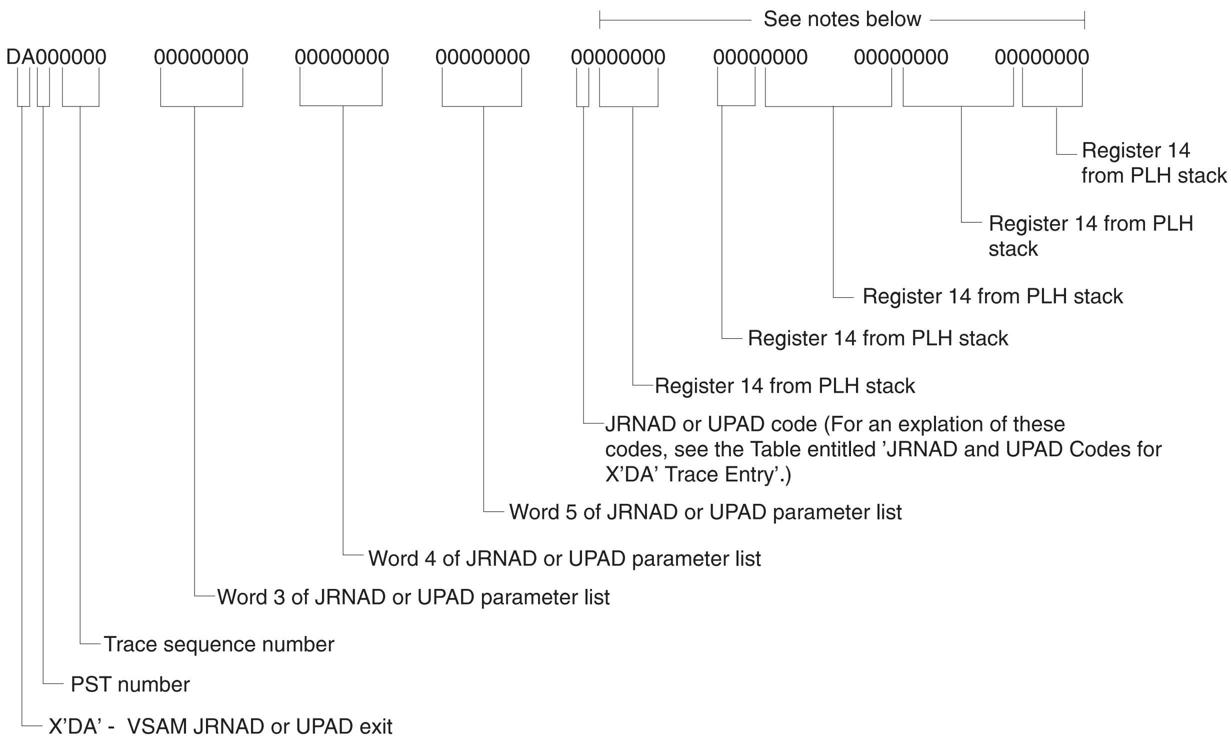


Figure 97. X'DA' Trace Entry

Notes to Figure 97:

1. The PLH stack entries are the registers of the last five VSAM record management modules that had control.
2. This information might be valuable to the VSAM support representatives if you need their assistance.
3. The modules are in LPA and will probably not be in the dump.
4. An AMBLIST of VSAM module IDAO19L1, with OUTPUT=BOTH specified, is needed to determine which CSECTS had control.

Table 31. JRNAD and UPAD Codes for X'DA' Trace Entry

Code	Code (Hex)	Meaning
JRNAD	0C	Logical records to be shifted in a KSDS
JRNAD	10	Cannot occur
JRNAD	14	Cannot occur
JRNAD	20	Control area split starting in a KSDS
JRNAD	24	Control interval read error
JRNAD	28	Control interval write error
JRNAD	2C	Control interval to be written
JRNAD	30	Control interval to be read and marked exclusive
JRNAD	34	Control interval ownership to be established
JRNAD	38	Control interval to be marked exclusive
JRNAD	3C	Create a new control interval
JRNAD	40	Release exclusive use of control interval
JRNAD	44	Mark control interval prefix invalid
JRNAD	48	Control interval read completed
JRNAD	4C	Control interval write completed
JRNAD	50	CI or CA split
UPAD	00	Wait requested on I/O or defer

Table 31. JRNAD and UPAD Codes for X'DA' Trace Entry (continued)

Code	Code (Hex)	Meaning
UPAD	04	Post ECB (XMEM only)

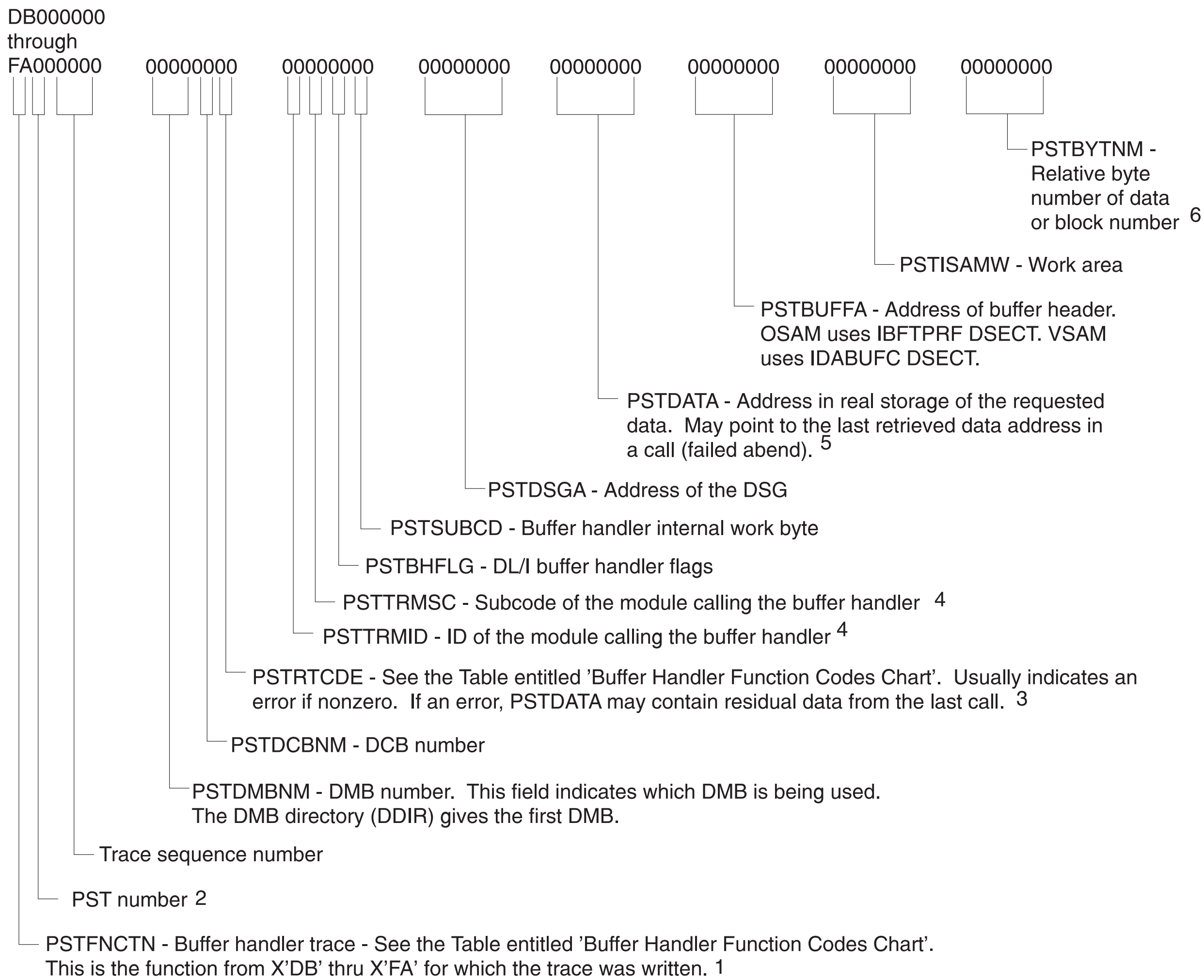


Figure 98. X'DB' through X'FA' Trace Entries

Notes to Figure 98:

1. The IMS internal function that was being performed.
2. Use only the trace entries with the correct PST number.
3. Shows how the call completed. (X'00' means successful completion.)
4. See Table 34 on page 233 for the module names which correspond to the module IDs.
5. Shows where the requested data is located in core only if the call completed successfully.
6. The RBA or block number that the call requested.

   If the call failed, the PSTDATA field might contain the address of the last segment successfully retrieved.

   Example: PSTRTCDE = X'04' (RBA past end of data set).

Buffer Handler Function Codes

PSTFNCTN is located at PST + X'1C4'.

Table 32. Buffer Handler Function Codes Chart

Code (Hex)	PSTFNCTN	Caller's Request Function
DB	PSTSRCHP	Search pool for record in range
DD	PSTRELLR	Release record ownership
DE	PSTRSTAT	Retrieve buffer pool statistics
DF	PSTVERFY	Verify VSAM data set
E0	PSTVPUT	Put record to VSAM data set
E1	PSTBKLCT	Block Locate
E2	PSTBYLCT	Byte Locate
E3	PSTISRCH	Check for duplicate ISAM block
E4	PSTIESDS	Create new ESDS/OSAM LRECL
E5	PSTPGUSR	Write LRECLS for user (PURGE)
E6	PSTBFALT	Mark record altered
E9	PSTFBSPC	Free space in buffer pool (BFPL)
EA	PSTOWTCK	Perform background write function
EB	PSTBYALT	Byte locate and mark altered
EC	PSTBFMPT	Mark buffers empty (BFPL)
ED	PSTCHKPT	Checkpoint
EE	PSTSTAPG	Batch STAE purge at ABEND
EF	PSTERRPG	Purge user for I/O error check
EF	PSTFRWRT	OSAM buffer forced write
F0	PSTSTLBG	Retrieve first LRECL by key
F1	PSTERASE	Erase logical record
F2	PSTSTLEQ	Retrieve by key EQ or GT
F3	PSTSTLCI	Retrieve key EQ or GT - repair CI
F4	PSTSTLIS	Retrieve by key REC to chain from insert logical record (KSDS)
F8	PSTGETNX	Retrieve next SEQ ROOT by key
F9	PSTCPYGU	Position by key for Image Copy
FA	PSTCPYGN	Get next record for Image Copy
	Space Management Function Codes	
31	PSTGTSPC	Get space for the segment
32	PSTFRSPC	Free space for the segment
34	PSTGTRAP	Get space close to root anchor PSTBYTNM. Request to turn off bit map bit. Refer to label PSTBTMPF.
35	PSTGZIDL	Get local serialization as a service to LRH00 during /ERE when IRLM as SLM is not there.
36	PSTRZIDL	Release local serialization
B1	PSTGTSPH	Request for space at BLOCK and OFFS B2-B5 are reserved for tracing PSTDATA. PSTOFFSET must point to the location requested.
	Open/Close Function Codes	
00	PSTOCCLS	This is a close call (bit 4=0)
01	PSTOCDMB	OPEN/CLOSE DMB address of DMB in register 2
02	PSTOCPCB	OPEN/CLOSE PCB address of PCB in register 2
04	PSTOCALL	OPEN/CLOSE all DMBs in the system
08	PSTOCOPN	This is an OPEN call
0C		Combine X'04' and X'08'

10	PSTOCDCB	OPEN/CLOSE DCB PSTDSGA = DSG, PSTDCBNM = DCB NUMBER
20	PSTOCLD	Open for load
21	PSTOCDMA	CLOSE and UNAUTHORIZE DMB address of DDIR in register 2
40	PSTOCDSG	OPEN/CLOSE DSG PSTDSGA = DSG
80	PSTOCBAD	OPEN not successful

Index Maintenance Function Codes		
A0	PSTXMDLT	Index maintenance for segment to be deleted
A1	PSTXMRPL	Index maintenance for segment to be replaced
A2	PSTXMISR	Index maintenance for segment to be inserted
A3	PSTXMUNL	Index maintenance for segment to be unloaded

Block Loader Function Codes		
00	PSTRSVDB	Reserve database resources
01	PSTDMBRD	Read DMB from ACBLIB
02	PSTPSBRD	Read PSB from ACBLIB
03	PSTINTRD	READ INTENT and DMB name lists from ACBLIB
04	PSTENQ	PI Processing is required
40	PSTEREFF	Free DB resources (SCHED failed)
80	PSTFREDB	Free DB resources (termination)

Buffer Handler Return Codes

Table 33. Buffer Handler Return Codes Chart

Return Code		Definition
PSTCLOK	'00'	Everything correct
PSTGTDS	'04'	RBN beyond data set
PSTRDERR	'08'	Permanent read error
PSTNDSPC	'0C'	No more space in data set
PSTBDCAL	'10'	Illegal call
PSTENDDA	'14'	End of data set encountered — no record returned
PSTNDTFD	'18'	Requested record cannot be found
PSTNWBLK	'1C'	New block created in buffer pool
PSTNPLSP	'20'	Insufficient space in pool.
PSTTRMNT	'24'	User must terminate, no space in pool.
PSTDUPLR	'28'	Logical record already in KSDS.

Space Management and Buffer Handler Module Trace IDs

In space management and DL/I buffer handler trace entries, a one-byte module ID identifies the calling module. A one-byte subcode identifies the specific call within the module. The calling module places the module ID in field PSTTRMID and the subcode in field PSTTRMSC before making the call. The buffer handler and space management then move these PST fields to the appropriate traces. Table 34 identifies the calling module.

The PSTTRMSC module subcodes are 0 through 9 and A through Z. If you need to find the point in the module where the call was made, scan for the TIDSCx label that corresponds to the module subcode. Subcode 0 corresponds to label TIDSC0, subcode 1 to label TIDSC1, subcode A to TIDSCA, and so forth.

Table 34. Space Management and Buffer Handler Module Trace IDs

ID Label	Module ID	Calling Module	Module Function
TIDDLA00	A	DFSDLA00	Call analyzer
TIDDLAS0	A	DFSDLAS0	Call analyzer SSA
TIDZDC00	A	DFSZDC00	GSAM Controller
TIDZDI00	B	DFSZDI00	GSAM Initialization
TIDZDI20	C	DFSZDI20	GSAM Initialize GB
TIDDLDC0	D	DFSDLDC0	DELETE/REPLACE
TIDZDI30	D	DFSZDI30	GSAM Buffering Initialization
TIDFLST0	E	DFSFLST0	Batch STAE exit
TIDZD110	E	DFSZD110	GSAM BSAM OPEN / CLOSE
TIDLRH00	F	DFSLRH00	LOCK request handler
TIDZD150	F	DFSZD150	GSAM VSAM OPEN / CLOSE
TIDSDLB0	G	DFSSDLB0	IRLM status routine
TIDZD210	G	DFSZD210	GSAM BSAM I/O
TIDFXC50	H	DFSFXC50	DB SYNC point
TIDZD250	H	DFSZD250	GSAM VSAM I/O
TIDDT400	I	DFSDT400	RSR DB Tracking
TIDZD310	I	DFSZD310	GSAM Buffer I/O
TIDDT500	J	DFSDT500	RSR DB MILESTONE PURGE
TIDDDLE1	K	DFSDDLE0	LOAD INSERT function
TIDZSR00	K	DFSZSR00	GSAM Extended checkpoint
TIDDDLE0	L	DFSDDLE0	LOAD INSERT function
TIDZSR10	L	DFSZSR10	GSAM Restart positioned
TIDPCSH0	M	DFSPCSH0	Partitioning Common Services Handler
TIDDLOC0	O	DFSDLOC0	OPEN/CLOSE
TIDDLOV0	O	DFSDLOC0	LOGICAL/VIRTUAL OPEN
TIDDCAP0	P	DFSDCAP0	Full-Function Data capture
TIDDDUI0	Q	DFSDDUI0	DUI processor
TIDDLR00	R	DFSDLR00	RETRIEVE function
TIDDHD00	S	DFSDHD00	Space Manager (INIT procedure)
TIDFRSP0	S	DFSFRSP0	Space Manager (free space)
TIDGGSP0	S	DFSGGSP0	Space Manager (GET space)
TIDMMUD0	S	DFSMMUD0	Space Manager (bit map update)
TIDRCHB0	S	DFSRCHB0	Space Manager (SEARCH block)
TIDRRHM0	S	DFSRRHM0	Space Manager (SEARCH bit map)
TIDRRHP0	S	DFSRRHP0	Space Manager (buffer pool)
TIDTOBH0	T	DFSTOBH0	I/O toleration buffer handler caller
TIDTOCL0	T	DFSTOCL0	I/O toleration DB close
TIDDPSB0	U	DFSDPSB0	PSB generator utility
TIDURDB0	U	DFSURDB0	DB Data Set Recovery utility
TIDURGP0	U	DFSURGP0	REORG/RELOAD, PREFIX update utility
TIDURGS0	U	DFSURGS0	REORG/LOAD, DB scan utility
TIDBACK0	V	DFSBACK0	BATCH backout utility
TIDURRL0	V	DFSURRL0	HISAM REORG/RELOAD utility
TIDURUL0	V	DFSURUL0	HISAM REORG/UNLOAD utility
TIDUCPD0	W	DFSUCPD0	UCF DB ZAP processor utility
TIDUCPE0	W	DFSUCPE0	UCF subroutines utility
TIDUICC0	W	DFSUICC0	Online Image Copy utility
TIDDXMT0	X	DFSDXMT0	Index maintenance
TIDRBOI0	Y	DFSRBOI0	Backout RESTART/DYN/BATCH
TIDRDBC0	Z	DFSRDBC0	Database backout control

Figure 99 shows an example of a DL/I trace. The trace entries show two GHU calls. All calls use PST 01. When activities for different PSTs are intermixed in the trace table, you need to examine only the entries for the PST of interest.

```
 FUNCTION         WORD 0    WORD 1    WORD 2    WORD 3    WORD 4    WORD 5    WORD 6    WORD 7                            PAGE 0001
* DL1 TRACE TABLE - DATE 89039 TIME 17450600 SKIP 0000 TOTAL SKIP 00000000 RECORD NUMBER 00000007
ANALYZE CALL     AA01008A  00008DE0  GHU         0A0D60  03080800  00004892  00004000  0008F200   ........GHU ...-.......K.. ...2.
VSAM EXIT        DA01008B  0272FA60  06000000  00002400  34B95982  B96E24B9  BCE6BA6E  50B9AE68   .......-...........B.>...W.>&...
PSTBYLCT         E201008C  00040100  D2014400  000A101C  0273720C  0272FA60  0274E45E  0000260C   S.......K.............-..U;....
VSAM EXIT        DA01008D  0272FAB0  06000000  00004800  34B95982  B96E24B9  BCE6BA6E  50B9AE68   ...................B.>...W.>&...
PSTBYLCT         E201008E  00030100  D2014400  000A205C  02739092  0272FAB0  0274E45E  00004892   S.......K......*...K......U;...K
VSAM EXIT        DA01008F  0272FB50  06000000  00002400  34B95982  B96E24B9  BCE6BA6E  50B9AE68   .......&...........B.>...W.>&...
PSTBYLCT         E2010090  00030100  D2014400  000A205C  0273D354  0272FB50  0274E45E  00002754   S.......K......*..L....&..U;....
PSTBYLCT         E2010091  00030100  D2014400  000A205C  0273D11C  0272FB50  0274E45E  0000251C   S..J....K......*..J....&..U;....
PSTBYLCT         E2010092  00030100  D2014400  000A205C  0273D354  0272FB50  0274E45E  00002754   S..K....K......*..L....&..U;....
PSTBYLCT         E2010093  00030100  D2014400  000A205C  0273D11C  0272FB50  0274E45E  0000251C   S..L....K......*..J....&..U;....
PSTBYLCT         E2010094  00030100  D2014400  000A205C  0273D020  0272FB50  0274E45E  00002420   S..M....K......*.......&..U;....
VSAM EXIT        DA010095  0272FAB0  06000000  00004800  34B95982  B96E24B9  BCE6BA6E  50B9AE68   ...N...............B.>...W.>&...
PSTBYLCT         E2010096  00030100  D2014400  000A205C  02739092  0272FAB0  0274E45E  00004892   S..O....K......*...K......U;...K
VSAM EXIT        DA010097  0272FB50  06000000  00002400  34B95982  B96E24B9  BCE6BA6E  50B9AE68   ...P...&...........B.>...W.>&...
PSTBYLCT         E2010098  00030100  D2014400  000A205C  0273D354  0272FB50  0274E45E  00002754   S..Q....K......*..L....&..U;....
ANALYZE CALL     AA010099  00008DE0  GHU         0A0D60  03280800  00004892  00004000  0008F200   ...R....GHU ...-.......K.. ...2.
 FUNCTION         WORD 0    WORD 1    WORD 2    WORD 3    WORD 4    WORD 5    WORD 6    WORD 7                            PAGE 0004
VSAM EXIT        DA01009A  0272FA60  06000000  00002400  34B95982  B96E24B9  BCE6BA6E  50B9AE68   .......-...........B.>...W.>&...
PSTBYLCT         E201009B  00040100  D2014400  000A101C  0273720C  0272FA60  0274E45E  0000260C   S.......K.............-..U;....
VSAM EXIT        DA01009C  0272FAB0  06000000  00004800  34B95982  B96E24B9  BCE6BA6E  50B9AE68   ...................B.>...W.>&...
PSTBYLCT         E201009D  00030100  D2014400  000A205C  02739092  0272FAB0  0274E45E  00004892   S.......K......*...K......U;...K
VSAM EXIT        DA01009E  0272FB50  06000000  00002400  34B95982  B96E24B9  BCE6BA6E  50B9AE68   .......&...........B.>...W.>&...
PSTBYLCT         E201009F  00030100  D2014400  000A205C  0273D354  0272FB50  0274E45E  00002754   S.......K......*..L....&..U;....
PSTBYLCT         E20100A0  00030100  D2014400  000A205C  0273D11C  0272FB50  0274E45E  0000251C   S.......K......*..J....&..U;....
PSTBYLCT         E20100A1  00030100  D2014400  000A205C  0273D354  0272FB50  0274E45E  00002754   S.......K......*..L....&..U;....
PSTBYLCT         E20100A2  00030100  D2014400  000A205C  0273D11C  0272FB50  0274E45E  0000251C   S..S....K......*..J....&..U;....
PSTBYLCT         E20100A3  00030100  D2014400  000A205C  0273D020  0272FB50  0274E45E  00002420   S..T....K......*.......&..U;....
VSAM EXIT        DA0100A4  0272FAB0  06000000  00004800  34B95982  B96E24B9  BCE6BA6E  50B9AE68   ...U...............B.>...W.>&...
PSTBYLCT         E20100A5  00030100  D2014400  000A205C  02739092  0272FAB0  0274E45E  00004892   S..V....K......*...K......U;...K
VSAM EXIT        DA0100A6  0272FB50  06000000  00002400  34B95982  B96E24B9  BCE6BA6E  50B9AE68   ...W...&...........B.>...W.>&...
PSTBYLCT         E20100A7  00030100  D2014400  000A205C  0273D354  0272FB50  0274E45E  00002754   S..X....K......*..L....&..U;....
```

Figure 99. Example of a DL/I Trace

DELETE/REPLACE—DL/I Trace Information

The DELETE/REPLACE module provides meaningful information when abnormal conditions arise leading directly to errors detected by Delete/Replace. This information can be found in the Delete/Replace work area (DLTWA).

Abends initiated by the Delete/Replace module (780, 796, 797, 798, 799, 802, 803, 804, 806, 807, 808, and 811) are traced in the DL/I trace table in a series of entries identified by an X'C4' in the first byte (TRACE FUNCTION CODE).

The first X'C4' entry in the series is provided by the routine that encountered the problem. Each additional entry is provided by the routine that called the routine which in turn wrote the prior entry in the table. Examining these entries in reverse sequence reveals the order in which control was passed from one routine to another.

A complete description of the trace table entry for Delete/Replace can be obtained by assembling:

```
DSECTS  CSECT
        DFSDLDC FUNC=DSFCTS
        END
```

Of great value in the Delete/Replace trace entry is the second word (called Entry1). This word uniquely identifies a Delete/Replace abend, and should be used by IBM and customers when submitting APARs for better problem description. In some cases, the Entry1 word from the next trace entry along with the first Entry1 word uniquely identifies the abend. The Entry1 format is:

```
BYTE 0    ID of routine supplying this entry
     1    ID of routine that encountered error
     2    Subcode number of abend if multiples
     3    Internal code for abend
```

Each routine within the Delete/Replace module has a unique one-byte identification number. The IDs can be obtained from the assembly listings of each of the four source modules which make up the Delete/Replace call. In general they are:

```
X'01'  to  X'1F'—control and common subroutines (DFSDLDC0)
X'20'  to  X'3F'—delete routines (DFSDLDD0)
X'40'  to  X'5F'—replace routines (DFSDLDR0)
X'60'  to  X'7F'—DLTWA build routines (DFSDLDW0)
```

Use the Entry1 word (the second word in the trace entry) when relating to a Delete/Replace problem in IMS with the IBM Support Center.

Retrieve Trace

When an application program executes and a problem occurs (such as damaged data or unexpected results), you can use the retrieve trace records to see how IMS responded to various calls in the application.

To turn on the retrieve trace, use either of these methods:

- At initialization time, specify DL/I=ON on the OPTIONS statement in member DFSVSMxx (for DB/DC) or DFSVSAMP (for batch) of the IMS.PROCLIB data set. The retrieve trace is turned on automatically. (See *IMS Version 7 Installation Volume 2: System Definition and Tailoring*.)
- For DB/DC and DBCTL environments, Use the `/TRACE SET ON TABLE RETR` command. If you start the DL/I trace by using the `/TRACE SET ON TABLE DLI` command, the retrieve trace is not automatically turned on. (See *IMS Version 7 Command Reference*.)

To quickly determine if the trace is in the dump, check field PSTDLR1 in the PST.

X'0700' Indicates the trace is on.

X'07FC' Indicates the trace is off.

Field PSTRTVTR of the PST contains the address of the trace table. (See Figure 80 on page 214.) The byte at PSTRTNDX contains the offset to the next entry in the table. (See Figure 100 on page 238.)

Every time an application issues a get or insert call, the retrieve module (DFSDLR00) is called. This module is very large and contains many subroutines. By looking at the retrieve trace, you can see the flow of control through the various subroutines of the retrieve module. As each subroutine calls another, a 2-byte hexadecimal entry is inserted into the trace table. (Byte 1 of the trace entry is the ID of the calling subroutine; byte 2 is the ID of the subroutine that is called.) Table 35 on page 236 lists the IDs, names, and functions of the various subroutines.

The retrieve trace table is filled from beginning to end. When the table becomes full, tracing starts at the beginning of the table, overlaying each old entry with the new entry.

The first entry in the trace table for a call is X'F1', which is paired with entries: X'2F' (UNQL), X'30' (ROOTISRT), or X'31' (QUAL). The presence of any of these entries indicates the beginning of a trace entry for a retrieve call. For an example of the retrieve trace, see Figure 100 on page 238.

Field JCBRTVTR in the JCB also contains retrieve trace information. JDBRTVTR contains the offsets to the initial entries in the trace table for the previous four DL/I calls that are associated with a database. The offset to the last call is in the low-order byte, and all offsets are shifted left at the start of each new call.

Example: The execution of an application results in an error message that indicates damaged data. You can refer to the retrieve trace table and interpret the entries in order to determine if the problem is caused by:

- An application error

- A database design error
- An internal IMS DB problem
- An IMS system problem related to pointers

If you determine that the problem was caused by an application or database design error, you can use the retrieve trace to debug and resolve the problem. Otherwise, you can do a keyword search. If the search results in a large number of problems, you can reduce the number of problems by including the name of the subroutine (listed in Table 35), which you found in the retrieve trace table.

Table 35. The Subroutines of the Retrieve Module (DFSDLR00)

Hex ID	Subroutine Title	Subroutine Description
01	BLDVKEY	Builds alternate parent's concatenated key in work area.
02	CSIIGEXT	Reads root based on SSA qualification. If found, GE at level one. If not found, GE at level 0.
03	DIVRSETU	Position (DIV) was not found at this level. Sets off EOC and sets on not posted first child and siblings.
04	ENQDQ	Handles all enqueue and dequeue for retrieve.
05	FNDLPNQ	Final physical root of LP SDB and enqueue it.
06	FORTHISL	Tries to get a segment that satisfies the call at this level or higher.
07	GEEXIT	Publishes GE status code or GB (if root SDBEOC on).
08	GETPSDB	Gets the PSDB of the segment pointed to by JCBACSC.
09	GETPRIME	Issues request for SETL to retrieve next higher root in database.
0A	STLALTPS	Processes request for data by key when an alternate processing sequence is used.
0B	ISRTMPOS	While positioning for insert, a matching segment was found; checks if permissible.
0C	ISRTPOS	Checks for LC insert to locate alternate parent, validate insert, or establish position on alternate twin chain.
0D	ISRTVER	Verifies segment in POSP points to segment in SDBPOSN for HDAM and HIDAM organizations.
0E	KDTEST	Compares value in SSA to value in segment or to key feedback for requalification.
0F	LCPTRTST	Used by CC=L processing to use PCL pointer, if any.
10	LTW	Main driver for requalification to determine the acceptability of current position.
11	LTWLRTN	Used by CC=L processing to see if on last or should use PCL pointer or continue trying (HS).
12	LTWLTST	Used by CC=L processing to find the last segment.
13	MOVEKEY	Moves key from segment to PCB key feedback.
14	MVSEGUSE	Moves the requested segment from the I/O area to the user area.
15	POSTCHLD	Captures child RBNs from input SDB prefix and places in SDBPOSN of dependent SDBs.
16	POSTME	Places search starting position for segment in SDB.
17	POSTTRY	Unqualified GN has found a segment. Posts the position and key.
18	POSTCURP	Moves position from JCB work words into SDB and sets post code.
19	POSTSDBN	Stores location of next segment on chain in JCB work words.
1A	READCUR	Locates current entry in passes SDB.

Table 35. The Subroutines of the Retrieve Module (DFSDLR00) (continued)

Hex ID	Subroutine Title	Subroutine Description
1B	RDLPCONK	Locates logical parent via its key.
1C	READNXT	Locates next segment from passes SDB.
1D	RDPHYPR	Locates physical pair of segments when passed SDB address of its pair.
1E	RESETMP	Initializes for unqualified call.
1F	RESETQMP	Compares previous call position in level table to current qualification where POS=M.
20	SCDCRSCK	Not first LR crossed and concatenated segment ISRT, builds concatenated key of LC physical parent.
21	SETEOC	Sets EOC in requested SDB. If logical parent enqueues outstanding, locates each and dequeues.
22	SETL	Provides interface to buffer handler for all external data requests.
23	SETLBG	Issues request for SETL to get first root in database.
24	SETPVEOC	Sets EOC on previous SDBs in the hierarchy having the same parent as the passed SDB.
25	SSAEVAL	Examines a segment to see if it satisfies the qualification.
26	SETCHEOC	Sets on SDBEOC of dependent SDBs.
27	STECHISB	Sets SDBEOC on for input SDB and siblings having same physical parent.
28	SETLMIKY	SETL to find key equal to or greater than key determined as minimum value for SSA.
29	STNPHISB	Sets EOC (if in use) and not posted for siblings of input SDB.
2A	THISLVOK	Found one at this level that satisfies the call. Uses it and checks for more levels in call.
2B	UNQGN	Gets next sensitive segment without violating parentage.
2C	VLEXP	Processes variable length segment and user data compaction.
2D	WIPEDN	Clears level table below level passed to bottom of table or below entry currently cleared.
2E	XDFTEST	Qualification is secondary index. Checks index entries to validate the position.
2F	UNQL	Master driver for calls without SSAs.
30	ROOTISRT	Routine for positioning to insert at physical root of database.
31	QUAL	Driver for qualified retrievals.
32	HSAMRTN	HSAM I/O interface routine.
33	RETRY	Retry routine for processing option GOT.
34	ISRTCHCK	Use two keys in DSG for root insert.
35	VALIDATE	Validate an EPS.
36	PARTCKRC	Check results of the validate.
F1	INIT	Initialization.

```
                    F O R M A T T E D   R E T R I E V E   T R A C E

         OFFSET  ---FROM---      ---TO ---          OFFSET  ---FROM---      ---TO ---

PSTRTVTR ---> 00  2A THISLVOK    15 POSTCHLD            56  16 POSTME      1A READCUR
              02  2A THISLVOK    13 MOVEKEY             58  1A READCUR     22 SETL
              04  31 QUAL        06 FORTHISL            5A  1C READNXT     1A READCUR
              06  06 FORTHISL    24 SETPVEOC            5C  1A READCUR     22 SETL
              08  06 FORTHISL    1C READNXT             5E  1E READNXT     19 POSTSDBN
              0A  1C READNXT     26 SETCHEOC            60  06 FORTHISL    25 SSAEVAL
              0C  1C READNXT     1A READCUR             62  06 FORTHISL    18 POSTCURP
              0E  06 FORTHISL    24 SETPVEOC            64  31 QUAL        2A THISLVOK
              10  24 SETPVEOC    21 SETEOC              66  2A THISLVOK    15 POSTCHLD
              12  24 SETPVEOC    21 SETEOC              68  2A THISLVOK    13 MOVEKEY
              14  24 SETPVEOC    21 SETEOC              6A  2A THISLVOK    14 MVSEGUSE
              16  06 FORTHISL    1C READNXT          -> 6C  06 FORTHISL    24 SFTPVEOC
              18  1C READNXT     26 SETCHEOC            6E  24 SETPVEOC    21 SETEOC
              1A  1C READNXT     1A READCUR             70  06 FORTHISL    1C READNXT
              1C  06 FORTHISL    02 CSILGEXT            72  1C READNXT     26 SETCHEOC
              1E  F1 INIT        31 QUAL                74  1C READNXT     1A READCUR
              20  31 QUAL        10 LTW                 76  1A READCUR     22 SETL
              22  10 LTW         25 SSAEVAL             78  1C READNXT     19 POSTSDBN
              24  25 SSAEVAL     0E KDTEST              7A  06 FORTHISL    25 SSAEVAL
              26  10 LTW         14 MVSEGUSE            7C  25 SSAEVAL     0F KDTEST
              28  14 MVSEGUSE    1A READCUR             7F  06 FORTHISL    18 POSTCURP
              2A  1A READCUR     22 SETL                80  31 QUAL        2A THISLVOK
              2C  10 LTW         2D WIPEDN              82  2A THISLVOK    15 POSTCHLD
              2F  31 QUAL        06 FORTHISL            84  2A THISLVOK    13 MOVEKEY
              30  06 FORTHISL    24 SETPVECC.           86  2A THISLVOK    14 MVSEGUSE
              32  24 SETPVEOC.   21 SETEOC.             88  F1 INIT        31 QUAL
              34  24 SETPVEOC    21 SETEOC              8A  31 QUAL        10 LTW
              36  24 SETPVEOC    21 SETEOC              8C  10 LTW         26 SETCHEOC
              38  06 FORTHISL    1C READNXT             8F  10 LTW         2D WIPFDN
              3A  06 FORTHISL    02 CSIIGEXT            90  10 LTW         1A READCUR
   Call start 3C  F1 INIT        31 QUAL                92  1A READCUR     22 SFTL
              3E  31 QUAL        10 LTW                 94  10 LTW         19 POSTSDBN
              40  10 LTW         14 MVSEGUSE            96  31 QUAL        18 POSTCURP
              42  14 MVSEGUSE    1A READCUR             98  31 QUAL        2A THISLVOK
              44  1A READCUR     22 SETL                9A  2A THISLVOK    15 POSTCHLD
              46  1C LTW         2D WIPEDN              9C  2A THISLVOK    13 MOVEKEY
              48  31 QUAL        06 FORTHISL            9F  2A THISLVOK    14 MVSEGUSE
              4A  06 FORTHISL    24 SETPVEOC            A0  F1 INIT        31 QUAL
              4C  24 SETPVEOC    21 SETEOC              A2  31 QUAL        10 LTW
              4E  24 SETPVEOC    21 SETEOC              A4  10 LTW         26 SETCHEOC
              50  24 SETPVEOC    21 SETEOC              A6  10 LTW         1A READCUR
              52  06 FORTHISL    1C READNXT             A8  1A READCUR     22 SFTL
              54  1C READNXT     16 POSTME              AA  10 LTW         19 POSTSDBN

                                                                 PSTRTNXD 6C
                  JCB                             JCBRTVTR

                           0019CF68 00190FF0  0019F40C 0019F694 A0CF1F3C 21400140
         214001C5 03C50140 00190F90 05900010  00000080 00000000 00004000 70016000
         001652A8 0019CF68 001984C0 00000000  00000000 00000000 000017C4 00000000
         00000000 00001704 0019A360 0019A36A  0019CF18 0519EA04 0019F210 00000000
         00000004 00D44001 00280000 00000101  00000C00 012C7FFF 00000000 00000000
         0019F520 0019E52C 0019E4F4 00000000  00000000 00000000 00000000 0019CD60
         00000000 00000000 00000000 00000000  04000000 0019CF90 000027C4 00000000
```

Figure 100. Example of a Retrieve Trace

Program Isolation-Related Problem Analysis

When invalid segment data is retrieved, or an unexpected user abend occurs during concurrent updates to a single database by more than one processing region under the protection of program isolation, improper enqueue or dequeue logic has been followed in IMS. Tools are available to properly document this occurrence. Correct and adequate documentation might depend on the ability to reproduce the error condition and on the availability of the IBM Support Center.

Limiting Locking Resources Used by an Application Program

In order to avoid resource problems that can be caused by runaway applications, you can limit the number of locks an application can have by using the LOCKMAX parameter.

The LOCKMAX Parameter

The LOCKMAX parameter can be specified on the PSBGEN statement or at execution time. The parameter has the following format: LOCKMAX=*n* where *n* is a number between zero and 255. Zero is the default and implies no maximum lock limit.

The number specified indicates units of 1000; for example, a specification of LOCKMAX=*5* means that the application cannot have more than 5000 locks at one time.

Restriction: While the LOCKMAX parameter allows you to limit the amount of resources used by an application, it cannot be used to initially specify the amount of resources to be used by an application. Use traditional methods for specifying these resources through the PSB.

Choosing a Value for LOCKMAX

To decide what value to use for LOCKMAX, analyze over a period of time the X'37', X'41', and X'5937' commit log records to determine the maximum number of locks being held per unit of work by the application. Each of these log records contains a ″high water lock count″ or maximum lock count, which is the maximum number of locks held by the application. The X'41' log record shows a zero for the number of locks held, except in DL/I and DBB Batch cases involved in block-level data sharing.

For a more complete description of the X'37' and X'41' log records, see Table 6 on page 113.

Exceeding the LOCKMAX Value

When the value specified for LOCKMAX is exceeded by an application, a pseudoabend of type U3301 results. Modules DFSLRHOO and DBFLRHOO set this pseudoabend when the return codes and feedback from either PI or IRLM indicate that the lock request failed because granting the lock would exceed the LOCKMAX value.

For more information about the LOCKMAX parameter and its uses, see *IMS Version 7 Administration Guide: System*.

Program Isolation (PI) Trace

One tool is the program isolation (PI) trace. It traces all calls to the IMS enqueue/dequeue module (DFSFXC10) and writes the trace entries to the system log as type X'67FA' records.

Entries with IDs X'C7', X'C8', X'C9', X'CA', X'CB', and X'CC' are PI entries. For the layout of these trace records, see "DL/I Trace Formats" on page 215.

In a DB/DC environment, you start the trace by entering the /TRACE command at the master terminal operator's console. For batch or DB/DC environments, you specify LOCK=OUT on the OPTIONS statement at system initialization time.

Save the log tape and submit it as APAR documentation. If you cannot ship the log tape with the APAR, you can use the File Select and Formatting Print utility (DFSERA10) with exit DFSERA40 to select and format records related to the problem from the log tape. See *IMS Version 7 Utilities Reference: Database and Transaction Manager* for a description of the File Select and Formatting Print utility.

| "Format of X'67' Log Record" on page 125 shows the layout of the X'67' log record. You can also find the
| layout of PI trace log record X'67FA' by assembling macro ILOGREC.

In analyzing the trace output, you see not only PI trace information but also lock manager trace information.

DL/I Call Image Capture Program

This tool (DFSDLTR0), which operates independently, traces and records all DL/I calls issued by an application or multiple applications. The output is in a format acceptable as input to the DL/I test program DFSDDLT0. This allows you to create the scenario that might have caused the problem. By inserting compare statements requesting SNAP documentation of DL/I control blocks before and after the suspected failure, the information collected helps in diagnosing the problem. For details about tracing calls with the DL/I Call Image Capture trace, see "DL/I Call Image Capture" on page 209 or *IMS Version 7 Application Programming: Database Manager*.

Log Analysis (Database Related)

The IMS log is one of the most useful of all IMS service aids. Understanding log records and what information they contain can be very beneficial. For all changes, write a copy of the segment before it is changed as well as a copy of the segment after it is changed, if applicable. This process not only facilitates backout and recovery, but it also is useful for diagnosis.

Analyzing log records is helpful whenever you suspect bad data or a pointer problem. Determine where the error is by referring to error messages or to the contents of the dump. When you identify the location of the problem, use the File Select and Formatting utility (DFSERA10) to print the log records for the block in error. Refer to Table 36 on page 241 to interpret the contents of the log records. You can determine what changes to the data have been made, and in what sequence the changes were made. This information is helpful in identifying the source of the error.

Sometimes, the error is caused by an internal IMS problem; other times, the error results from incorrect data that is entered by a user or by an application.

To obtain a complete listing of all control blocks, DB, DC, and log records, assemble module DFSADSCT.

CICS puts a header on log records. To obtain the log records when running with CICS, the DD statement pointing to the CICS journal must specify DCB=RECFM=VB. This allows the File Select and Formatting utility to strip off the header.

Example: An abend is issued against a database. You have used other diagnostic tools to analyze the call. Now you must look at the database itself. Follow these steps when looking at the database:

1. Analyze the buffer to identify what seems to be wrong. (See Figure 101 on page 241.) The first indication that something is wrong is usually found in the buffer.
2. Look at the changes to that buffer (block) on the log.
3. Determine if the bad data is actually on the database.
4. If required, determine if the image copy is propagating the bad block.

Figure 101 shows the general areas of database analysis.

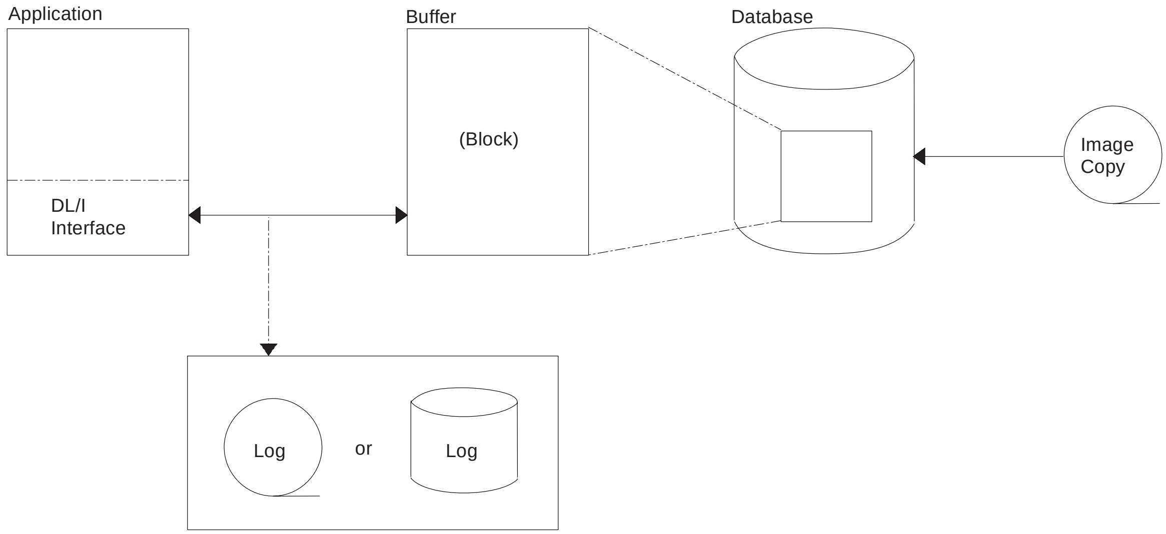


Figure 101. General Areas of Database (DB) Analysis

You can use Table 36 to assist you in the analysis of output from log record type X'50'.

If any differences are detected in the mapping of the DSECT, you can obtain a current copy by assembling the macro ILOGREC.

Table 36. Database Log Record DSECT

Offset	Field	Length	Description
DLOGB	DSECT		
00	DLENGTH	2	Length of log record
02	DLOGZZ	2	Zeros for QSAM
04	DLOGCODE	1	Log record type
05	DLOGSCDE	1	Log record subrecord (X'50' X'51' X'52')
06	DLOGPSTN	2	PST number
08	DLOGRTKN	16	Recovery token
18	DLOGSTCK	8	CPU store clock (STCK)
20	DLOGVIMS	1	DLOG IMS Version/Release X'80' Version 6
21	DDATE	3	Date from SCDDATE (yydddf)
24	DTIME	4	Time from SCDTIME (hhmmsstf)
28	DLOGDBF1	1	Flag 1 X'80' Record written during backout X'40' Record from DB/DC X'20' Record from batch region X'10' New date/time from DFSFTIM0 X'08' Commit each GU call (Mode=SNGL) X'04' First log record this sync interval X'02' First log record of a segment X'01' Last log record of a segment

Table 36. Database Log Record DSECT (continued)

Offset	Field	Length	Description
29	DLOGDBF2	1	Flag 2 X'80' Database is nonrecoverable X'40' KSDS ERASE prohibited X'20' Bit map update for lock tracing
2A	DLOGDBOR	1	Database organization X'70' DEDB direct organization X'40' DL/I HDAM database X'20' DL/I HIDAM database X'10' Data entry database (DEDB) X'08' Primary or secondary index database X'04' HISAM or SHISAM database
2B	DLOGDSOR	1	Data set organization X'80' VSAM access method X'40' OSAM access method X'08' Entry sequenced data set X'04' Key sequenced data set
2C	DPGMNAME	8	PSB name
34	DDBDNAME	8	Database name
3C	DDSID	1	Data set ID (DCB number)
3D	DLOGSLVL	1	Data share level (for DBRC registered databases)
3E	DLOGCALL	1	Describe DL/I call issued by application program X'80' ISRT call X'40' REPL call X'20' DLET call X'10' ROLL/ROLB/ROLS call (backout)
40	DLOGRBA	4	OSAM RBN or VSAM RBA (LRECL)
44	DLOGBLKO	2	Offset of RBA within block
4C	DLOGXTOF	2	Database extension section offset (not used)[1]
4E	DLOGDSOF	2	Data sharing section offset[1]
50	DLOGIDOF	2	RACF userid offset[1]
52	DLOGTKOF	2	Tracking (XRF) section offset[1]
54	DLOGDLOF	2	DL/I call section offset (not used)[1]
56	DLOGKYOF	2	Key data section offset[1]
58	DLOGSPOF	2	Space management section offset[1]
5A	DLOGUNOF	2	UNDO data offset[1]
5C	DLOGREOF	2	REDO data offset[1]
Data Sharing Section (DLOGDSHUR DSECT)			
00	DLOGDSSN	4	Data set sequence number (DSSN)
04	DLOGLSN	6	Lock sequence number (LSN)
RACF/SIGNON Userid (DLOGID DSECT)			
00	DLOGUSER	8	RACF userid
Buffer and Lock Tracking for DL/I in XRF-capable Systems (DLOGTRCK DSECT)			
00	DLOGPOOL	2	Pool size for buffer tracking
02	DLOGBUFF	2	Buffer number for buffer tracking
04	DLOGHASH	4	Root hash value

Table 36. Database Log Record DSECT (continued)

Offset	Field	Length	Description
0C	DLOGLFL1	1	Change logger lock flag X'80' Log record is for root segment X'40' Log record is for dependent segment X'20' Bypass reacquiring restart locks X'10' Get bid lock on DDATAID X'08' Function is erase X'04' Index maintenance X'02' Organization is SHISAM X'01' Hash is for logical parent
KSDS Key Data Section (DLOGKEY DSECT)			
00	DLOGKYF1	1	X'40' KSDS key X'20' Key is being erased
02	DLOGKLEN	2	Length of key
04	DLOGKDAT	variable	Key data
Space Management Section for HD Inserts and Deletes (DLOGSPCE DSECT)			
00	DLOGSPF1	1	Space management flags X'40' Demand space request X'20' Get free space request (ISRT) X'10' Free space request (DLET)
02	DLOGSOFF	2	Offset of space management request
04	DLOGSLEN	2	Length of space management request
UNDO/REDO Data Section (DLOGDATA DSECT)			
00	DLOGDFLG	1	X'80' Last data element in this section X'40' Data is compressed using MVS services
01	DLOGDFUN	1	Describe physical function being logged by this request X'80' Physical insert X'40' Physical replace X'20' Physical delete X'10' Space management create X'08' Free space element
02	DLOGDOFF	2	Offset of data in buffer
04	DLOGDLEN	2	Length of data (DLOGDDAT)
06	DLOGDDAT	variable	Variable length data
00		 2 variable	Compressed data format in DLOGDDAT Expanded data length Compressed data
	DBCKCHN	6	Back chain[2]
	DBLGSEG	8	Logical logger sequence number[2]

Notes:

1. To find each section, add the offset to the beginning of the log record.
2. The log back chain and logical logger sequence number are at the end of the log record.

Sequential Buffering Service Aids

When you receive a message or abend that indicates a problem with Sequential Buffering (SB), several diagnostic tools are available to you. Some of these tools are useful for diagnosing other IMS database-related problems; these are described elsewhere in this book:

- DL/I trace table entries
- Dump formatting of IMS control blocks
- SNAPs of IMS control blocks during pseudoabends

The //DFSSTAT statistics report is also a useful tool for evaluating a potential Sequential Buffering problem. For information about //DFSSTAT, see *IMS Version 7 Utilities Reference: Database and Transaction Manager*.

SB provides additional problem determination tools, which are described in this section:

- SBSNAP and SBESNAP options
- SB IMAGE CAPTURE option and the SB Test program (DFSSBHD0 utility)
- The SB COMPARE option

For most invocations of SB pseudoabend buffer handler functions, entries in the DL/I trace tables are provided. The SB trace table entries are:

X'6F' Search/read by RBN

X'6C' Refresh SB buffer after a write

X'69' Invalidate SB buffers

X'6A' Evaluate SB buffering

X'6B' Describe why SB was or was not used for the application

In addition, the X'D1' DL/I trace table entry created by DFSNOTB0 contains some information about invalidation of SB buffers.

SBSNAP Option

Use the SBSNAP option when you receive a message saying that either Sequential Buffering:

- Has been activated when you don’t expect it to be
- Has not been activated when you expect it to be activated

The SBSNAP option generates a SNAP of the relevant control blocks and areas involved in the calls of the OSAM buffer handler to the SB buffer handler. IMS monitors the physical I/O being done by individual applications and then uses SB I/O reference pattern-analysis algorithms to select the most efficient method of data access. When you suspect a problem with these algorithms, the SBSNAP option provides diagnostic output you can analyze. The information that is provided in the SNAPs provides an indication of why SB chose between issuing a random read of one single block and a sequential read of multiple consecutive blocks.

As a result of analyzing SBSNAP output, you might realize you need to reorganize the database, redesign the database, or set different thresholds for the SB definition. The SBSNAP option is also useful when you are tuning your usage of SB after you’ve installed IMS or migrated to a new version.

To activate the SBSNAP option, provide a SBSNAP control statement in the //DFSCTL file. (See *IMS Version 7 Installation Volume 2: System Definition and Tailoring* for detailed information.)

SNAPs are written to the IMS log as type X'67EE' records. You can format and print these records by using the File Select and Formatting Print utility with exit routine DFSERA30. For information about this utility, see *IMS Version 7 Utilities Reference: Database and Transaction Manager*.

The SBSNAP option often creates a very large amount of SNAP output. You might therefore decide to limit the SNAP to a specific short period of the application execution. To limit the SBSNAP option to one period of the application execution, use the START and STOP keywords on the SBSNAP control statement. The syntax for these keywords is:

```
START=n STOP=m
```

where *n* and *m* are the numbers of calls made to the SB buffer handler by the executing application.

To determine what values to use for *n* and *m*, look at the SPBSTCNB fields in the DL/I trace table and, if available, SNAP dumps (created by SBESNAP option). For each application, IMS maintains these call numbers in the SBPST, in its SBPSTCNB field. This field is periodically written to:

- The X'6A' DL/I trace table entry
- SNAPs that are created by the optional SBESNAP facility

Specifying START=*n* activates the SBSNAP option during the *n*the call to the SB buffer handler; specifying STOP=*m* deactivates the SBSNAP option during the *m*the call to the SB buffer handler.

SBESNAP Option

The SBESNAP option SNAPs the control blocks that are necessary for understanding the reason the SB evaluation logic did or did not recommend use of SB. You activate the SBESNAP option by providing a SBESNAP control statement in the //DFSCTL file (see *IMS Version 7 Installation Volume 2: System Definition and Tailoring* for detailed information).

SNAPs are written to the IMS log as type X'67FD' records. You can format and print these records by using the File Select and Formatting Print utility with exit DFSERA30. For information about this utility, see *IMS Version 7 Utilities Reference: Database and Transaction Manager*.

SB IMAGE CAPTURE Option and SB Test Program (DFSSBHD0 Utility)

The combined use of the SB IMAGE CAPTURE option and of the SB Test program (DFSSBHD0 utility) is useful for:

- Investigations of the SB I/O reference pattern analysis algorithms
- Investigations of the impact of changes to user-specifiable SB parameter values (the BUFSETS parameter value)

The combined use of the SB IMAGE CAPTURE option and the DFSSBHD0 utility allows the same SB buffer handler call sequence (issued during the processing of a specific real-life application with specific real-life DBs) to be run multiple times. Running the same SB buffer handler call sequence multiple times is useful when:

- You need to use the SBSNAP option but do not know exactly when to Start or Stop the SBSNAP option.
- You want to experiment with different SB algorithm parameters and observe the impact of these changes on the //DFSSTAT statistics.
- You want to test changes to the SB I/O reference pattern analysis algorithms and observe the impact of these changes on the //DFSSTAT statistics.

You activate the SB IMAGE CAPTURE option by providing a SBIC control statement in the //DFSCTL file (see *IMS Version 7 Installation Volume 2: System Definition and Tailoring* for more information). The SB Test program (DFSSBHD0 utility) is described in the *IMS Version 7 Utilities Reference: Database and Transaction Manager*.

SB COMPARE Option

You activate the SB COMPARE option when you suspect that the SB buffer handler returns incorrect block images into the buffers of the OSAM buffer handler. When you activate the SB COMPARE option, the SB buffer handler performs a self-check to see whether this suspicion is correct and provide problem determination information when the SB buffer handler really returns incorrect data.

When the SB COMPARE option is active, the SB buffer handler compares each block image that is returned to the OSAM buffer handler with the corresponding block image that is stored on DASD. When the comparison detects a mismatch between the two block images, the SB buffer handler invokes the SNAP-specific function, which produces a SNAP that describes the mismatch and contains:

- Relevant buffers and control blocks of DL/I
- The OSAM buffer handler
- The SB buffer handler

Module DFSSBSN0 then issues an abend (for batch) or a pseudoabend (for DB/DC, DBCTL, and CICS).

Exception: In a data-sharing environment, the SB buffer handler sometimes returns a back-level block image to the OSAM buffer handler. Therefore, in data sharing, the SB COMPARE option does not issue abends or pseudoabends.

You activate the SB COMPARE option by providing a SBCO control statement in the //DFSCTL file. Refer to *IMS Version 7 Installation Volume 2: System Definition and Tailoring* for more information on the SBCO control statement in the //DFSCTL file.

SNAPs are written to the IMS log as type X'67EF' records. You can format and print these records by using the File Select and Formatting Print utility with exit DFSERA30. For information about this utility, see *IMS Version 7 Utilities Reference: Database and Transaction Manager*.

GSAM Control Block Dump—DFSZD510

When a GSAM error occurs or when a DUMP or SNAP call is issued to a GSAM PCB, a formatted dump of the GSAM control blocks is written to the file that is defined as DDNAME IMSERR or SYSPRINT. You can use this GSAM control block dump (named DFSZD510) to diagnose GSAM problems.

Example: Some situations in which you would use a GSAM control block dump are when you receive a message identifying a GSAM error, or when you are having problems repositioning a GSAM data set when you are trying to restart an application that previously failed.

The control blocks that are included in the dump are the:

- GSAM pointer table (GPT)
- GSAM load table (GLT)
- GSAM data set control block (GB)
- GSAM queue control block (GQCB)
- GSAM buffer control block (GBCB)
- IMS program control block (PCB)
- Data event control block (DECB)
- Request parameter list (RPL)

To produce a DSECT that shows the layout of the GSAM control blocks, assemble macro IGLI.

Figure 102 on page 247 shows an example of a formatted GSAM control block dump, and Figure 103 on page 248 shows an example of an unformatted GSAM control block dump.

Example of a Formatted GSAM Control Block Dump

In Figure 102, key eye catchers are shown in boldface to make these parts of the dump easier for you to find. Each problem is different, but diagnosing almost all GSAM problems will involve at least these key areas of the dump.

```
                                    * * * GSAM CONTROL BLOCKS DUMP * * *
 07A010 GSAM POINTER TABLE
             GPTCNTLR 800271D8  GPTERROR       00  GPTFC        GHU  GPTF1    0007A220  GPTF2    0004D50C
             GPTF3    00000000  GPTF4    00000000  GPTGB    0007A0C0  GPTGLT   0007A060  GPTHSEVC       08
             GPTMAIN  00001350  GPTMODE        00  GPTPCB   0007A090  GPTPMBLK 00009C90  GPTPSBL  00005540
             GPTRS1   00009C58  GPTSAVE  00079000  GPTSZS       0800  GPTSZW       0800  GPTTRACE 00009DF0
             GPTTYPE        00  GPTWORK  00079800
 07A060 GSAM LOAD TABLE
             GLTBSAM  8007B0C0  GLTBUFIO 00000000  GLTCBDMP 8007CCB0  GLTCNTLR 800271D8  GLTGPT   0007A010
             GLTOPENB 80032118  GLTOPENV 00000000  GLTVSAM  00000000
 07A090 IMS PGM CONTROL BLK
             DBPCBDBD DBD37877  DBPCBFLG       02  DBPCBGB  0207A0C0  DBPCBLEV     0000  DBPCBMKL 0000000C
             DBPCBNSS 0000FFFF  DBPCBPRO     L     DBPCBSFD           DBPCBSTC       AM  DBPCBURL 00000000
             DBPCBRRA 00000000  00000000
 07A0C0 GSAM BLOCK
             GBBFPORT     0000  GBBLKLEN     0000  GBBLKOH1     0001  GBBLKOH2     FFE0  GBBLKREF 00000401
             GBBLKSI      01C2  GBBQCB   00000000  GBBUFFER 00064CA0  GBBUFFSW       08  GBBUFNO        01
             GBCDISP      0000  GBCHAIN  0007A220  GBCRTNCD     0028  GBCSEVCD       08  GBCTRS       0000
             GBDCBPTR B007A178  GBDDNAME GS378770  GBDECB   0007A1D4  GBDEVTYP     208E  GBDSORG        81
             GBERRSW        00  GBEXLST  8607BEA2  GBGPTPTR 0007A010  GBGSAMSW       50  GBIOAREA 00093000
             GBLENLEN     0000  GBLRECL      0096  GBMAXTR      BB60  GBMINRCL     0000  GBNVOL       0001
             GBOPENSW       D1  GBPCBPTR 0007A090  GBPRTNCD     0000  GBRECFM        90  GBRECPTR 00064D36
             GBREQC       6201  GBREQP       0020  GBREQU       6201  GBRPLPTR 0007A1D4  GBRRAPTR 00091B88
             GBSERA       0000  GBSERR       0600  GBSUPVR        00  GBTRCALC     BB60  GBTRECL      0096
             GBURTNCD       AM  GBVLSQ       0001
 07A178 DATA CONTROL BLOCK (DCB)
             DCBBFTEK       06  DCBBLKCT 04FDBEBC  DCBBLKSI     01C2  DCBBUFCB 01064C98  DCBBUFL      01C2
             DCBBUFNO       01  DCBBUFOF       00  DCBCHECK 00C894B0  DCBCIND1       00  DCBCIND2       00
             DCBCNTRL 00D57F48  DCBDDNAM           DCBDEBAD 009D1554  DCBDEN         AD  DCBDEVT        2E
             DCBDSORG     4000  DCBDVTBA   FDBEBC  DCBEOBR  01D57650  DCBEOBW  00D57650  DCBEODA    07BEBA
             DCBEODAD 0607BEBA  DCBEXLST 9007A110  DCBFDAD1 00000000  DCBFDAD2 05000104  DCBFUNC        A0
             DCBIFLG        C8  DCBIFLGS       00  DCBIOBA  410050F0  DCBIOBAD 00005088  DCBIOBL        09
             DCBKEYCN       00  DCBKEYLE       00  DCBLRECL     0096  DCBMACR      97D8  DCBMACRF     2424
             DCBMODE        00  DCBNCP         01  DCBODEB  00005088  DCBOFFSR       30  DCBOFFSW       30
             DCBOFLGS       92  DCBOPTCD       00  DCBPRTOV       AD  DCBPRTSP       00  DCBREAD  92C897D8
             DCBRECFM       90  DCBREL     2EADA0  DCBRELAD 00000000  DCBRELB  002EADA0  DCBSTACK       00
             DCBSVCXL 00005088  DCBSYNA    07BF68  DCBSYNAD 0907BF68  DCBTIOT      007C  DCBTRBAL     ADA0
             DCBTRTCH       00  DCBWCPL        01  DCBWCPO        30  DCBWRITE 92C897D8
 07A1D4 DECB
             7F000000  00200000  B007A178  00064CA0  000050F8  00000000
        064CA0 GB BUFFER
      064CA0   D7C1D9E3  D5E4D460  F0F0F0F0  F0F0F940  40404040  40404040  40404040  40404040   *PARTNUM.0000009                 *
      064CC0   40404040  40404040  40404040  40404040  40404040  40404040  40404040  40404040   *                                *
      064CE0   40404040  40404040  40404040  40404040  40404040  40404040  40404040  40404040   *                                *
      064D00   40404040  40404040  40404040  40404040  40404040  40404040  40404040  40404040   *                                *
      064D20   40404040  40404040  40404040  40404040  40404040  4040D7C1  D9E3D5E4  D460F0F0   *                      PARTNUM.00*
      064D40   F0F0F0F1  F0404040  40404040  40404040  40404040  40404040  40404040  40404040   *00010                           *
      064D60   40404040  40404040  40404040  40404040  40404040  40404040  40404040  40404040   *                                *
      064D80   40404040  40404040  40404040  40404040  40404040  40404040  40404040  40404040   *                                *
      064DA0   40404040  40404040  40404040  40404040  40404040  40404040  40404040  40404040   *                                *
      064DC0   40404040  40404040  40404040  D7C1D9E3  D5E4D460  F0F0F0F0  F0F0F840  40404040   *            PARTNUM.0000008     *
      064DE0   40404040  40404040  40404040  40404040  40404040  40404040  40404040  40404040   *                                *
      064E00   40404040  40404040  40404040  40404040  40404040  40404040  40404040  40404040   *                                *
      064E20   40404040  40404040  40404040  40404040  40404040  40404040  40404040  40404040   *                                *
      064E40   40404040  40404040  40404040  40404040  40404040  40404040  40404040  40404040   *                                *
      064E60       4040                                                                         *  ..............................*
 07A1F0 IMS PGM CONTROL BLK
             DBPCBDBD DBD3787X  DBPCBFLG       02  DBPCBGB  0207A220  DBPCBLEV     0000  DBPCBMKL 0000000C
             DBPCBNSS 0000FFFF  DBPCBPRO     G     DBPCBSFD           DBPCBSTC           DBPCBURL 00000000
             DBPCBRRA 00000000  00000000
 07A220 GSAM BLOCK
             GBBFPORT     0000  GBBLKLEN     0000  GBBLKOH1     0001  GBBLKOH2     FFE0  GBBLKREF 00000000
             GBBLKSI      01C2  GBBQCB   00000000  GBBUFFER 00000000  GBBUFFSW       00  GBBUFNO        01
             GBCDISP      0000  GBCHAIN  0007A0C0  GBCRTNCD     0000  GBCSEVCD       00  GBCTRS       0000
             GBDCBPTR 8007A2D8  GBDDNAME GS378770  GBDECB   0007A334  GBDEVTYP     208E  GBDSORG        81
             GBERRSW        00  GBEXLST  00000000  GBGPTPTR 0007A010  GBGSAMSW       00  GBIOAREA 00000000
             GBLENLEN     0000  GBLRECL      0096  GBMAXTR      BB60  GBMINRCL     0000  GBNVOL       0001
             GBOPENSW       C0  GBPCBPTR 0007A1F0  GBPRTNCD     0000  GBRECFM        90  GBRECPTR 00000000
             GBREQC       0020  GBREQP       0020  GBREQU       0020  GBRPLPTR 0007A334  GBRRAPTR 00000000
             GBSERA       0000  GBSERR       C200  GBSUPVR        00  GBTRCALC     BB60  GBTRECL      0000
             GBURTNCD           GBVLSQ       0000
```

Figure 102. Formatted GSAM Control Block Dump (Part 1 of 2)

```
07A2D8 DATA CONTROL BLOCK (DCB)
             DCBBFTEK       00  DCBBLKCT 00000000  DCBBLKSI     01C2  DCBBUFCB 00000000  DCBBUFL      01C2
             DCBBUFNO       00  DCBBUFOF       00  DCBCHECK 00000001  DCBCIND1       00  DCBCIND2       00
             DCBCNTRL 00000001  DCBDDNAM GS378770  DCBDEBAD F8F7F7D6  DCBDEN         00  DCBDEVT        00
             DCBDSORG     4000  DCBDVTBA   000000  DCBEOBR  01000001  DCBEOBW  00000001  DCBEODA    000001
             DCBEODAD 00000001  DCBEXLST 90000000  DCBFDAD1 00000000  DCBFDAD2 00000000  DCBFUNC        00
             DCBIFLG        00  DCBIFLGS       F8  DCBIOBA  00000001  DCBIOBAD 00000001  DCBIOBL        00
             DCBKEYCN       00  DCBKEYLE       00  DCBLRECL     0096  DCBMACR      2424  DCBMACRF     F3F7
             DCBMODE        00  DCBNCP         01  DCBODEB  00000001  DCBOFFSR       00  DCBOFFSW       00
             DCBOFLGS       02  DCBOPTCD       00  DCBPRTOV       00  DCBPRTSP       00  DCBREAD  02002424
             DCBRECFM       90  DCBREL     000000  DCBRELAD 00000000  DCBRELB  00000000  DCBSTACK       00
             DCBSVCXL 00000001  DCBSYNA    000001  DCBSYNAD 00000001  DCBTIOT      C7E2  DCBTRBAL     0000
             DCBTRTCH       00  DCBWCPL        00  DCBWCPO        00  DCBWRITE 02002424
07A334 DECB
             00000000  00800000  00000000  00000000  00000000  00000000
                                   ***END OF DUMP***
```

Figure 102. Formatted GSAM Control Block Dump (Part 2 of 2)

Example of an Unformatted GSAM Control Block Dump

```
0007A000 C7E2C1D4 40C2D3D6 C3D2E240 C8C5D9C5    800271D8 00000000 0007A060 00005540   *GSAM BLOCKS HERE...Q........... *
0007A020 00009C90 00009DF0 0007A1F0 0007A220    D7E4D9C7 0007A0C0 00000000 00079800   *.......0...0....PURG............*
0007A040 00079000 08000800 00001350 00009C58    0007A0C0 00005180 00000000 00000000   *................................*
0007A060 800271D8 0007A010 00000000 80032118    8007B0C0 00000000 00000000 00000000   *...Q............................*
0007A080 00000000 8007CCB0 00000000 00000000    C4C2C4F3 F7F8F7F7 00004040 D3404040   *................DBD37877..  L   *
0007A0A0 0207A0C0 40404040 40404040 0000000C    0000FFFF 00000000 00000000 00000000   *....        ....................*
0007A0C0 0007A220 00000401 00010000 00010096    00000096 01C20000 0000208E 0001FFE0   *.....................B..........*
0007A0E0 40400000 00289081 06000000 02830283    12020000 5008D101 00000000 00000000   *  ....................J.........*
0007A100 0007A010 0007A090 B007A178 0007A1D4    8607BEA2 00093000 00091B88 00000000   *...............M................*
0007A120 00064CA0 00064D36 BB60BB60 00000000    C7E2F3F7 F8F7F7D6 00000000 00000000   *................GS378770........*
0007A140 00000000 00000000 00000000 00000000    00000000 00000000 00000000 00000000   *................................*
      LINE 0007A160  SAME AS ABOVE
0007A180 00050001 04FDBEBC 002EADA0 01064C98    01C24000 00005088 0607BEBA 9007A110   *.................B .............*
0007A1A0 007C2424 009D1554 92C897D8 00C894B0    0907BF68 000001C2 30013030 410050F0   *.........H.Q.H.........B.......0*
0007A1C0 01D57650 00D57650 00000096 00D57F48    00000000 7F000000 00200000 B007A178   *.N...N.......N..................*
0007A1E0 00064CA0 000050F8 00000000 00000000    C4C2C4F3 F7F8F7E7 00004040 C7404040   *.......8........DBD3787X..  G   *
0007A200 0207A220 40404040 40404040 0000000C    0000FFFF 00000000 00000000 00000000   *....        ....................*
0007A220 0007A0C0 00000000 00000000 00010096    00000000 01C20000 0000208E 0001FFE0   *.....................B..........*
0007A240 40400000 00009081 C2000000 02830283    00200000 0000C001 00000000 00000000   *  ......B........................*
0007A260 0007A010 0007A1F0 8007A2D8 0007A334    00000000 00000000 00000000 00000000   *.......0...Q....................*
0007A280 00000000 00000000 BB60BB60 00000000    C7E2F3F7 F8F7F7D6 00000000 00000000   *................GS378770........*
0007A2A0 00000000 00000000 00000000 00000000    00000000 00000000 00000000 00000000   *................................*
      LINE 0007A2C0  SAME AS ABOVE
0007A2E0 00000000 00000000 00000000 00000000    01C24000 00000001 00000001 90000000   *.................B .............*
0007A300 C7E2F3F7 F8F7F7D6 02002424 00000001    00000001 000001C2 00000000 00000001   *GS378770...............B........*
0007A320 01000001 00000001 00000096 00000001    00000000 00000000 00800000 00000000   *................................*
0007A340 00000000 00000000 00000000 00000000                                          *................                *
0007A700 84000000 18800000 000300CC FFFB26B4    00000000 00000000 00000000 00000000   *................................*
0007A720 0004D0A8 00080008 0004D0B0 00100010    0004D0B2 00020002 0004D0B4 0004D0B8   *................................*
0007A740 0004D0BC 0004D0C0 00080008 0004D0C8    00080008 00000000 40404040 40404040   *...............H........        *
0007A760 10004040 40404040 40404040 40404040    40404040 40404040 40404040 40404040   *..                              *
0007A780 00000000 00000000 00000000 00000000    00000000 00000000 00000000 00000000   *................................*
0007A7A0 00000000 00000000 00000000 00000000    00000000 00000000 0004D114 00009DF0   *..........................J....0*
0007A7C0 009B6020 00000000 00000000 00093000    0004DD54 00000000 00000000 00029E50   *................................*
0007A7E0 00000000 00000000 00000000 00000000    C4C4D3E3 F0F1F340 D3D6C1C4 40404040   *................DDLT013 LOAD    *
0007A800 00000000 00000000 00000000 00000000    00000000 00000000 00000000 00000000   *................................*
      LINES 0007A820-0007A860  SAME AS ABOVE
0007A880 00000000 00008500 0004D0A8 00000000    00000000 080073E8 00000000 00000000   *.......................Y........*
0007A8A0 00000000 00000000 00000000 00000000    00000000 00000000 00000000 00000000   *................................*
0007A8C0 00000000 080073E8 0004D050 00000000    00000000 00000000 00026B70 000641D8   *.......Y.......................Q*
0007A8E0 00000000 00000000 00000000 00000000    00000000 00000000 00000000 00053040   *............................... *
0007A900 00000000 00000000 000300CC 00000000    00000000 00000000 00000000 00000000   *................................*
0007A920 00000000 00000000 0004D94C 00000000    00000000 00000000 00000000 00000000   *..........R.....................*
0007A940 00000000 00000000 00000000 00000000    00000000 00000000 00000000 00000000   *................................*
0007A960 00000000 00000000 00000000 00000000    00009DA0 00000000 00010C00 0004D350   *..............................L.*
0007A980 00000000 00000000 00000000 00000000    00000000 00000000 00000000 00000000   *................................*
```

Figure 103. Unformatted GSAM Control Block Dump (Part 1 of 2)

```
      LINE 0007A9A0  SAME AS ABOVE
0007A9C0 84000000 00000000 00000000 00000000    00000000 00000000 00000000 00000000   *................................*
0007A9E0 00000000 00000000 00000000 00000000    00000000 0088266F 11173205 02000000   *................................*
0007AA00 0004D050 00000000 00000000 00000000    00000000 00000000 00000000 00000000   *................................*
0007AA20 00000000 00000000 00000000 00000000    00000000 00000000 00000000 00000000   *................................*
0007AA40 00060040 00000000 00000000 00000000    00000000 00000000 00000000 00000000   *... ............................*
0007AA60 00000000 00000000 00000000 00000000    00000000 00000000 00000000 00000000   *................................*
0007AA80 00000000 00000000 00000000 00000000    00000000 00000000 00000000 0002CEE6   *...............................W*
0007AAA0 00000000 00000000 00000000 0005713F    00057040 00000000 07FC4040 40404040   *................... ......      *
0007AAC0 00000000 00000000 00000000 00057340    00000000 00057140 00000000 00000000   *............... ....... ........*
0007AAE0 00000000 00000000 00000000 00000000    00000000 00000000 00000000 00000000   *................................*
      LINE 0007AB00  SAME AS ABOVE
0007AB20 00000000 00000000 0004DD64 00000000    00000000 00000000 0004DD90 00000000   *................................*
0007AB40 00000000 00000000 00000000 08000000    000C0000 00000000 00000000 00000000   *................................*
0007AB60 00000000 00000000 00000000 00000000    00000000 00000000 00000000 00000000   *................................*
      LINE 0007AB80  SAME AS ABOVE
0007ABA0 00000000 00000000 00000000 00000000    00000000 00000000 00000000 80049040   *............................... *
0007ABC0 00009DA8 0004D554 4003AAE2 0002AF6C    00009DA0 00009DF0 00009C90 00009DF0   *......N. ..S...........0.......0*
0007ABE0 0008FD64 00009DF8 00093000 00000000    00093000 00090548 00000004 0004D050   *.......8........................*
0007AC00 0003A7A0 00000000 0004D50C 0004D59C    FF02AFD2 0002BCEC 00009DA0 0004D050   *..........N...N....K............*
0007AC20 00009C90 0004D50C 00009E38 00009D90    00093000 00000000 80093000 080073E8   *......N........................Y*
0007AC40 00000004 0004D050 0002AF6C 00000000    0004D554 0004D5E4 FF02D5C6 00034100   *..................N...NU..NF....*
0007AC60 000073E8 0004D050 0002D9D0 00009DF0    0002CCD8 00029E50 080073E8 00005500   *...Y......R....0...Q.......Y....*
0007AC80 00093000 00000000 0004D0A8 0004D050    0002BCD8 00000000 0004D59C 0004D62C   *...................Q......N...O.*
0007ACA0 FF034196 00034BD4 000073E8 0004D050    0002D9D0 00009DF0 0002CCD8 00029E50   *.......M...Y......R....0...Q....*
0007ACC0 080073E8 00005500 00093000 0004D050    0004D0A8 080073E8 00034100 00000000   *...Y...................Y........*
0007ACE0 0004D5E4 0004D674 FF034E7A 0003C8B8    00000000 0004D050 0002D9D0 00009DF0   *..NU..O.......H...........R....0*
0007AD00 0002CCD8 00029E50 080073E8 00005500    00093000 0004D050 00000000 080073E8   *...Q.......Y...................Y*
0007AD20 00034BD4 00000000 0004D62C 0004D6BC    00000000 00000000 00000000 00000000   *...M......O...O.................*
0007AD40 00000000 00000000 00000000 00000000    00000000 00000000 00000000 00000000   *................................*
0007AD60 00000000 00000000 00000000 00000000    0004D674 0004D704 00000000 00000000   *..................O...P.........*
0007AD80 00000000 00000000 00000000 00000000    00000000 00000000 00000000 00000000   *................................*
0007ADA0 00000000 00000000 00000000 00000000    00000000 00000000 0004D6BC 0004D74C   *..........................O...P.*
0007ADC0 00000000 00000000 00000000 00000000    00000000 00000000 00000000 00000000   *................................*
      LINE 0007ADE0  SAME AS ABOVE
0007AE00 0004D704 0004D794 00000000 00000000    00000000 00000000 00000000 00000000   *..P...P.........................*
0007AE20 00000000 00000000 00000000 00000000    00000000 00000000 00000000 00000000   *................................*
0007AE40 00000000 00000000 0004D74C 0004D7DC    00000000 00000000 00000000 00000000   *..........P...P.................*
0007AE60 00000000 00000000 00000000 00000000    00000000 00000000 00000000 00000000   *................................*
0007AE80 00000000 00000000 00000000 00000000    0004D794 0004D824 00000000 00000000   *..................P...Q.........*
0007AEA0 00000000 00000000 00000000 00000000    00000000 00000000 00000000 00000000   *................................*
0007AEC0 00000000 00000000 00000000 00000000    00000000 00000000 0004D7DC 0004D86C   *..........................P...Q.*
0007AEE0 00000000 00000000 00000000 00000000    00000000 00000000 00000000 00000000   *................................*
      LINE 0007AF00  SAME AS ABOVE
0007AF20 0004D824 0008BD98 00000000 00000000    00000000 00000000 00000000 00000000   *..Q.............................*
0007AF40 00000000 00000000 00000000 00000000    00000000 00000000 00000000 00000000   *................................*
0007AF60 00000000 00000000 0004D86C 0004D8FC    00000000 00000000 00000000 00000000   *..........Q...Q.................*
0007AF80 00000000 00000000 00000000 00000000    00000000 00000000 00000000 00000000   *................................*
0007AFA0 00000000 00000000 00000000 00000000    0004D8B4 00000000 00000000 00000000   *..................Q.............*
0007AFC0 00000000 00000000 00000000 00000000    00000000 00000000 00000000 00000000   *................................*
0007AFE0 00000000 00000000 00000000 00000000    00000002 00000000 D3C7E6C1 00518000   *........................LGWA....*
00081560                   47F0F034 2FC4C6E2    C6D3D3C7 F060F1F3 F060D3D6 C7C9C3C1   *        .00..DFSFLLG0.130.LOGICA*
```

Figure 103. Unformatted GSAM Control Block Dump (Part 2 of 2)

Recovering from Out-of-Space Sx37 Abends on GSAM Data Sets

When an application program is inserting records into a GSAM DASD data set and space on the data set runs out, an Sx37 abend occurs. The proper restart procedure depends on the physical characteristics of the GSAM data set and IMS's method of checkpointing the position in the data set. For information about repositioning GSAM data sets, see the "XRST Call" section in *IMS Version 7 Application Programming: Database Manager*.

When an Sx37 abend occurs, you typically solve the problem by copying the data set and allocating more space for the copy. You can copy the data set with IEBGENER or some other utility that reads and writes logical records. Do not do this for blocked GSAM BSAM DASD data sets if you plan to restart using the copy. You must copy the physical records, not just the logical records. You can use IEBGENER for this, but you must specify different DCB parameters.

You can use the following procedure to recover from an Sx37 abend on a blocked GSAM data set. (A blocked data set has a record format of FB or VB.)

1. Copy the file to a larger data set using IEBGENER, but specify RECFM=U for the record format. You must use RECFM=U for both the input and output data sets. This copies the physical records as they exist. No reblocking is done. The copy must be to a like device type (one with the same track size). If the data set resides on multiple volumes, only the last volumes of data can be copied. GSAM keeps position by relative volume, by relative track within the volume, and by relative physical block within the track
2. You must change the RECFM parameter for the copied file back to its original value, FB or VB. You can do this with any program that opens the data set. It is straightforward to do this using IEBGENER. Execute IEBGENER with a SYSUT2 statement referring to the new data set. This DD statement must specify DCB=(RECFM=*xx*), where *xx* is the original GSAM data set record format value. You must also specify DISP=MOD. SYSUT1 must be a dummy data set. This causes IEBGENER to open the data set for output. IEBGENER does not copy any records to the data set, but it will rewrite the DSCB with the updated RECFM value at close time.
3. You can now use the copy to restart the program from a checkpoint.

If the GSAM data set resides on SMS-managed volumes, you can use the following procedure:

1. Under SMS, add extra volumes to the storage group, if necessary, and increase the number of volumes allowed for the DATACLAS keyword.
2. Using IDCAMS, enter the command `ALTER dsn ADVOL(*)` to indicate that additional volumes are available to the data set.

Chapter 9. DC—Data Communication Service Aids

This chapter describes diagnostic aids and techniques used during data communication problem analysis. It does not apply to a Database Control (DBCTL) environment. Included are:

- The terminal communication task trace, which shows the last few communications analyzer and device-dependent module interactions
- The data communication (DC) trace, which accumulates a history of device and line activity on the IMS log data set
- The DLA3LOG trace, which is useful in analyzing problems associated with IMS and the application program
- A procedure to help you determine if any receive-any buffers are left
- A procedure to help you find the active save set
- A description of the IMS-VTAM interface
- IBM 3270 error recovery analysis
- Message Format Service normal logic flow for BTAM activity
- Message Format Service module traces

Terminal Communication Task Trace

When you experience a hung output device (such as a terminal, line, or node), you can use the terminal communication task trace to diagnose the problem.

You can use information you find in the terminal communication task trace to build keywords for your search string, or you can use the information when you are reviewing existing APAR descriptions to determine whether they describe the problem you are experiencing.

All IMS terminal communication tasks are dispatched by the IMS communication analyzer (module DFSICIO0). This module traces its own flow, as well as the flow through device-dependent modules (DDMs), by using register 0 of the communication analyzer's save area. (For this reason, this trace is often referred to as the REG0 trace.) The communication analyzer uses the high-order 2 bytes of register 0 to trace the analyzer entry point, and it uses the low-order 2 bytes to trace the DDM entry point.

In the DC portion of the IMS dump, find the save area sets that hold data about the various IMS processes that were executing prior to the dump. If one of these save areas sets is for DFSICIO0, you can then look at the corresponding register 0 to find the communication task trace entries.

Entry Points

The following list identifies the analyzer entry points. Look at the content of register 0 (for module DFSICIO0); the high-order 2 bytes of register 0 identify the analyzer entry points.

Analyzer Entry Point (Hex)
Processing Description

1 Process an input segment from a terminal.

2 Perform a logical read operation to the terminal.

3 Determine which system function is to be performed next for this line and terminal (or node).

4 Issue GET NEXT to message queue.

5 Perform a logical write operation to the terminal.

6 WRITE successful; dequeue message or call DDM at DD1.

7 Notify master terminal of I/O error; cancel input; return output message to queue.

8 Return output message to queue; cancel input.

9 Generate an error message; cancel input; return output message to queue.

A Idle the line; cancel output; return output message to queue.

B Resend the last message sent from a given LTERM.

C Idle the line.

The low-order 2 bytes of register 0 identifies the entry points for the device-dependent modules (DDMs), as listed below:

DDM Entry Point (Hex)
Processing Description

1 WRITE/SEND setup: Set up output buffer to write current buffer.

2 WRITE/SEND interruption: Error check last output operation.

3 READ/RECEIVE setup: Set up to perform a poll or read.

4 READ/RECEIVE interruption: Error check, determine terminal responding, and deblock input segment.

5 Cleanup: Restore control blocks after DFSICI00 error.

6 Build: Move output message from a queue buffer (MFS buffer) to a line buffer.

7 Logon: VTAM OPNDST/CLSDST processing.

8 Prepare for output: VTAM

F MFS output format control (DFSCOFC0) was entered.

Trace Records

The entries in the first 2 bytes indicate what processing the analyzer (DFSICIO0) has performed. The entries in the last 2 bytes indicate what processing the DDMs have performed. As new entries are added, existing entries shift to the left. When the 2-byte area fills, the oldest entry is overwritten by the next-oldest entry. Therefore, the right-most entry of each 2-byte portion of register 0 identifies the most recent analyzer or DDM activity.

Figure 104 shows the format of a sample terminal communications task trace record.

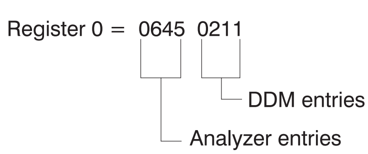


Figure 104. Example of a Terminal Communication Task Trace Entry

The sample terminal communication task trace entry in Figure 104 indicates that the analyzer entries are 6, 4, and 5; DDM entries are 2, 1, and 1. An analysis of this trace data would yield the flow information shown in Table 37.

Table 37. Example Processing Flow for a Terminal Communication Task Trace Entry

Entry Point	Trace ID	Processing Description
2	DDM2	A write interrupt occurred.
6	A06	Write completed successfully.
1	DDM1	Another buffer was required.

Table 37. Example Processing Flow for a Terminal Communication Task Trace Entry (continued)

Entry Point	Trace ID	Processing Description
4	A04	Room in the buffer is allowed for another message segment. (GN was issued to the message queue.)
1	DDM1	This segment was placed in the buffer, filling it or EOM was detected. Setup for the write operation was completed.
5	A05	Output operation was requested.

Trace Output

You can find the terminal communication task trace in any IMS dump, either in register 0 (corresponding to module DFSICIO0) or in the CLB section of the dump for the terminal involved in the problem.

If you look at the CLB section of the dump, the information in field CLBTEMP1 is the same as what is in register 0 (described in "Trace Records" on page 252). Fields CLBTEMP4 and CLBTEMP5 contain the Julian date and time at which the IMS task (ITASK) associated with the line or node returned to the IMS dispatcher (module DFSIDSP0). This information is useful when diagnosing a hung or lost terminal. In an IMS control region dump, you can determine when the last activity occurred on the line or node and what processing path was taken.

DC Trace

The data communication (DC) trace enables you to obtain information about the program flow within the communications analyzer and between the analyzer and the device dependent modules (DDMs).

Starting the Trace

To start the DC trace for any terminal in the IMS network, enter one of the following /TRACE commands from the master terminal or the MVS console.

Specify at least level 3 in the command because buffer contents are usually required for complete diagnosis. If you specify level 4, the trace writes a save area set for certain entries (C00-C12, D05, AER1, and AER2).

- For VTAM terminals:
```
/TRACE SET ON NODE P1 LEVEL 1|2|3|4 MODULE DDM|MFS|ALL
```
- For BTAM terminals:
```
/TRACE SET ON LINE P1 LEVEL 1|2|3|4 MODULE DDM|MFS|ALL
```
- For ISC links:
```
/TRACE SET ON NODE P1 LEVEL=1|2|3|4 MODULE DDM|MFS|ALL
                      or
/TRACE SET ON NODE P1 USER P2
```
- For logical LINKs:
```
/TRACE SET ON LINK P1,..,Pn|ALL LEVEL 1|2|3|4 MODULE DDM|MFS|ALL
```
- For UNITTYPE:
```
/TRACE SET ON UNITTYPE P1,..,Pn LEVEL 1|2|3|4 MODULE DDM|MFS|ALL
```
- For an XRF environment:
```
/TRACE SET ON NODE xxx TAKEOVER

/TRACE SET ON LINE xxx TAKEOVER

/TRACE SET ON LINK xxx TAKEOVER
```

For a detailed description of the `/TRACE` command, see *IMS Version 7 Command Reference*.

XRF Notes

- The `/TRACE SET ON NODE xxx TAKEOVER` command starts the trace for the specified terminals during takeover only.
- You can enter this command only from the active system in an XRF environment.
- After a terminal has switched successfully, the trace is automatically turned off for that terminal.
- Because this command is recovered across restart and takeover, you need to enter it only once. After a cold start, you must enter the command again.
- Tracing occurs only if the session was active at the time of the takeover.
- If you enter a /TRACE command with and without the TAKEOVER keyword, the last command you entered is in effect.
- You can issue this command for VTAM nodes, MSC links, and BTAM lines during takeover.
- The /TRACE SET OFF NODE xxx TAKEOVER, /TRACE SET OFF LINE xxx TAKEOVER, or /TRACE SET OFF LINK xxx TAKEOVER command turns off the trace anytime before takeover.

Stopping the Trace

To stop the DC trace, enter one of the following commands from the master terminal or the MVS console.

- For VTAM terminals:
  ```
  /TRACE SET OFF NODE P1
  ```
- For BTAM terminals:
  ```
  /TRACE SET OFF LINE P1
  ```
- For ISC links:
  ```
  /TRACE SET OFF NODE P1
                    or
  /TRACE SET OFF NODE P1 USER P2
  ```
- For logical LINKs:
  ```
  /TRACE SET OFF LINK P1,...,Pn|ALL
  ```
- For UNITTYPE:
  ```
  /TRACE SET OFF UNITTYPE P1,...Pn
  ```
- For an XRF environment:
  ```
  /TRACE SET OFF NODE xxx TAKEOVER

  /TRACE SET OFF LINE xxx TAKEOVER

  /TRACE SET OFF LINK xxx TAKEOVER
  ```

Printing the Trace Records

The DC trace snaps DC control blocks and I/O buffers to the OLDS/WADS as X'6701' log records. These records are archived to the system log data set (SLDS). To print the trace records, use the File Select and Formatting Print utility (DFSERA10). Specify E=DFSERA30 to format the records before printing. The following example shows the JCL you might use to print DC trace records.

```
//  JOB jobname
//S EXEC PGM=DFSERA10
//SYSPRINT DD SYSOUT=A
//SYSUT1   DD DSN=DSN of SLDS,..............
//SYSIN    DD *
CONTROL  CNTL
OPTION   PRINT O=5,V=6701,L=2,T=X,E=DFSERA30
//
```

where

O = Offset
L = Length

V = Value
T = Type
E = Exit

Even if the DC trace was started for many terminals, you can print trace entries for a specific terminal by using the following OPTION statement.

```
CONTROL CNTL DDNAME=....
OPTION  PRINT O=5,T=X,L=1,V=67,C=M
OPTION  PRINT O=89,T=C,L=8,V=xxxxxxxx,C=E,E=DFSERA30
```

where xxxxxxxx = terminal (node) name

Be aware that a trace record might span several X'6701' log records. If you use the OPTIONS statements above, only the first log record is printed.

For complete instructions on running the File Select and Formatting Print utility, see *IMS Version 7 Utilities Reference: System*.

Content of the Trace Records

You can evaluate DC trace records when doing any of the following activities:

- Debugging user errors in exit routines or user modifications relating to communications
- Debugging errors in other entities in the communication network (such as programmable terminals or other host processors)
- Building a keyword string to search for known problems
- Evaluating existing APAR descriptions to isolate problems that are most like the one you are experiencing

The first line of each trace record shows the ID:

```
ID= xxx    SEGNO= mm RECNO= nnnnnnnn TIME HH.MM.SS.TT DATE YY.DDD
```

xxx can be any of the following trace record identifiers (IDs):[4]

ID	Description
A xx	Communication analyzer activity (DFSICIO0)
AERx	Access method error
C xx	Communication analyzer activity (DFSCIOC0 in DFSICIO0)
CI04	TM shared queues re-read error detected
CIO2	DDM SDC read for output
CIO3	DDM conditional SDC ’wash’ output
CMEA	Before calling Message Control/Error exit DFSCMUX0
CMEB	After calling Message Control/Error exit DFSCMUX0
CMEI	Message Control/Error exit interface processing
COFC	Entry to the output format control, MFS-supported devices (DFSCOFC0)
CRTU	Output User Creation user exit routine failure
CVCT	VTAM trace. This log record is written even though DC trace is not active on the terminal/link.
CVCV	XRF class 2 takeover trace. This log record is written for XRF class 2 terminals during takeover, even though DC trace is not active on the terminal.

4. An asterisk (*) in this list is a wildcard character, meaning that any character can replace the asterisk.

D xx Device-Dependent Module activity (DDM)

DDxx Output processing by DFSCOFC0

DSIM SIMLOGON attempt of a dynamic terminal

ESIM SIMLOGON error for a dynamic terminal

FERR MFS-block fetch error

FESx Front-end switch user exit routine activity

FEXT Before field edit exit routine

FMTx Message Format Service activity (MFS)

HCSW
XRF class 1 takeover trace. This log record is written for XRF class 1 terminals during takeover, even though DC trace is not active on the terminal.

ICLR Message router activity

MTRP Block verification error

SDC1 DDM SDC output read error

SDC2 DDM SDC message reread error

SEXT Before segment edit exit routine

SGNX Signon user exit routine failure

SPCL Close spool data set

SPOP Open spool data set

SPRE Read spool data set

SPWR Write spool data set

TRCE Non-SNA 3270 error

VTPO Non-posting of ECB trace (DFSVTPO0)

Exception: MSC has its own analyzer module and entry types.

Table 38 shows the types of data communication (DC) trace records and what each trace record contains. Some of the acronyms used in the table are:

SEG Segment (DECAREA buffer)

MFS MFS input work/MFS output work

QBUF Queue buffer

IOPUF
TP buffer

S25 Save area 2-5

SALL Save area all

Table 38. DC Trace Records

Trace ID	Function	Traced by	When Traced or /TRACE Option	What Is Traced
A01	Process input. [1]	DFSICIO0 [9]	ALL, DDM	CTB, CLB, CXB, CRB, CIB, CCB, QBUF, IOBUF, INPCNTS, OUTCNTS, EMHB [2]

Table 38. DC Trace Records (continued)

Trace ID	Function	Traced by	When Traced or /TRACE Option	What Is Traced
A02	Do read. [1]	DFSICIO0 [9]	ALL, DDM	CTB, CLB, CXB, CRB, IOBUF, EMHB [2]
A03	What is next.	DFSICIO0 [9]	ALL, DDM	CTB, CLB, CRB, CTT
A04	Get Next segment.	DFSICIO0 [9]	ALL, DDM	CTB, CLB, CNT
A05	Do write. [1]	DFSICIO0 [9]	ALL, DDM	CTB, CLB, CXB, CRB, CCB, IOBUF, EMHB [2]
A06	After good write.	DFSICIO0 [9]	ALL, DDM	IOB, CTB, CLB, CXB, CRB, CCB
A07	After bad write. [1]	DFSICIO0 [9]	ALL, DDM	IOB, CTB, CLB, CRB, CCB, IOBUF, EMHB [2]
A08	Cancel message, do not DEQ.	DFSICIO0 [9]	ALL, DDM	CTB, CLB, CRB
A09	Generate system message. [1]	DFSICIO0 [9]	ALL, DDM	CTB, CLB, CRB, MFS
A10	Quiesce without stopping.	DFSICIO0 [9]	ALL, DDM	CTB, CLB, CRB, CCB
A11	Retrieve last DEQD message.	DFSICIO0 [9]	ALL, DDM	CTB, CLB, CNT, CRB
A12	Wait for ASYNC I/O or output ENQ.	DFSICIO0 [9]	ALL, DDM	CTB, CLB, CRB, CCB, IOBUF, EMHB [2]
AER1	Access method error.	DFSICIO0 [9]	Always	CTB, CLB, CNT, QBUF, SALL, CTT, PCB
AER2	Access method error. [3, 1]	DFSICIO0 [9]	Always	IOB, CTB, CLB, CNT, CXB, CRB, CIB, CCB, QBUF, IOBUF, SALL, CTT, PCB, EMHB [2]
C00	Get queue buffer.	DFSICIO0 [9]	ALL, MFS	CTB, CNT, CIB, SALL
C01	Reposition queue buffer.	DFSICIO0 [9]	ALL, MFS	CTB, CNT, CIB, SALL
C02	Get Next.	DFSICIO0 [9]	ALL, MFS	CTB, CNT, CIB, SALL
C03	DEQ output.	DFSICIO0 [9]	ALL, MFS	CTB, CNT, CIB, SALL
C04	Place output back in queue.	DFSICIO0 [9]	ALL, MFS	CTB, CNT, CIB, SALL
C05	Find output.	DFSICIO0 [9]	ALL, MFS	CTB, CNT, CIB, SALL
C06	Get new output message or QMGR call.	DFSICIO0 [9]	ALL, MFS	CTB, CNT, CIB, SALL
C07	Free input buffer.	DFSICIO0 [9]	ALL, MFS	CTB, CNT, CIB, SALL
C08	Get output buffer.	DFSICIO0 [9]	ALL, MFS	CTB, CNT, CIB, SALL
C09	User output edit.	DFSICIO0 [9]	ALL, MFS	CTB, CNT, CIB, SALL
C10	Call queue MGR.	DFSICIO0 [9]	ALL, MFS	CTB, CNT, CIB, SALL
C11	Get DDM work buffer.	DFSICIO0 [9]	ALL, MFS	CTB, CNT, CIB, SALL
C12	Free DDM work buffer.	DFSICIO0 [9]	ALL, MFS	CTB, CNT, CIB, SALL
C13	Free receive-any buffer.	DFSICIO0 [9]	ALL, MFS	CTB, CNT, CIB, SALL
CIO2	DDM SDC read output	DFSCIO20	ALL DDM	copy ctl blk list from CVCT entry
CIO3	DDM SDC 'wash' output	DFSCIO30	ALL DDM	copy ctl blk list from CVCT entry
CMEA	Before calling Message Control/Error exit.	DFSCMEI0		CTB, CLB, CRB, QBUF, IOBUF, INPCNTS, OUTCNTS, DDM, MSNB
CMEB	After calling Message Control/Error exit.	DFSCMEI0		CTB, CLB, CRB, QBUF, IOBUF, INPCNTS, OUTCNTS, DDM, MSNB
CMEI	Message Control/Error exit interface processing.	DFSCMEI0		CTB, CLB, CRB, QBUF, IOBUF, INPCNTS, OUTCNTS, DDM, MSNB
COFC	Let MFS edit output.	DFSICIO0 [9]	ALL, DDM	CTB, CLB, CNT, CRB, CIB, IOBUF, EMHB [2]

Table 38. DC Trace Records (continued)

Trace ID	Function	Traced by	When Traced or /TRACE Option	What Is Traced
CRTU	Output User Creation exit routine failure.	DFSCRTU0	Always	See notes[10]
CVCT	VTAM TRACE/ABORT. [1]	DFSCVCT0	ALL, DDM	CTB, CLB, CNT, CRB, IOBUF, CTT, INPCNTS, EMHB [2]
CVCV	XRF class 2 takeover. [1]	DFSCVCV0	Always	CLB, CTB, CTT, LLB, LTB, LXB, LU6WA, CNT, CRB, SPQB, CTC, MSNB, EMHB, IOBUF, DDM
D01	Write setup.	DFSICIO0 [9]	ALL, DDM	CTB, CLB, CNT, CRB, CIB, QBUF, S25
D02	Write interrupt. [1]	DFSICIO0 [9]	ALL, DDM	IOB, CTB, CLB, CRB, IOBUF, S25, EMHB [2]
D03	Read setup.	DFSICIO0 [9]	ALL, DDM	CTB, CLB, CNT, CRB
D04	Read interrupt. [1]	DFSICIO0 [9]	ALL, DDM	IOB, CTB, CLB, CRB, IOBUF, S25, EMHB [2]
D05	Cleanup.	DFSICIO0 [9]	ALL, DDM	IOB, CTB, CLB, CNT, CXB, CRB, CIB, CCB, MFS, QBUF, IOBUF, SALL, EMHB [2]
D07	LOGON. [1]	DFSICIO0 [9]	ALL, DDM	CTB, CLB, CNT, CRB
DD6M	Output build (MFS).	DFSCOFC0	ALL, DDM	CTB, CLB, CNT, CRB, CIB, SEG, MFS, IOBUF, S25, EMHB [2]
DD6S	Output build (Non-MFS).	DFSCOFC0	ALL, DDM	CTB, CLB, CNT, CRB, CIB, IOBUF, S25, EMHB [2]
DD8	Prepare for output.	DFSCOFC0	ALL, DDM	CTB, CLB, CNT, CRB, CIB, IOBUF, S25, EMHB [2]
DDM1	Write set up through COFC.	DFSCOFC0	ALL, DDM	CTB, CLB, CNT, CRB, CIB, MFS, IOBUF, S25, EMHB [2]
FERR	MFS block fetch error. [3]	DFSCFEO0	Always	CIB, CTT, MFSBPCA, MFSTRACE 4
FES1	Entry to front end switch user exit.	DFSICIO0 [9]		CTB, CLB, CNT, QBUF, S25
FES2	Exit from front end switch user exit.	DFSICIO0 [9]		CTB, CLB, CNT, QBUF, S25
FEXT [5]	Before field edit exit.	DFSCFEI0	MFS	CTB, CIB
FMT1	Return from DFSFEIO or unformatted input.	DFSICIO0 [9]	ALL, MFS	CTB, CLB, CIB, IOBUF, EMHB [2]
FMT2	MFS go to DFSFEIO formatted input.	DFSICIO0 [9]	ALL, MFS	CTB, CLB, CIB, IOBUF, EMHB [2]
FMT3	MFS complete process MSG segment.	DFSICIO0 [9]	ALL, MFS	CTB, CLB, CIB, MFS, QBUF
FMT4	Get next input.	DFSICIO0 [9]	ALL, MFS	CTB, CLB, CIB
FMT6	Clean up resources.	DFSICIO0 [9]	ALL, MFS	CTB, CLB, CIB
HCSW	XRF class 1 takeover. [1]	DFSHCSW0	Always	IOBUF, CNT, CRB, CTT, CTB, CLB
ICLR	Message router.	DFSICLR0	Always	CTB, CLB, CTT, PCB
MTRP [8]	Block verification error.	DFSCFEO0		CLB, CIB, MFS, CTT
MTRP [7]	Block verification error.	DFSCFEI0		CLB, CIB, MFS, CTT
SDC1	DDM SDC read error	DFSCIO20	ALL DDM	copy ctl blk list from CVCT entry
SDC2	DDM SDC reread error	DFSICIO4	ALL DDM	copy ctl blk list from CVCT entry
SEXT [6]	Before segment edit exit.	DFSCFEI0	MFS	CTB, CIB
TRCE	Non-SNA 3270 error.	DFSDN130, DFSDN140, DFSDS060	Always	IOB, CTB, CLB, S25, CTT

Table 38. DC Trace Records (continued)

Trace ID	Function	Traced by	When Traced or /TRACE Option	What Is Traced
VTPO	Rejected posting of ECB.	DFSVTPO0	ALL, DDM	See notes[11]

Notes:

1. See “Diagnosing Line and Terminal Problems” for more information on this trace code.
2. Fast Path EMHB buff traces (if present) with I/O buffers
3. Module return code saved in CLBTEMP4
4. Return codes from DFSFFRH0 (block fetch), MFSTRACE (when in MFSTEST) or MFSBPCA (when not in MFSTEST); MFSTRACE=MFSTEST trace parms, MFSBPCA=MFS Buffer Pool Control Area:

   Offset in Hex

   0 Current pool space in use

   4 Maximum space used

   5 Status flag

X'80'	I/O active for a task
X'40'	Task(s) queued for I/O
X'20'	A task dequeued and posted

   9 Error status

X'BB'	BLDL error
X'FF'	READ error

   A Block name for BLDL error

   10 BLDL return code on error

   12 Sense from read error

   14 CSW status from read error

   16 Block name for read error

   20 List for BLDL macro
5. Besides CIB and CTB:

   PARMLIST
   Parameter list to be passed to EXIT

   FIELD Field data before exit
6. Besides CIB and CTB:

   PARMLIST
   Parameter list to be passed to EXIT

   SEGMENT
   Segment data before exit
7. SEXT is logged if TRAP 1 is set by /TRACE and a buffer overwrite occurs.
8. MTRP is logged if TRAP 1 is set by /TRACE and a buffer overwrite occurs. In addition to the blocks, the DIF/DOF, MID/MOD, MFBP, and FRE are traced. If in output, R9 is also traced.
9. The MSNB control block content will be traced by DFSICIO0 if the /DEQ LTERM, /DEQ NODE, or the /DEQ MSNAME command is entered with the PURGE or PURGE1 keywords.
10. The CRTU trace entry is mapped in “Format of 6701 Log Record with CRTU Identifier” on page 261.
11. The VTPO trace entry is mapped in “Format of 6701 Log Record with VTPO Identifier” on page 262.

Diagnosing Line and Terminal Problems

The trace records with the following identifier are useful in diagnosing line and terminal problems:

A01 TERMINAL INPUT READY FOR IMS PROCESSING

I TP BUF
Contains input "device segment" 6 to 36 bytes from the beginning of the buffer. The data is preceded by a 2-byte length and 2 bytes of zeros.

A02 PRIOR TO ISSUING VTAM OR BTAM I/O REQUEST. (LOGICAL READ)

CLB For BTAM, the first 12 words are the BTAM DECB. See BTAM documentation. The BTAM operation type is at offset X'04'. For remote 3270:

X'0001'
Special poll (read sense/status)

X'0401'
Read initial (general poll)

X'0082'
Write initial

X'0084'
Write continue

Offset X'0C' contains the address in TP BUF to read into or write from.

I TP BUF
The input TP buffer contains data to be written if this is an output operation. For VTAM nodes, the RPL begins at offset X'08'.

A05 PRIOR TO ISSUING VTAM OR BTAM I/O REQUEST. (LOGICAL WRITE)

CLB Refer to the information for record A02.

O TP BUF
The output TP buffer contains data to be written if this is an output operation. For VTAM nodes, the RPL begins at offset X'08'.

A07 GENERATE 'UNABLE TO RECEIVE/OUTPUT' MESSAGE

See the preceding D02 or D04 record for the cause.

A09 GENERATE ERROR MESSAGE

See the preceding D02, D04, or D07 record for the cause.

AER2 SHOULD NOT OCCUR ERROR HAS OCCURRED

CLB Offset X'3E' contains the error message number in hexadecimal. All available control blocks and buffers are logged. This record is produced even if the trace is not set on.

CRTU OUTPUT USER CREATION EXIT ROUTINE FAILURE

See section "Format of 6701 Log Record with CRTU Identifier" on page 261.

CVCT VTAM DEVICE SUPPORT TRACE

CLB Normally offset X'1C' contains the complemented IMS message key of an IMS master terminal message. All available control blocks and buffers are logged. This record is produced even if the trace is not set on.

I TP BUF of O BUF
The VTAM RPL begins at offset X'08'.

CVCV XRF CLASS 2 TAKEOVER TRACE

This log record is written for XRF class 2 terminals during takeover, even though DC trace is not active on the terminal. This record can be used to diagnose subsequent session failures when used in conjunction with CVCT records.

D02 BTAM OR VTAM HAS POSTED I/O COMPLETE. (LOGICAL WRITE INTERRUPT)

CLB For BTAM, the first 12 words are the BTAM DECB. See BTAM documentation.

Offset X'00' =
Post code

X'7F' for BTAM = normal completion

X'40' for VTAM = normal completion

Other key fields are DECFLAGS and DECERRST. For VTAM, key fields are CLBVFLAG and CLBLOST.

IOB The BTAM IOB contains CCWs and CSW. Refer to *MVS/ESA Data Areas* for the format of the control blocks.

O TP BUF
The output TP buffer may contain sense/status information for remote 3270 if the last BTAM operation was specific poll. For VTAM nodes, the VTAM RPL begins at offset X'08'.

D04 BTAM OR VTAM HAS POSTED I/O COMPLETE. (LOGICAL READ INTERRUPT)

CLB Refer to the information for record D02.

IOB Refer to the information for record D02.

I TP BUF
The input TP buffer contains data read from the terminal if the last operation was a read or poll. For VTAM nodes, the RPL begins at offset X'08'.

D07 DEVICE DEPENDENT INITIALIZATION/TERMINATION

CLB Refer to information for record D02.

O TP BUF
The VTAM RPL begins at offset X'08'.

HCSW
XRF CLASS 1 TAKEOVER TRACE

This log record is written for XRF class 1 terminals during takeover, even though DC trace is not active on the terminal. This record can be used to diagnose subsequent session failures when used in conjunction with CVCT records.

VTPO REJECTED POSTING OF ECB

See section “Format of 6701 Log Record with VTPO Identifier” on page 262.

Format of 6701 Log Record with CRTU Identifier

The following example provides a map of the formatted CRTU log record.

Table 39. Map of formatted CRTU log Record

Offset	Hex Code	Description
+0	H	Length of Buffer
+2	XL5	Internal use
+7	X	DFSCRTU0 Return Code (see below)
+8	XL64	Internal use
+48	CL8	Input Lterm Name
+50	XL52	Internal use

DFSCRTU0 Return Codes (decimal): The following are the return codes and their meanings.

```
 4  'ENVIRONMENT' INCORRECT (i.e., NO ETO,
    NO DFSINSX0 WITH SHARED QUEUES).

16  DUPLICATE LTERM/SMB NAME.

20  NO USER DESCRIPTOR COULD BE LOCATED
    FOR USE IN CREATING USER STRUCTURE.

24  INVALID INPUT LTERM NAME.

28  DFSINSX0 REJECTED USER-CREATION REQUEST.

32  STORAGE COULD NOT BE OBTAINED TO CREATE
    USER STRUCTURE.

36  STATIC USER ALREADY EXISTS.

40  INSERT EXIT PRAMETER ERROR: INVALID LTERM
    NAME, BAD FORMAT.

48  AVAILABLE.

52  LATCHING ERROR OCCURRED.
```

Format of 6701 Log Record with VTPO Identifier

| If an APPC or OTMA message is discarded because of a send type error, IMS does not log a type 6701–CMEA/CMEB record for the error. It does log type 6701–CMEA/CMEB records for errors related to other devices, though. The lack of type 6701–CMEA/CMEB records makes debugging for the User Message Control/Error exit routine (DFSCMUX0) difficult.

Table 40. VTCB Posting in DFSVTPO0

Offset	Hex Code	Description
+0	X	Function code
	X'00'	VTCB is to be posted
	X'04'	VTCB is to be released
	X'08'	Check if ACB can be closed
	X'0C'	Delete a VTCB
	X'10'	Stacked logon for static CLB
	X'14'	NSEXIT for static CLB
	X'18'	NSEXIT for dynamic CLB
	X'1C'	LOSTERM for static CLB
	X'20'	LOSTERM for dynamic CLB
+1	X	Type of checking RQD for post
	X'04'	Post if node is active
	X'08'	Post if node not active
	X'0C'	Post if idle and not active
	X'10'	Hard post the node
	X'14'	Post an MSC LLB

Table 40. VTCB Posting in DFSVTPO0 (continued)

Offset	Hex Code	Description
+2	X	Conditional data for posting
	X'80'	Type is ISC parallel session
	X'40'	Type is MSC LLB
	X'20'	Z-NET cancel in progress
		On detection of an error, this byte contains one of the following reject codes:
	X'01'	VTCB not specified
	X'02'	Inspection failed—check subcode
	X'03'	Node not idle
	X'04'	RQR failed—check subcode
	X'05'	Node active—check subcode
	X'06'	Node not alive—check subcode
	X'07'	Invalid request
	X'08'	MSC link already posted
	X'09'	MSC send outstanding
	X'0A'	Node already dispatched
	X'20'	No VTCB to delete
	X'30'	CINIT rejected by PLU (NSX)
	X'31'	VTAM error (NSX)
	X'40'	Stacked logon procedure failure
+3	X	Posting-rejection subcode[1]
	X'01'	Node already dispatched (RQR)
	X'02'	Node already posted (RQR)
	X'03'	Unpostable I/O (RQR)
	X'04'	Clear issued (RQR)
	X'05'	Inact performed (RQR)
	X'01'	SPQB not found (INSPECT)
	X'02'	No match on CLB ADDR (INSPECT)
	X'03'	VOPEN not on (INSPECT)
	X'04'	VTCB not found by scan (INSPECT)
	X'05'	No match on VTCBs (INSPECT)
	X'06'	CIDs don’t match (INSPECT)
	X'07'	VOPEN not set (INSPECT)
	X'08'	Temporary VTCB (INSPECT)
	X'01'	No /idle node CMD (POSTRTN)
	X'02'	Node inoperable (POSTRTN)
	X'03'	Node dispatched (POSTRTN)
	X'04'	Line already posted (POSTRTN)
	X'05'	V2SND is set (POSTRTN)
	X'06'	Not XRF sync mode (POSTRTN)
	X'07'	Not SCIP exit with clear (POSTRTN)
|	X'08'	SCIP exit bindrace done (POSTRTN)
+4	0F	Post code
+4	X	NSEXIT flag
	X'80'	Cleanup RU
	X'40'	Notify RU
+5	X	NSEXIT type for CLBLOST
+6	X	Reason code for CTBRTERM
+7	X	Notify reason code
+8	F	VTCB address
+C	CL8	VTAM node name
+14	F	CID

Table 40. VTCB Posting in DFSVTPO0 (continued)

Offset	Hex Code	Description
+18	CL8	SPQB name if parallel session
+20	0F	CLBNCID for a stacked logon
+20	F	Sense data (NSEXIT)

Note:

1. This byte contains an additional “qualifier” subcode.

Example of DC Trace Output

```
INTERNAL TRACE RECORD          ID = D 07  SEGNO=00  RECNO = 0000013B  TIME  08.40.59.68   DATE  88.047
CLB
 02248078 000000   40D6D7D5 00000000  00000000 00000000   00000000 00000000  00000000 00000000   * OPN............................*
 02248098 000020   00000000 00000000  C2F0D7F0 F6404040   00000100 022480FC  00000000 00000000   *........B0P06   ................*
 022480B8 000040   00000000 00000000  00010000 00000000   022480FC 80000000  00000000 00000000   *................................*
 022480D8 000060   00000000 00000000  00000000 00040000   00000000 00000000  40000000 00000000   *........................ .......*
 022480F8 000080   00000000                                                                      *....                           *
CTB
 022480FC 000000   00038CC8 02248078  00000000 000B2000   00000000 082A0000  0000FFFF 0003614C   *...H........................../<*
 0224811C 000020   00000000 00000000  022481C4 00004040   40404040 40400000  00000000 00000000   *..........AD..       ..........*
 0224813C 000040   00000000 00000000  00000000 00000000   00000000 00000000  00000000 00000000   *................................*
 0224815C 000060                      SAME AS ABOVE
INP CNTS
 0003614C 000000   00000000 00000000  00000000 00000000   00000000 00820084  00000000 C2F0D7F0   *....................B.D....B0P0*
 0003616C 000020   F6404040 00000001  022480FC 000371F0   FFFF0909 00000000  00000000            *6   ...........0............    *
NEXT CNT
 000371F0 000000   00000000 00000000  00000000 00000000   00000000 00820084  00000000 D4E3D6D4   *....................B.D....MTOM*
 00037210 000020   C1E2E340 00000001  022480FC 00000000   FFFF0909 00000000  00000000            *AST ........................    *
INTERNAL TRACE RECORD          ID = C 08  SEGNO=00  RECNO = 0000013C  TIME  08.40.59.84   DATE  88.047
CLB
 02248078 000000   40D6D7D5 00000000  00000000 00000000   00000000 00000000  00000000 00000000   * OPN............................*
 02248098 000020   00000000 00000000  C2F0D7F0 F6404040   00000100 022480FC  00000000 00000000   *........B0P06   ................*
 022480B8 000040   00000000 00000000  00010000 00000000   022480FC 80000000  00000000 00000000   *................................*
 022480D8 000060   00000000 00000000  00000000 00040000   00000000 00000000  40000000 00000000   *........................ .......*
 022480F8 000080   00000000                                                                      *....                           *
CTB
 022480FC 000000   00038CC8 02248078  00000000 000B2000   00000000 082A0000  0000FFFF 0003614C   *...H........................../<*
 0224811C 000020   00000000 00000000  022481C4 00004040   40404040 40400000  00000000 00000000   *..........AD..       ..........*
 0224813C 000040   00000000 00000000  00000000 00000000   00000000 00000000  00000000 00000000   *................................*
 0224815C 000060                      SAME AS ABOVE
CIB
 022481C4 000000   40404040 40404040  00000000 00004040   40404040 00000000  00000000 00000000   *        ......     ............*
 022481E4 000020   00000000 02C70000  00000000 00000000   00004040 40404040  40400000 40404040   *.....G............        ..    *
 02248204 000040   40404040 00004040  40404040 00000000   00000000 00180050  00000000 00000000   *    ..       ............&;.......*
 02248224 000060   40404040 40404040                                                             *                               *
INTERNAL TRACE RECORD          ID = A 05  SEGNO=00  RECNO = 0000013D  TIME  08.40.59.86   DATE  88.047
CLB
 02248078 000000   00000000 00000000  00000000 02235008   00000000 00000000  00000000 00000000   *..............&;................*
 02248098 000020   00000000 00000000  C2F0D7F0 F6404040   10000100 022480FC  00000000 00000000   *........B0P06   ................*
 022480B8 000040   01C80000 00000000  00010000 00000000   022480FC 80000000  00000000 00000000   *.H..............................*
 022480D8 000060   00000000 02235000  00000000 00000000   00000000 00000000  40000000 00000000   *......&;................ .......*
 022480F8 000080   00000000                                                                      *....                           *
CTB
 022480FC 000000   00038CC8 02248078  00000000 000B2000   00000000 082A0000  0000FFFF 0003614C   *...H........................../<*
DFSERA30  -  FORMATTED LOG PRINT                                                                                     PAGE  009
 0224811C 000020   00000000 00000000  022481C4 00004040   40404040 40400000  00000000 00000000   *..........AD..       ..........*
 0224813C 000040   00000000 00000000  00000000 00000000   00000000 00000000  00000000 00000000   *................................*
 0224815C 000060                      SAME AS ABOVE
O TP BUF
 02235000 000000   01C80088 00000000  00201670 00000000   00000000 00000000  00001000 00800000   *.H.H............................*
 02235020 000020   0002FD14 00000000  00000000 02235088   20800000 00000000  00000000 00000000   *..............&H...............*
 02235040 000040   10308050 00000000  80800000 44000000   00000000 00000000  00000000 00000000   *...&;...........................*
 02235060 000060   00000000 00000000  80008010 00000000   00000000 00000000  00000000 00000000   *................................*
 02235080 000080   00000000 00440000  D0000040 00000000   02248078 C2F0D7F0  F6404040 D9C5C3D6   *........... ........B0P06   RECO*
 022350A0 0000A0   D9C44040 00000000  00000000 41080002   00000001 00000000  00000000 00000000   *RD  ............................*
 022350C0 0000C0   00000000 00000000  00000000 00000000   00000000 00000000  00000000 00000000   *................................*
 022350E0 0000E0 TO 02235160 000160  SAME AS ABOVE
 02235180 000180   00000000 00000000  00000000 00000000   FF00403F C181AA55  01900000 00000000   *.................. .AA..........*
 022351A0 0001A0   00000000 00000000  00000000 00000000   00000000 00000000  00000000 00000000   *................................*
 022351C0 0001C0   00000000 00000000                                                             *........                       *
INTERNAL TRACE RECORD          ID = D 07  SEGNO=00  RECNO = 0000013E  TIME  08.41.00.43   DATE  88.047
CLB
 02248078 000000   40000000 00000000  00000000 02235008   00000000 00000000  00000000 00000000   * .............&;................*
 02248098 000020   00000000 00000000  C2F0D7F0 F6404040   10020100 022480FC  00000000 00050007   *........B0P06   ................*
 022480B8 000040   0840598F 0088047F  00010000 00000000   022480FC 80000000  00000000 00000000   *. ...H."........................*
 022480D8 000060   00000000 02235000  00000000 00000000   00000000 00000000  40000000 00000000   *......&;................ .......*
 022480F8 000080   00000000                                                                      *....                           *
```

Figure 105. Data Communication (DC) Trace Records (Part 1 of 2)

```
CTB
 022480FC 000000   00038CC8 02248078  00000000 000B2000   00000000 082A0000  0000FFFF 0003614C   *...H............................/<*
 0224811C 000020   00000000 00000000  022481C4 00004040   40404040 40400000  00000000 00000000   *..........AD..        ..........*
 0224813C 000040   00000000 00000000  00000000 00000000   00000000 00000000  00000000 00000000   *................................*
 0224815C 000060                     SAME AS ABOVE
INP CNTS
 0003614C 000000   00000000 00000000  00000000 00000000   00000000 00820084  00000000 C2F0D7F0   *.....................B.D....B0P0*
 0003616C 020020   F6404040 00000001  022480FC 000371F0   FFFF0909 00000000  00000000            *6   ...........0............    *
NEXT CNT
 000371F0 000000   00000000 00000000  00000000 00000000   00000000 00820084  00000000 D4E3D6D4   *.....................B.D....MTOM*
 00037210 000020   C1E2E340 00000001  022480FC 00000000   FFFF0909 00000000  00000000            *AST ........................    *
```

Figure 105. Data Communication (DC) Trace Records (Part 2 of 2)

Diagnosing Problems in the Message Requeuer

The message requeuer processor module (DFSQMRQ0), which is part of the IMS Transaction Manager (TM) component, provides diagnostics for diagnosing errors while running the IMS/ESA Message Requeuer (MRQ) licensed program (5655-038). Although problems can be diagnosed separately in the MRQ product via SCRAPLOG records and in the IMS message requeuer processor module via 6701-MRQE diagnostic records, MRQ and the message requeuer processor work together to allow the requeuing to IMS message queue data sets of any messages that might have been lost due to an IMS cold start or other reasons. Therefore, this section describes the MRQ licensed program and its associated SCRAPLOG diagnostic records, as well as the IMS message requeuer processor module and its associated 6701-MRQE diagnostic records.

In this section, information concerning SCRAPLOG records applies to SCRAPSEL and SCRAPCAN records, as well. The SCRAPSEL, SCRAPCAN, and SCRAPLOG data sets are generated by the FMQSELCT, FMQCANCL, and FMQINSRT modules of MRQ, respectively. These data sets are identical in both format and function.

The diagnostics described in this section can help you if you are experiencing problems with a message being requeued. Whenever a message you are trying to requeue is rejected, MRQ prints an insert report telling you what messages were not requeued to a given LTERM.

For a schematic of how the message requeuer function works in the product and where it fits into the IMS Transaction Manager and System Services, see Figure 106 on page 267.

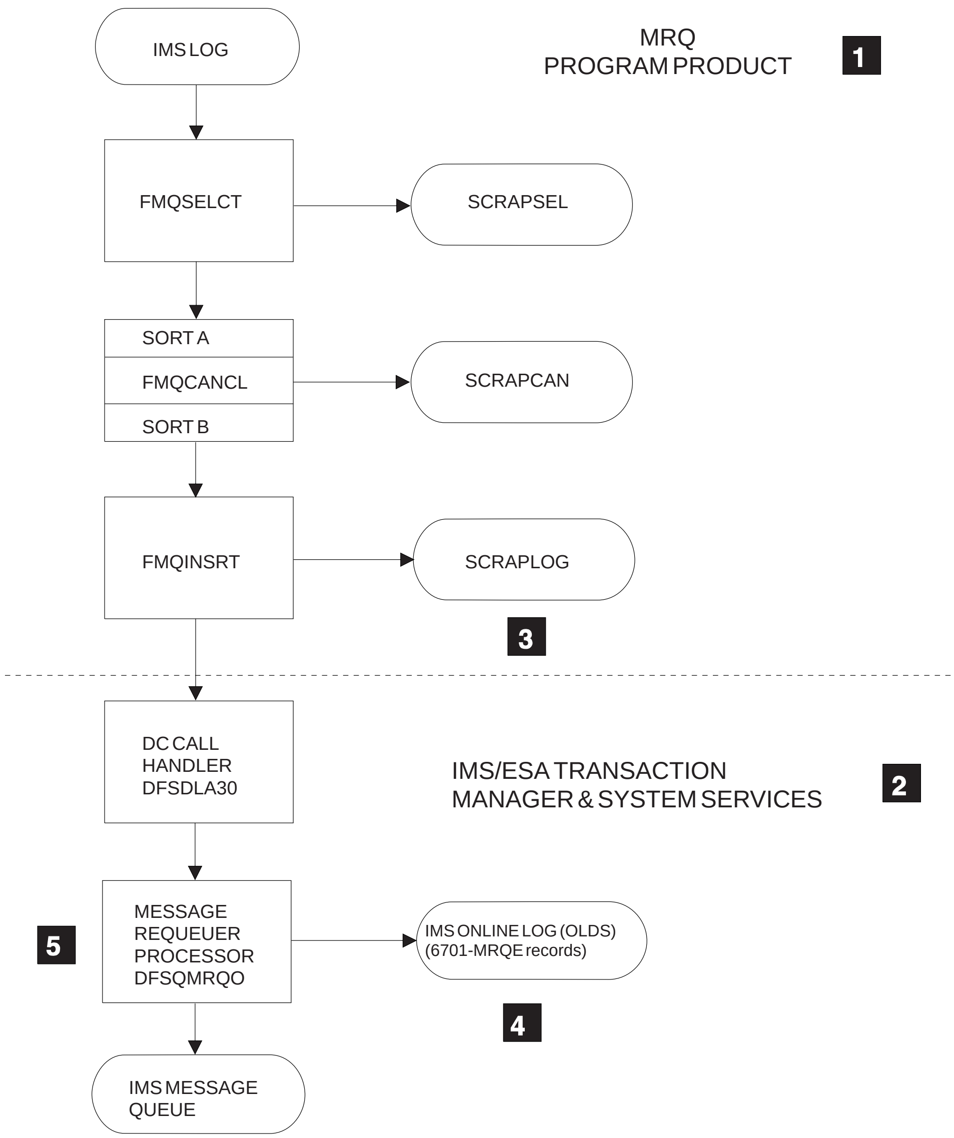


Figure 106. Relationship of Message Requeuer Licensed Program to IMS Transaction Manager & System Services

The Message Requeuer Program Product

The MRQ program product (**1** in Figure 106) consists of 3 modules, as follows:

Module	Function
FMQSELCT	Selects messages for requeuing
FMQCANCL	Analyzes and cancels messages
FMQINSRT	Inserts messages back to IMS for requeuing to message queue data sets

The MRQ module FMQSELCT selects messages to be requeued from the IMS online log data sets (OLDSs) or system log data sets (SLDSs). Based on recovery modes, the messages are analyzed, sorted, and collected. Some messages might be canceled by the FMQCANCL module. The messages to be reinserted are passed to the FMQINSRT routine for insertion into the IMS message queues.

The FMQSELCT module, FMQCANCL module, and the sort utilities run as stand-alone MVS jobs or steps. The FMQINSRT routine runs as an IMS BMP, and inserts the messages to the IMS data communication (DC) call handler (module DFSDLA30). The DC call handler calls the message requeuer processor (module DFSQMRQ0) to reinsert and requeue the messages.

The FMQINSRT module uses an alternate modifiable teleprocessing control block (ALT TPPCB) and an application interface block (AIB) to issue ISRT and PURG calls to IMS TM to requeue the messages.

The Message Requeuer Processor Module (DFSQMRQ0)

When the message requeuer processor (**5** in Figure 106 on page 267) in the IMS/ESA Transaction Manager (**2** in Figure 106) detects an error while reinserting a message, the following diagnostics are provided:

1. The TPCBSTAT code in the MRQ alternate PCB is set to ’MR’.
2. The application interface block (AIB) return code (AIBRETRN) is set to X'000000F0'.
3. The AIB reason code (AIBREASN) is set to a unique hexadecimal value for each type of error. For a list of AIBREASN codes, see Appendix B, “AIBREASN Codes for Message Requeuer Errors,” on page 449.
4. The TPCB, AIB, I/O area (containing the message being inserted) and other pertinent control blocks are logged to the OLDS (**4** in Figure 106) in the form of a type 6701-MRQE log record. (For more information on these records see “Using 6701-MRQE Diagnostic Records” on page 270.)
5. The TPCBSTAT, AIBRETRN, and AIBREASN codes are passed back to the FMQINSRT program.
6. The FMQINSRT program records the error in an MRQ prefix and writes the MRQ prefix and the message being inserted to the MRQ SCRAPLOG data set. For more information on how to print this SCRAPLOG record, see “Sample JCL for Printing SCRAPLOG Records” on page 270.
7. The FMQINSRT routine keeps counts of messages discarded and groups these by reason code and destination. These groupings are shown in a SYSOUT report when the FMQINSRT BMP finishes executing. The SYSOUT report can be used, in combination with SCRAPLOG data sets and 6701-MRQE records logged to the IMS log data set, to analyze the error.

   When the error is corrected, it might be possible to rerun the FMQINSRT program (using the SCRAPLOG data set as input) and reinsert the messages that failed.

   Related Reading: See *IMS Message Requeuer Program Description/Operations Manual* for more details about the SCRAPLOG data set, SYSOUT reports, and PCB/AIB error codes.

As shown in Figure 106 on page 267, after the message requeuer processor module detects an error, both SCRAPLOG records and 6701-MRQE diagnostic records are written. You need details about both of these types of records to diagnose problems. For details on SCRAPLOG diagnostic records, see Using SCRAPLOG Diagnostic Records. For details on 6701-MRQE diagnostic records, see “Using 6701-MRQE Diagnostic Records” on page 270.

Using SCRAPLOG Diagnostic Records

As part of your diagnosis process for problems with the Message Requeuer, you use SCRAPLOG records (**3** in Figure 106 on page 267). This section provides the following details:

- An explanation of SCRAPLOG records
- A sample record
- Information about which key fields are of special interest
- Instructions for printing SCRAPLOG records

By analyzing SCRAPLOG records, you can sometimes determine that an LTERM (to which messages were to be requeued) doesn’t exist. In this case, you can fix the problem and rerun the job so the messages will be requeued.

SCRAPLOG Records

The SCRAPLOG record consists of a 66-byte (hexadecimal 42) MRQ prefix, followed by the actual message being inserted. The actual message is either a 4002 record (that is, a message from a DUMPQ or SNAPQ checkpoint) or a 01 (input) or 03 (output) message record. The record shown in Figure 107 is a 01 input record. The LOGREC type (4002, 01, or 03) is at offset 4 in the MRQ prefix segment and at offset 46 (which is offset 4 in the scrapped record).

Sample Record Written to the SCRAPLOG by FMQINSRT

Figure 107 is a hexadecimal dump of a record written to the SCRAPLOG data set by the FMQINSRT routine.

```
01 RECORD
 00000000 000000   00E40000 01000000  08000005 D3E3C5D9   D4F44040 C5E3D9C1  D5F1F940 E3C5D9D4   *............LTERM4  ETRAN19 TERM*
 00000020 000020   F4404040 0092318F  0944105F 00000010   81041000 0000D4D9  000000F0 00001084   *4   ..................MR........*
 00000040 000040   020100AC 000001D0  81100800 00050800   000500A8 80000040  8180C000 0000E3C5   *..............................TE*
 00000060 000060   D9D4F440 40400001  00000000 00000092   318F0944 105FD3E3  C5D9D4F4 4040C5E3   *RM4   ................LTERM4  ET*
 00000080 000080   D9C1D5F1 F9400000  00000000 0000C4C6   E2D4D6F2 40400040  82000000 00090000   *RAN19 ........DFSMO2  ..........*
 000000A0 0000A0   00000000 0001C5E3  D9C1D5F1 F940D3E3   C5D9D4F4 40400201  014C0000 00000000   *......ETRAN19 LTERM4............*
 000000C0 0000C0   00000000 00000000  00000000 00000000   00000000 00000014  8300D3E3 C5D9D4F4   *..........................LTERM4*
 000000E0 0000E0   40404040 40404040  40400000 0000                                              *         ....                 *
```

Figure 107. Sample SCRAPLOG Record Written by FMQINSRT

Key Fields of SCRAPLOG Records and Their Offsets

Table 41 shows some key fields of the MRQ records and their offsets, with values taken from the record shown in Figure 107.

Table 41. Key Fields of MRQ Diagnostic Records and Their Offsets

Offset	Length	Value	Description
04	01	01	Log Code = 01 = IMS input message
08	04	08000005	DRRN of the message read from the message queue where it was recovered. This is useful in tracing where the message came from, if necessary.
0C	08	LTERM4	Source = input LTERM name
14	08	ETRAN19	Dest = destination TRANCODE name
1C	08	TERM4	LUNAME, for LU6.2 or VTAM
24	04	0092318F	Date = date of message
28	04	0944105F	Time = time of message
36	02	MR	TPCBSTAT = MR = DFSQMRQ0 detected an error
38	04	000000F0	AIBRETRN = DFSQMRQ0 detected an error
3C	04	00001084	AIBREASN = unique reason code for the message being discarded (scrapped). 1084 indicates that the message is nonrecoverable (that is, specify `INQUIRY=NORECOV` on the TRANSACT macro for `TRAN CODE=ETRAN19`).
40	01	02	Destination system ID for MSC
41	01	01	Source system ID for MSC
42	Variable	Variable	Start of the 01 or 03 log record that was scrapped. This area maps to the 6701-MRQE I/O AREA starting at offset 24.
46	01	01	Log Code = 01 = IMS input message

Sample JCL for Printing SCRAPLOG Records

Figure 108 shows sample JCL that you can use to print SCRAPLOG records. You use these SCRAPLOG records to help diagnose problems with the Message Requeuer.

```
//SCRAPPRT  JOB
//*  PRINT FMQSELCT SCRAPSEL
//JOBLIB  DD DISP=SHR,DSN=IMS610.RESLIB
//SELECT  EXEC PGM=DFSERA10,REGION=512K
//SYSPRINT  DD SYSOUT=A
//SYSUT1 DD DSN=MRQ.SCRAPSEL,DISP=SHR
//SYSIN     DD *
CONTROL  CNTL
OPTION   PRINT E=DFSERA30
END
/*
//CANCEL  EXEC PGM=DFSERA10,COND=EVEN,REGION=256K
//*  PRINT FMQCANCL SCRAPCAN
//SYSPRINT  DD SYSOUT=A
//SYSUT1 DD DSN=MRQ.SCRAPCAN,DISP=SHR
//SYSIN     DD *
CONTROL  CNTL
OPTION   PRINT E=DFSERA30
END
//INSERT  EXEC PGM=DFSERA10,COND=EVEN,REGION=256K
//*  PRINT FMQINSRT SCRAPLOG
//SYSPRINT  DD SYSOUT=A
//SYSUT1 DD DSN=MRQ.SCRAPLOG,DISP=SHR
//SYSIN     DD *
CONTROL  CNTL
OPTION   PRINT E=DFSERA30
END
/*
```

Figure 108. Sample JCL for Printing SCRAPLOG Records

You need to use your SCRAPLOG records in combination with 6701-MRQE records to effectively diagnose MRQ problems.

Using 6701-MRQE Diagnostic Records

This section provides the following details about 6701-MRQE diagnostic records (**4** in Figure 106 on page 267):

- An explanation of 6701-MRQE diagnostic records
- A sample record
- Sample JCL for printing a record
- Control blocks logged at time of error and their mapping macros
- Some key fields to look for when diagnosing using 6701-MRQE records
- Some normal and abnormal errors associated with 6701-MRQE records

6701-MRQE Diagnostic Records

An IMS error detected while MRQ is requeuing messages results in the logging of a 6701-MRQE diagnostic record. The message being requeued is then discarded (written to the SCRAPLOG), and the MRQ BMP (FMQINSRT) proceeds on to the next message. Each type of error is accompanied by a unique reason code that is set in the application interface block reason code field (AIBREASN). For a list and explanations of AIBREASN codes, see Appendix B, “AIBREASN Codes for Message Requeuer Errors,” on page 449.

When the FMQINSRT step completes, a report of messages scrapped and grouped by reason code is produced. A report of messages scrapped and grouped by destination name is also produced. See *IMS Message Requeuer Program Description/Operations Manual* for an explanation of these reports.

Data mapping in the sample 6701-MRQE record shown in Figure 109 may vary from that for your release level of IMS. See the mapping macros listed in Table 43 on page 274 for your IMS system for the correct offsets for your release level.

Sample 6701-MRQE Record

```
INTERNAL TRACE RECORD           ID = MRQE  SEGNO=00  RECNO = 000000AC  TIME  16.48.16.30   DATE  92.321
PCB
 02CC90F0 000000   00300038 00020048  40404040 00000000   004DA054 00000000  0000BFBC C1D3E3D7   *........    .....(..........ALTP*
 02CC9110 000020   C3C2F0F1 00000000  00000000 00000000   00000000 02CC90F0  40404040 40404040   *CB01..................0        *
 02CC9130 000040   0100D4D9 00000000  00000000 00000000   00000000 03052C68  00000000 00000000   *..MR............................*
 02CC9150 000060   00000000 00000000                                                             *........                       *
AIB
 02BEBD18 000000   C4C6E2C1 C9C24040  00000080 40404040   40404040 C1D3E3D7  C3C2F0F1 C5E3D9C1   *DFSAIB  ....        ALTPCB01ETRA*
 02BEBD38 000020   D5F1F940 00000000  00000000 00CC0000   00000000 00000000  00000000 00000000   *N19 ............................*
 02BEBD58 000040   000000F0 00001084  00000000 0000BFF4   03000000 00000000  00000000 00000000   *...0...D.......4................*
 02BEBD78 000060   00000000 00000000  00000000 00000000   00000000 00000000  00000000 00000000   *................................*
I/O AREA
 02C33000 000000   00D50000 5BD4D9D8  D4E2C700 04100000   00000000 00000000  00000000 00000000   *....$MRQMSG.....................*
 02C33020 000020   00000000 00B50000  01D08110 08000005   08000005 00A88000  00408180 C0000000   *..........A..........Y... A.....*
 02C33040 000040   E3C5D9D4 F4404040  00010000 00000000   0092318F 0944105F  D3E3C5D9 D4F44040   *TERM4    .........K.....¬LTERM4  *
 02C33060 000060   C5E3D9C1 D5F1F940  00000000 00000000   C4C6E2D4 D6F24040  00408200 00000009   *ETRAN19 ........DFSMO2  . B.....*
 02C33080 000080   00000000 00000001  C5E3D9C1 D5F1F940   D3E3C5D9 D4F44040  0909014C 00000000   *........ETRAN19 LTERM4  ...<....*
 02C330A0 0000A0   00000000 00000000  00000000 00000000   00000000 00000000  00148300 D3E3C5D9   *..........................C.LTER*
 02C330C0 0000C0   D4F44040 40404040  40404040 00090300   C8C5D3D3 D6                            *M4          ....HELLO           *
PST
 02BEB060 000000   00000000 8C043848  00C4EE40 02F34900   02BD5858 04000004  00000000 00000000   *.........D. .3..................*
 02BEB080 000020   02D14040 00300038  00010018 40404040   40404040 004DA054  00000000 0000BF54   *.J  ........        .(..........*
 02BEB0A0 000040   C9D6D7C3 C2404040  00000000 00000000   00000000 00000000  02BEB084 D3E3C5D9   *IOPCB   ...................DLTER*
 02BEB0C0 000060   D4F44040 10004040  0092318F 0944105F   00000000 00000000  00000000 D4D9D8D7   *M4  ..  .K.....¬............MRQP*
 02BEB0E0 000080   E2C24040 40404040  40404040 00000000   00000000 00000000  03052C68 00000000   *SB          ....................*
 02BEB100 0000A0   00000000 00000000  00000000 00000000   00000000 00000000  00000000 00000000   *................................*
 02BEB120 0000C0   00000000 00000000  00000000 00000000   00110000 00000000  02C33840 02D0C610   *.........................C. ..F.*
 02BEB140 0000E0   004F3180 00000000  00000000 02C33000   02BEA544 00000000  00000000 00000000   *.|...........C....V.............*
 02BEB160 000100   00000000 02CC9020  00000000 00000000   00000000 00000000  D4D9D8D9 C5D7C340   *........................MRQREPC *
 02BEB180 000120   C2D4D740 40404040  00000000 00000000   00000000 00000000  00000000 00000000   *BMP     ........................*
 02BEB1A0 000140   00000000 00000000  00000000 00000000   00000000 00000000  00000033 00000011   *................................*
 02BEB1C0 000160   00000000 00000000  00000000 00000000   00000000 00000000  00000000 00000000   *................................*
 02BEB1E0 000180   00000000 00000000  00000000 00000000   00000000 00000000  00000000 02C338A8   *..............................C.Y*
 02BEB200 0001A0   00000000 00000000  C5C0FFFF 4A000000   C9E2D9E3 4140BA07  02CC90F0 00000000   *........E.......ISRT. .....0....*
 02BEB220 0001C0   00000001 00C4F5A8  00000000 00000000   00000000 00000000  00000000 00000000   *.....D5Y........................*
 02BEB240 0001E0   00000000 00000000  00000000 02000000   00000000 00000000  00000000            *............................    *
INTERNAL TRACE RECORD           ID = MRQE  SEGNO=01  RECNO = 000000AE  TIME  16.48.16.30   DATE  92.321
CONTINUE
 02BEB25C 0001FC   00000000 00000000  00000000 82B020AA   00000000 00000000  00000000 00000000   *............B...................*
 02BEB27C 00021C   00000000 00000000  00000000 00000000   02A783E0 00000000  00000000 00000000   *.................XC.............*
 02BEB29C 00023C   00000000 00000000  00000000 00000000   00000000 00000000  00000000 00000000   *................................*
 02BEB2BC 00025C                      SAME AS ABOVE
 02BEB2DC 00027C   00000000 C9E2D9E3  00000000 00000000   00000000 02CC90F0  00000000 00000000   *....ISRT...............0........*
 02BEB2FC 00029C   00000000 00000000  00000000 00000000   00000000 00000000  00000000 00000000   *................................*
 02BEB31C 0002BC   00000000 00000000  00000000 00000000   00000000 00000004  00F76480 00000000   *..........................7......*
 02BEB33C 0002DC   00000840 00000000  00010000 02BEB3D8   10000000 000000C0  00000000 00008000   *... ...........Q................*
 02BEB35C 0002FC   00008000 20008000  00000000 00000000   00000000 02BEA558  00000000 00000000   *......................V.........*
 02BEB37C 00031C   00000000 00000000  00000000 00000000   00000000 00000000  00000000 00000000   *................................*
 02BEB39C 00033C                      SAME AS ABOVE
```

Figure 109. Sample 6701-MRQE Record (Part 1 of 3)

```
DFSERA30  -  FORMATTED LOG PRINT                                                                                         PAGE  0010
 02BEB3BC 00035C   00000000 00000000  00000000 00000000   00000000 00000BC8  08000000 02BEB060   *.......................H.......-*
 02BEB3DC 00037C   00000000 00000000  00000000 00000000   00000000 00000000  00000000 00000000   *................................*
 02BEB3FC 00039C   00000000 00000000  00000000 00000000   00000000 00000000  00000000 02A625A8   *..............................W.Y*
 02BEB41C 0003BC   00000000 00000000  00000000 00000000   02BBE040 0000AF00  00000000 00164F31   *.................. ..........|.*
 02BEB43C 0003DC   00000000 00000000  00000000 00000000   00000000 00000000  00000000 00000000   *................................*
 02BEB45C 0003FC   02CC9000 00000000  00000000 00000000   00000000 00000000  00000000 00000000   *................................*
 02BEB47C 00041C   00000000 82B0224E  00000000 00000000   00000000 00000000  00000000 00000000   *....B..+........................*
 02BEB49C 00043C   00000000 02BEA13F  02BEA040 00000000   07004040 40404040  00000000 00000000   *........... ......      ........*
 02BEB4BC 00045C   00000000 02BEA340  00000000 02BEA140   00000000 00000000  00000000 00000000   *......T ....... ................*
 02BEB4DC 00047C   00000032 00000000  00000000 02A69B40   02BE9040 00000000  00C13190 00000000   *............W. ... .....A......*
 02BEB4FC 00049C   02BEBC28 7FFFF000  00001000 0101001B   00000000 02D0C450  02D0C714 A69AEFF4   *....".0...............D&;.G.W..4*
 02BEB51C 0004BC   B43B4104 00F6B760  02BCB048 004DA000   00CC0000 00000000  00800000 00000000   *.....6.-.....(..................*
 02BEB53C 0004DC   00000000 D2469AD2  469AD246 02BEA610   00000000 00000000  00000000 00000000   *....K..K..K...W.................*
 02BEB55C 0004FC   02BEA63C 02BEBD18  00000000 C6D4D8C9   D5E2D9E3 02D0C610  00000000 00000000   *..W.........FMQINSRT..F.........*
 02BEB57C 00051C   000C4040 00000000  00000000 00000000   00000000 00000000  00000000 00000000   *..  ............................*
 02BEB59C 00053C   00000000 00000000  00000000 00000000   00000000 00000000  00000000 00000000   *................................*
 02BEB5BC 00055C   0056967A 00000000  20000000 00000000   00000000 00000000  00000000 00000000   *..O:............................*
 02BEB5DC 00057C   00000000 00000000  00FE8600 00000000   00164F31 02D0C524  00000000 00000000   *..........F.......|...E.........*
 02BEB5FC 00059C   00000000 82BF5110  80000000 02BEB648   82B12A28 82ACF988  41000000 02BEB060   *....B...........B...B.9H.......-*
 02BEB61C 0005BC   82BF5110 02CC9068  02BEB060 02D0C610   02C33840 82B12926  02C33000 02BCB048   *B..........-..F..C. B....C......*
INTERNAL TRACE RECORD          ID = MRQE  SEGNO=02  RECNO = 000000AF  TIME  16.48.16.30   DATE  92.321
CONTINUE
 02BEB63C 0005DC   00000064 00C4F5A8  02B121D8 00000000   02BEB600 02BEB690  82AD1196 82F2EC60   *.....D5Y...Q............B..OB2.-*
 02BEB65C 0005FC   00000000 00000004  02CC9020 02CC90F0   02C33000 00000000  02C33840 82B12926   *...............0.C.......C. B...*
 02BEB67C 00061C   00000000 02BEB060  82ACF9E8 00C4F5A8   02AD1132 00000000  02BEB648 02BEB6D8   *.......-B.9Y.D5Y...............Q*
 02BEB69C 00063C   82F2FBE2 02F2FC12  00000000 00000020   02CC9020 02CC90F0  02C33000 00000000   *B2.S.2.................0.C......*
 02BEB6BC 00065C   02C33024 02C33078  02BEBD18 02BEB060   02F2FC60 00C4F5A8  82F2EC60 00000000   *.C...C.........-.2.-.D5YB2.-....*
 02BEB6DC 00067C   02BEB690 02BEB720  82F2FDAA 000607F0   00000004 02A69B40  02CC9020 02CC90F0   *........B2.....0.....W. .......0*
 02BEB6FC 00069C   02C33000 02F2FF44  02A69BD8 02BEB690   02CC9138 02BEB060  02F2FC60 00C4F5A8   *.C...2...W.Q......J....-.2.-.D5Y*
 02BEB71C 0006BC   02F2FC12 00000000  02BEB6D8 02BEB768   80060979 82AFEDF8  00C4F5A8 02BEB060   *.2.........Q........B..8.D5Y...-*
 02BEB73C 0006DC   00000410 0000060C  00060C28 000003E0   02A69B50 02BEB63C  02A69B98 02BEB060   *.................W.&;....W.Q...-*
 02BEB75C 0006FC   80060A38 00C4F5A8  000607F0 00000000   02BEB720 02BEB7B0  82B0037B 02B00CA0   *.....D5Y...0............B..#....*
 02BEB77C 00071C   000E0418 00000418  02AFFDF8 000E0418   00000000 00000410  02BEB068 82B00364   *...........8................B...*
 02BEB79C 00073C   00C16E00 02BEB060  00C16000 00C4F5A8   82AFEDF8 00000000  02BEB768 02BEB7F8   *.A>....-.A-..D5YB..8...........8*
 02BEB7BC 00075C   82B0037B 02B00CA0  00000001 6481630C   02AFFDF8 00C4F5A8  00000000 02A69B40   *B..#.........A.....8.D5Y.....W. *
 02BEB7DC 00077C   02A69BD8 02BEB690  02CC9138 02BEB060   00C16000 00C4F5A8  82AFEDF8 00000000   *.W.Q......J....-.A-..D5YB..8....*
 02BEB7FC 00079C   02BEB7B0 02BEB840  82B00ED5 82DEA2E8   00C4F5A8 02BF51B0  02AFFDF8 00000024   *....... B..NB.SY.D5Y.......8....*
 02BEB81C 0007BC   00000004 02A69B40  02A69BD8 02BEB690   02CC9138 02BEB060  00C16000 00C4F5A8   *.....W. .W.Q......J....-.A-..D5Y*
 02BEB83C 0007DC   02B00DE0 00000000  02BEB7F8 02BEB888   800FACA3 82B27150  02F341D8 02BF74B0   *...........8...H...TB..&;3.Q....*
 02BEB85C 0007FC   02F341D0 00000000  02F341D8 02BF74A0   02F34000 00C4EE40  02CC9138 02BEB060   *.3.......3.Q.....3 ..D. ..J....-*
 02BEB87C 00081C   00043008 00C4F5A8  000FABB8 00000000   02BEB840 02BEB8D0  800FACA3 82B27150   *.....D5Y........... .......TB.&;*
 02BEB89C 00083C   02F341D8 02BF74B0  02F341D0 00000000   02F341D8 02BF74A0  02F34000 00C4EE40   *.3.Q.....3.......3.Q.....3 ..D. *
 02BEB8BC 00085C   02CC9138 02BEB060  00043008 00C4F5A8   000FABB8 00000000  02BEB888 02BEB918   *..J....-.....D5Y...........H....*
 02BEB8DC 00087C   82AC1CB1 82B27150  00000000 02BF51B0   00C39230 03052C68  02BF6680 02BF74A0   *B...B..&;........CK.............*
 02BEB8FC 00089C   00000000 00C39200  00C2A120 02BEB060   82AC13CE 00C4F5A8  02AC11E8 00000000   *.....CK..B.....-B....D5Y...Y....*
 02BEB91C 0008BC   02BEB8D0 02BEB960  82AC3B47 02AC424E   00000000 03052C68  02F507F8 000001FF   *.......-B......+.........5.8....*
 02BEB93C 0008DC   03052C68 03052C60  82AC3AE8 00C39200   00C2A120 03052C84  0000001C 00C4F5A8   *.......-B..Y.CK..B.....D.....D5Y*
 02BEB95C 0008FC   02AC39C0 00000000  02BEB918 02BEB9A8   00000000 00000000  00000000 00000000   *...............Y................*
 02BEB97C 00091C   00000000 00000000  00000000 00000000   00000000 00000000  00000000 00000000   *................................*
```

Figure 109. Sample 6701-MRQE Record (Part 2 of 3)

```
DFSERA30  -  FORMATTED LOG PRINT                                                                                                       PAGE  0011
 02BEB99C 00093C   00000000 00000000  00000000 00000000   02BEB960 02BEB9F0  00000000 00000000   *..................-...0........*
 02BEB9BC 00095C   00000000 00000000  00000000 00000000   00000000 00000000  00000000 00000000   *................................*
 02BEB9DC 00097C   00000000 00000000  00000000 00000000   00000000 00000000  02BEB9A8 02BEBA38   *...........................Y....*
 02BEB9FC 00099C   00000000 00000000  00000000 00000000   00000000 00000000  00000000 00000000   *................................*
INTERNAL TRACE RECORD            ID = MRQE  SEGNO=03  RECNO = 000000B0  TIME  16.48.16.30   DATE  92.321
CONTINUE
 02BEBA1C 0009BC   00000000 00000000  00000000 00000000   00000000 00000000  00000002 00000000   *................................*
 02BEBA3C 0009DC   02BEB9F0 02BEBA80  00000000 00000000   00000000 00000000  00000000 00000000   *...0............................*
 02BEBA5C 0009FC   00000000 00000000  00000000 00000000   00000000 00000000  00000000 00000000   *................................*
 02BEBA7C 000A1C   00000000 00000000  02BEBA38 02BEBAC8   00000000 00000000  00000000 00000000   *...............H................*
 02BEBA9C 000A3C   00000000 00000000  00000000 00000000   00000000 00000000  00000000 00000000   *................................*
 02BEBABC 000A5C   00000000 00000000  00000000 00000000   02BEBA80 02BEBB10  00000000 00000000   *................................*
 02BEBADC 000A7C   00000000 00000000  00000000 00000000   00000000 00000000  00000000 00000000   *................................*
 02BEBAFC 000A9C   00000000 00000000  00000000 00000000   00000000 00000000  02BEBAC8 02D091A8   *...........................H..JY*
 02BEBB1C 000ABC   00000000 00000000  00000000 00000000   00000000 00000000  00000000 00000000   *................................*
 02BEBB3C 000ADC                      SAME AS ABOVE
 02BEBB5C 000AFC   02D092C8 00000000  82F2EE52 00000000   00001084 00000004  02CC9020 02CC90F0   *..KH....B2.........D...........0*
 02BEBB7C 000B1C   02C33000 00000000  02C33024 02C33078   02BEBD18 02BEB060  02F2FC60 00C4F5A8   *.C.......C...C.........-.2.-.D5Y*
 02BEBB9C 000B3C   82F2EC60 00000000  C6D4D8C9 D5E2D9E3   80001100 00000025  80000000 00000000   *B2.-....FMQINSRT................*
 02BEBBBC 000B5C   00000000 80801000  002A2800 00410000   00000000 00000000  00000000 00000000   *................................*
 02BEBBDC 000B7C   00000000 0092321F  1647193F 00000000   00000000 00000000  00000000 00000000   *.....K..........................*
 02BEBBFC 000B9C   00000000 00000000  00000000 00000000   00000000 02BDF640  00000000 00000000   *......................6 ........*
 02BEBC1C 000BBC   00000000 00000000  00000000 00000000   00000000 00000000  00000000 00000000   *................................*
 02BEBC3C 000BDC   00000000 00000000  00000000                                                   *............                    *
QTPDST
 03052C68 000000   00000000 00000000  00000000 00000005   00000000 0A82007C  00000001 D3E3C5D9   *.....................B.@....LTER*
 03052C88 000020   D4F14040 00000001  03089100 00000000   FFFF0909 00000000  00000000            *M1  ......J.................    *
QSAPWKAD
 02BF74A0 000000   00000000 00000000  00000000 00000000   D8D4C7D9 00005B18  00005B18 02BEB060   *................QMGR..$...$....-*
 02BF74C0 000020   00C4F5A8 0029E604  000FACA2 00000000   00000000 00000000  00000000 00000000   *.D5Y..W....S....................*
 02BF74E0 000040   00000000 00000000  00000000 00000000   00000000 00000000  00000000 00000000   *................................*
 02BF7500 000060                      SAME AS ABOVE
 02BF7520 000080   00000000 02F34280  00000000 00000000   00000000 00000000  00000000 00000000   *.....3..........................*
 02BF7540 0000A0   00000000 00000000  00000000 00000000   00000000 00000000  00000000 00000000   *................................*
 02BF7560 0000C0   00000000 00000000  80800068 00000000   02C33090 000A0000  00000005 00000000   *.................C..............*
 02BF7580 0000E0   0300000E 00000000  00000000 00000000   00000000 00000000  00000000 00000000   *................................*
 02BF75A0 000100   00000000 00000000  00000000 00000000   00000000 00000000  00000000 00000000   *................................*
 02BF75C0 000120                      SAME AS ABOVE
 02BF75E0 000140   00000000 00000000  00000000 00000000   00000000 0000000C                      *........................       *
INTERNAL TRACE RECORD            ID = MRQE  SEGNO=04  RECNO = 000000B1  TIME  16.48.16.30   DATE  92.321
CONTINUE
 02BF75F8 000158   00000000 8C043848  00000000 04000004   0C0B0000 00000000  02F341D0 00000000   *.........................3......*
 02BF7618 000178   00000000 004A0000  00000000 00000000   00000000 00000000                      *........................       *
PSTDCA
 02BDF640 000000   D3D5C2D8 80A09A00  D3E3C5D9 D4F14040   02908000 03052C68  00000000 00000000   *LNBQ....LTERM1  ................*
 02BDF660 000020   00000000 02F2F666  00000000 00000000   00000000 00000000  00000000 00000000   *.....26.........................*
 02BDF680 000040   00000000 00000000  00000000 00000000   00000000 00000000  00000000 00000000   *................................*
DFSERA30  -  FORMATTED LOG PRINT                                                                                                       PAGE  0012
 02BDF6A0 000060                      SAME AS ABOVE
REG14-12
 02BEBB64 000000   82F2EE52 00000000  00001084 00000004   02CC9020 02CC90F0  02C33000 00000000   *B2.........D...........0.C......*
 02BEBB84 000020   02C33024 02C33078  02BEBD18 02BEB060   02F2FC60 00C4F5A8  82F2EC60            *.C...C.........-.2.-.D5YB2.-    *
```

Figure 109. Sample 6701-MRQE Record (Part 3 of 3)

The following table explains some of the key fields in the sample 6701-MRQE record shown in Figure 109 on page 271.

Table 42. Explanations of Fields in 6701-MRQE Diagnostic Record

Block	Offset	Length	Value	Description
PCB	42	02	D4D9	TPCBSTAT = MR = DFSQMRQ0 detected an error
AIB	1C	08	ETRAN19	AIBRSNM2 = Destination name of the scrapped message
	40	04	000000F0	AIBRETRN = DFSQMRQ0 detected an error
	44	04	00001084	AIBREASN = Unique reason code for the message being discarded (scrapped). 1084 indicates message is nonrecoverable (INQUIRY=(,NORECOV) on the TRANSACT macro for TRANCODE=ETRAN19).

Table 42. Explanations of Fields in 6701-MRQE Diagnostic Record (continued)

Block	Offset	Length	Value	Description
I/O AREA	00	24		MSGMRQPF = Condensed MRQ prefix passed to DFSDLA30 by the FMQINSRT BMP
	24	VAR		QLOGMSGP = The message buffer being scrapped starts here and consists of prefix segments and the first (or only) user segment of the message. These segments are mapped by QLOGMSGP.
	24	02	00B5	Length of message buffer
	28	01	01	Log Code = 01 = IMS input message
	29	01	D0	Flag = 10 = MSGFNRQU = message is nonrecoverable
	2A	01	81	DestType = 81 = Trancode Dest, 82=Lterm, User, MSNAME Dest
	34	02	00A8	Preflen = Length of prefix data. This length + 24 = start of user segment data (A8 + 24 = CC).
	50	04	0092318F	Date = Date of message
	54	04	0944105F	Time =Time of message
	58	08	LTERM4	Source = Input LTERM name
	60	08	ETRAN19	Dest = Destination TRANCODE name
PST	1B0	04	ISRT	CallFunc = Message Requeuer call function (ISRT or PURG)
	158	04	00000033	I Count = Count of ISRT calls so far (good and bad)
	5C	04	00000011	P Count = Count of PURG calls so far (good and bad) = Messages requeued
REG14-12	00	3C		Registers 14 through 12 at time of error in DFSQMRQ0. R0 = 00001084 = AIBREASN code.

Note: QTPDST, QSAPWKAD, and PSTDCA were not used yet on this call and contain residual data.

Sample JCL for Printing the 6701-MRQE Diagnostic Records

```
//LOGPRNT   JOB
//JOBLIB  DD DISP=SHR,DSN=IMS610.RESLIB
//IMSLOG0 EXEC PGM=DFSERA10,REGION=512K
//SYSPRINT  DD SYSOUT=A
//SYSUT1 DD DSN=IMS610.OLDSP0,DISP=SHR
//SYSIN     DD *
CONTROL  CNTL
OPTION   PRINT O=5,V=6701,L=2,C=M,E=DFSERA30
OPTION   PRINT O=9,V=MRQE,L=4,T=C,C=E,E=DFSERA30
END
/*
```

Figure 110. Sample JCL for Printing 6701-MRQE Records

Control Blocks Logged at Time of Error (and Their Mapping Macros)

The 6701-MRQE diagnostic record contains the following control blocks and data areas which are logged if they are available at the time of the error:

Table 43. Control Blocks and Data Areas Logged at Time of Error for 6701-MRQE Records

Block	Description	Mapping Macro
PCB	Program Control Block	IDLI TPCBBASE=0,CALLER=IMS

Table 43. Control Blocks and Data Areas Logged at Time of Error for 6701-MRQE Records (continued)

Block	Description	Mapping Macro
AIB	Application Interface Block AIBRETRN, AIBREASN codes	DFSAIB DFSMRAEQ
I/O AREA	Input/Output Area	QLOGMSGP
PST	Partition Specification Table	IDLI PSTBASE=0
QTPDST	Queue Manager Destination Block	ICLI CNTBASE=0, or IAPS SMBBASE=0 (CNT/LNB or SMB) DSECT for QAB/TIB not provided
QSAPWKAD	Queue Manager Work Area	QSAPWKAD
QMBA	Queue Manager Buffer Area	DFSQMGR FUNC=QDSECT
PSTDCA	DL/I Call Parm Area	No DSECT
REG14-12	Registers 14 through 12	No DSECT

Normal Errors and Their AIBREASN Codes

Some errors might be normal. For example, the following AIBREASN codes are considered normal:

AIBREASN **Explanation**

00001080 Message destination is an LU 6.2 synchronous logical unit (LU) name and as such is considered nonrecoverable.

00001084 Message destination is nonrecoverable either because the destination transaction code name was defined as NORECOV or the message was received from a LU 6.2 LU in synchronous conversation mode, which implies nonrecoverable.

00001088 Message was already canceled by IMS. Most likely the cause of this is an output message that was canceled when the application program abended or issued a ROLL or ROLB call.

000010A4 The message being passed by FMQINSRT is an internal IMS message that is not recoverable.

00002014 The message is being purged (enqueued to a temporary destination) and the temporary destination name of the message is an inquiry type LTERM.

For a list and explanations of other AIBREASN codes, see Appendix B, “AIBREASN Codes for Message Requeuer Errors,” on page 449.

Abnormal Errors That Can Be Expected

Some errors are not normal but can be expected. An example is when the source or destination name is not found, an error which could occur if the system had been re-GENed and the resource name was deleted. In any case, it is important to determine the AIBREASN code, destination name, and other characteristics of the message to determine whether or not the error can be expected.

Obtaining Diagnostics in Addition to SCRAPLOG and 6701-MRQE

There might be times when the 6701-MRQE diagnostic records and the SCRAPLOG records combined do not provide diagnostic detail adequate to diagnose the problem efficiently. In this case, you can obtain additional diagnostic details by issuing the following command:

```
/TRACE SET ON PROGRAM pgmname
```

where *pgmname* is the name of the appropriate MRQPSB.

`/TRACE SET ON` *pgmname* causes the logging of additional 6701-MRQB records when the MRQ BMP is processing. 6701-MRQB diagnostic records are almost identical to 6701-MRQE records, with the exception of MRQB appearing where MRQE normally does. You can use these records to obtain additional diagnostic detail. The *pgmname* value is the default MRQ PSBNAME. This value might have been overridden on the MSGQUEUE MRQPSBN= parameter at system generation. To determine if your installation has overridden the name, either consult with your IMS systems administrator or issue the IMS command `/DISPLAY PROGRAM MRQPSB`.

If `PROGRAM MRQPSB` displays as an invalid name, your installation has overridden the default MRQPSB. Consult with your system administrator for the correct name for your installation.

Related Reading: For additional information on the `/TRACE` command, see *IMS Version 7 Command Reference*.

The records contained in this program are in addition to the existing program trace records logged by DFSDLA30. Records logged by DFSDLA30 are types 6701-LA3A and 6701-LA3B, which contain the TPCB, I/O AREA (64 bytes), and PST control blocks. See “DLA3LOG Trace” on page 286 for more information and a sample of the LA3A and LA3B records.

With the program trace set on, for each ISRT call to insert a message (or segment of a message), there will be an LA3A, MRQB, and LA3B record. For each PURG call (which completes and enqueues a message) there is one LA3A and LA3B log record. If an error is detected while processing either call, an additional MRQE record is logged. The MRQE records are logged regardless of whether the program trace is on when an error is detected.

How to Tell When Messages Have Been Successfully Requeued

Messages that are successfully requeued by the Message Requeuer are logged to the OLDS with an identical 01 (input) or 03 (output) log record as the original with the exception of the following:

MSGCFLG3=MSGC3MRQ (that is, Message + 19 = 40) is set to indicate that this message was requeued by the Message Requeuer. This flag is propagated to other messages that originate from this message. (That is, if the message is an input transaction message the flag is propagated to the output response messages when the transaction message is processed. Or, if the message is an MSC message, it is propagated to messages in other IMS/MSC systems when the message is sent across the MSC link.)

Figure 111 shows an input transaction message to TRANCODE=ETRAN18 from input LTERM=LTERM10 that was requeued by the Message Requeuer.

```
01 RECORD
 00000000 000000   00BE0000 01C18110  0800001B 0800001B   00A88000 00408180  C0400000 E3C5D9D4   *.....AA..........Y... A.. ..TERM*
 00000020 000020   F1F04040 00010000  00000000 0092318F   0944330F D3E3C5D9  D4F1F040 C5E3D9C1   *10  .........K......LTERM10 ETRA*
 00000040 000040   D5F1F840 00000000  00000000 C4C6E2D4   D6F24040 00408200  00000009 00000000   *N18 ........DFSM02  . B.........*
 00000060 000060   00000001 C5E3D9C1  D5F1F840 D3E3C5D9   D4F1F040 0909014C  00000000 00000000   *....ETRAN18 LTERM10 ...<........*
 00000080 000080   00000000 00000000  00000000 00000000   00000000 00148300  D3E3C5D9 D4F1F040   *......................C.LTERM10 *
 000000A0 0000A0   40404040 40404040  00120301 C5E3D9C1   D5F1F840 C8C5D3D3  D64B0000 00C7       *        ....ETRAN18 HELLO....G  *
```

Figure 111. Sample Log Record Showing Successfully Requeued Message

Diagnosing Message Routing Problems

There are several user message routing exits that can be used to route or control message processing in a Transaction Manager (TM) or TM/Multiple Systems Coupling (MSC) environment. These user message routing exits are listed here:

- DFSCMTR0 (terminal routing) can route input messages entered from terminals.
- DFSNPRT0 (replacement for DFSCMTR0) can also route input messages from terminals and has additional routing capabilities.
- DFSCMLR0 can route local messages received on a MSC link.
- DFSCMPR0 can route transaction output messages inserted into an alternate PCB set by a `CHNG` call.

- DFSMSCE0 is a consolidated exit replacement for the above four exits. DFSMSCE0 has considerably more routing capabilities.

There are several traces, messages, and information fields in the message prefix area that can be used to diagnose message routing problems in the user exits and in IMS. This information is discussed below.

DFS070 UNABLE TO ROUTE MESSAGE RSN=xxyy

Message DFS070 is issued when any one of the following conditions occur:

- IMS attempts to enqueue a message.
- Any of these TM/MSC exits attempts to reroute a message:
  - DFSMSCE0–Message Routing.
  - DFSMSTR0–Terminal Routing.
  - DFSNPRT0–Input Message Routing.
  - DFSCMLR0–Link Receive.
  - DFSCMPR0–Program Routing.
- A /FORMAT command is entered and an error is encountered while routing a message.

DFS070 Diagnostic Message

Here is an example of the DFS070 diagnostic message:

```
DFS070 UNABLE TO ROUTE MESSAGE RSN=0104
```

The RSN code identifies the module that issued the message (01 = DFSICIO0) and the reason for the error (04 = Prefix buffer length is too large).

In this case DFSICIO0 called the message generator (DFSCLMR0) with R1 = 00680046.

```
Where x'00680046' = module identifier, reason code,message key
                         x'0068' = 0104 (decimal)
                                   01 = Module that issued message = DFSICIO0
                                   04 = Prefix buffer length is too large

                         x'0046  = 70 (decimal) = DFS070 MESSAGE KEY
```

The following table lists:

- The labels used for the module identifier
- The module identifier
- The module function or name

The labels can be used to scan the module source code to locate where the message was issued from.

Table 44. DFS070 Module Identifier Table

LABEL	MODULE IDENTIFIER (decimal)	FUNCTION (MODULE NAME)
MSUK	00	Unknown module or DFSMSCEC requestor
MSTR	01	DC Communication Manager (DFSICIO0)
MSTRAP	02	LU 6.2 Receive LU Manager (DFSRLM10)
MSTROT	03	OTMA Receive LU Manager (DFSYTIB0)
MSPR	04	DC Call Handler (DFSDLA30)
MSLR	05	MSC Analyzer (DFSCMS00)
MSFM	06	/FORMAT Command Processor (DFSICLK0)
MSTE	08	IMS Termination (DFSTRM00)
MSINIT	10	IMS Initialization (DFSIINB0)

The following table lists:
- The label used for the reason code
- The reason code value
- The description of the error

The labels can be used to scan the module source code to locate where the message was issued from.

Table 45. DFS070 Reason (RSN) Codes Table

LABEL	REASON CODE DEC/HEX	DESCRIPTION
PFXUPRER	02/02	User requested 2 user prefix segments (code 8E). Programmer response: The routine that was setting up to call the DFSMSCE0 user exit determined that a user prefix segment had already been obtained. The programmer may need to turn on the DFSMSCE0 trace to determine which routine is setting the field, MSCEUPR (DFSMSCEP) or the flag, MSCEB2RET (DFSMSCEB).
PFXIPRER	03/03	User requested two Workload router prefix segments (code 8F). Programmer response: The routine that was setting up to call the DFSMSCE0 user exit determined that a user prefix segment had already been obtained. The programmer may need to turn on the DFSMSCE0 trace to determine which routine is setting the field, MSCEUPR (DFSMSCEP) or the flag, MSCEB2RET (DFSMSCEB).
PFTOOBIG	04/04	Prefix buffer length is too large. Programmer response: The user prefix segment size field, MSCEUPRL (DFSMSCEP) or the workload router prefix segment size field, MSCEIPRL (DFSMSCEP) is greater than 512. The programmer may need to turn on the DFSMSCE0 trace to determine which routine is setting the field, MSCEUPR or MSCEIPR (DFSMSCEP) to a value larger than 512.
GBPFER	05/05	DFSPOOL error on get prefix buffer. Programmer response: Failure to get storage for the user prefix segment or the workload router prefix segment through the DFSPOOL macro from the HIOP pool.
URCERR1	06/06	User exit return code negative. Programmer response: The program routing exit, DFSCMPR0 or the link receive routing exit, DFSCMLR0 returned a negative return code.
URCERR2	07/07	DFSBCB error getting BCB block. Programmer response: The program routing exit (DFSCMPR0) or the link receive routing exit (DFSCMLR0) returned an invalid return code.
GMSBERR	08/08	DFSBCB error getting BCB block. Programmer response: Failure to get storage for the MSEB block through the DFSBCB macro.
LRBADSID	09/09	Bad SYSID detected. Programmer response: In getting the address for the LNB that is associated with either the origin SID or the SID that is specified by the caller, a bad SYSID was detected.

Table 45. DFS070 Reason (RSN) Codes Table (continued)

IPFX	10/0A	Queue Manager insert prefix error. Programmer response: In an effort to update the MESSAGE PREFIX (01/03) log record a prefix update call was made (DFSQMGR0) to add the user prefix segment and/or the workload router segment. The prefix update routine was unable to add the segment.
ICLR1ERR	11/0B	Non zero return code from DFSICLR1 (DFSICLR0).
AVMLKERR	12/0C	Destination is an invalid type for AVM/ISC link.
MSCEFL1E	15/0F	DFSMSCEC user exit routing flag is in error. Programmer response: An invalid option was requested for the user routing exit flag 1 (MSTRFL1/MSLRFL1/MSPRFL1). Refer to the DFSMSCEP macro for valid options. Check the user exit parameter in the 6701-MSCE record to determine which option was requested. These options are usually set by IMS code.
USRXIFER	16/10	DFSUSRX interface error. Programmer response: The macro, DFSMSCEC invoking DFSUSRX0 through the DFSUSRX macro received a non-zero return code. The value is in field, MSCEBRC in the DFSMSCEB block. Possible values returned are: 1. 04 the user exit routine specified has not been defined (the address in UXDT is zero) 2. 2) Unable to get an interface block via the DFSBCB macro. DFSBCB return code is in field, MSCEBSSRC in the DFSMSCEB block.
IONAMCHG	18/12	User exit changed the destination name of the I/O PCB message. Programmer response: The user exit (DFSMSCE0) set flag MSPR2CHG in field, MSPRFL2 to request that the destination name, MSPRDEST be changed. The PCB is the I/O PCB that cannot be changed. Check the user exit parameter in the 6701-MSCE record to determine which option was requested.
IOROUTE	19/13	User exit requested reroute I/O PCB message. Programmer response: The user exit, DFSMSCE0 requested a routing option of: MSPR2RMT,/MSPR2LSQ,/MSPR2SRC,/MSPR2NDR in field, MSPRFL2. This is invalid if the PCB is the I/O PCB. Refer to the user exit parameter in the 6701-MSCE record to determine which command was requested.
CMDINV	20/14	User exit changed the destination name to a command (such as: `/CMDVERB`). Programmer response: The user exit, DFSMSCE0 changed the destination name to a command. Refer to the user exit parameter in the 6701-MSCE record to determine which command was requested.
SQGINV	21/15	User Link receive exit override MSNAME in segment because destination is not an MSNAME. Programmer response: User exit, DFSMSCE0 in a shared queues group link receive exit failed due to the destination not being an MSNAME.
REGFAIL	22/16	Local shared queue registration (DFSSQIF FUNC=INFRM) failed for the transaction when the user exit requested MSLR2LSQ=1 or MSTR2LSQ=1.

Table 45. DFS070 Reason (RSN) Codes Table (continued)

NOTRANCD	23/17	Terminal routing exit routed the message to a remote IMS (MSTR2RMT=1) but the destination type at MSTRDEST is an unsupported TRANCODE (such as: remote routing is not allowed for LTERM or FAST PATH exclusive TRANCODE).
DSIDINV	24/18	The Terminal, Link Receive or the Program Routing exit returned an invalid destination SYSID (for example: either field, MSTRDSID, MSLRDSID, or MSPRDSID is invalid).
DMSNINV	25/19	The Terminal, Link Receive, or Program routing exit returned an invalid destination MSNAME (for example: either field, MSTRDMSN, MSLRDMSN, or MSPRDMSN is invalid).
SSIDINV	26/1A	The Link Receive exit rerouted an intermediate message (MSLR1INT=1) to this local IMS by setting MSLR2LOC=1, but the message had an invalid return (source) SYSID so this IMS could not accept it locally.
RMT2INV	27/1B	The Terminal, Link Receive, or Program routing exit indicated routing the message to a remote MSC link by setting MSTR2RMT, MSLR2RMT, or MSPR2RMT however the exit did not set either of the corresponding destination SYSID or MSNAME fields (for example: either MSTRDSID, MSLRDSID, or MSPRDSID was left set to zero, or MSTRDMSN, MSLRDMSN, or MSPRDMSN was left set to blanks).
SRC2INV	28/1C	The Program routing exit requested the message be routed to the source MSC system by setting MSPR2SRC=1 however the message cannot be routed because either: • MSC is not available. • Or the source SYSID is not valid because the application program has not issued a get unique (GU). • The application program is a non-message driven BMP.
NDR2INV	29/1D	The Program Routing exit requested a direct routing message be overridden by setting MSPR2NDR=1 however either: • MSC is not available. • This is not a direct routed message with a MSNAME destination. • The overriding name in the front of the I/O area is not valid.
RMT2FSR	30/1E	The Terminal routing exit indicated to route the message to a remote MSC link by setting MSTR2RMT=1, but the input ISC node was set to process the message as a Front End Switch message by the user Front End Switch exit (DFSFEBJ0). Front End Switch messages cannot be routed to MSC links.
RSPROUTE	31/1F	The Link receive exit requested that a response message (MSLR1RSP=1) be rerouted by either setting one of the MSLRFL2 reroute flags. Response messages may not be rerouted.
INBCHGID	33/21	CHANGEID not supported. Programmer response: The user exit (DFSMSCE0) did not use the DFSMSCSV macro or generate module entry code. IMS initialization expects a branch instruction around the character information of entry code. Refer to the sample version of the provided user exit DFSMSCE0's use of DFSMSCSV for more information.

Table 45. DFS070 Reason (RSN) Codes Table (continued)

INBIDLNG	35/23	Character string ’VECTOR’ not present. Programmer response: The user exit (DFSMSCE0) did not use the DFSMSCSV macro or generate module entry code. IMS initialization expects the entry code to contain a length of the module entry code at a given offset. Refer to the sample version of the provided user exit DFSMSCE0’s use of DFSMSCSV for more information.
INBNVECT	35/23	Character string ’VECTOR’ not present. Programmer response: The user exit, DFSMSCE0 did not use the DFSMSCSV macro or module entry code to provide the character string ″VECTOR″ in its entry code. Refer to the sample version of the user exit DFSMSCE0’s use of DFSMSCSVfor more information.
PFXUINVA	36/24	Upon return from the user exit IMS detected that the user prefix at MSCEUPR is invalid. Possible causes are: • Length not in range of 5 to 512 bytes. • Address of prefix is invalid. Must be address obtained by IMS or within HIOP pool. • Length has been changed (MSCEBUPRL). • Address of user exit prefix has changed (MSCEBUPR). • Prefix code not 8E. The programmer may need to turn on the DFSMSCE0 trace to trace the fields, MSCEBUPR and MSCEBUPRL within the DFSMSCEB block.
PFXIINVA	37/25	Upon return from the user exit, IMS detected the Workload Router prefix at MSCEIPR is invalid. Programmer response: • Length not in range of 5 to 512 bytes. • Address of prefix is invalid. Must be address obtained by IMS or within HIOP pool. • Length has been changed (MSCEBIPRL). • Address of workload router prefix has changed (MSCEBIPR). • Prefix code is not 8F. The programmer may need to turn on the DFSMSCE0 trace to trace the fields, MSCEBIPR and MSCEBIPRL within the DFSMSCEB block.
EXIOVLAY	38/26	User exit overlaid the 512 byte user work area buffer. Programmer response: The user exit, DFSMSCE0 appears to have overlaid the 512 byte workarea. The overlay character string, SCDSMCON is inserted at the end of the 512 byte workarea, MSEBIBOV before calling the user exit, DFSMSCE0 and is checked on return. Refer to the user exit DFSMSCEB in the 6701-MSCE record to help determine the overlay.

Table 45. DFS070 Reason (RSN) Codes Table (continued)

EXBOVLAY	39/27	User exit overlaid the MSEB BCB block name (Overlay Check). Programmer response: The user exit (DFSMSCE0) appears to have overlaid the DFSMSCEB block. The DFSBCB system service inserts a character string (MSEB) at the end of the DFSMSCEB block. IMS will abend when the DFSMSCEB block is returned by way of a DFSBCB release request. The DFS070 message will assist in determining when the overlay occurred. Refer to the user exit parameter in the 6701-MSCE record to help determine the overlay.
EXPOVLAY	40/28	User exit overlaid the parameter list (Overlay Check). Programmer response: The user exit, DFSMSCE0 appears to have overlaid the user exit parameter list (DFSMSCEP). The overlay character string, SCDSMCON is inserted at the end of the parameter list, DFSMSCEP before calling the user exit, DFSMSCE0 and is checked on return. Refer to the user exit parameter in the 6701-MSCE record to help determine the overlay.

Codes 41 through 52 apply to the `/FORMAT` command.

FMFND	41/29	The CNT for the terminal to be formatted was not found.
FMRCNT	42/2A	The specified terminal is a remote LTERM.
FMDLNB	43/2B	The specified terminal is a dynamic MSNAME (LNB).
FMMFST	44/2C	The destination terminal (different from the input terminal) is not MFS-formatted.
FMLRESMD	45/2D	The destination terminal is in line response mode.
FMTRESMD	46/2E	The destination terminal is in terminal response mode.
FMCONV	47/2F	Conversation is active on the destination terminal (when LTERM was specified in the command).
FMINP	48/30	The terminal is in input mode only.
FMEXCL	49/31	The terminal was in exclusive mode (when LTERM was specified in the command).
FMQBUF	50/32	The call to Queue Manager failed for a `PUT LOCATE` call.
FMIPREF	51/33	The `INSERT PREFIX` call to Queue Manager failed.
FMMSGNR	52/34	The call to enqueue the message failed.

Using the DFSMSCE0 Routing Exit Trace

The DFSMSCE0 TM/MSC Message Routing Exit trace writes a 6701-MSEA log record when the exit is entered and a 6701-MSEB log record when the exit returns to IMS to process the reroute request. The trace can be activated individually for each exit entry point that processes a message routing request. The following information is traced:

- Exit parameter area, DFSMSCEP
- 512 byte work area
- Message
- Message prefix
- Message segment being inserted

- Other work area storage

This trace is very useful for diagnosing problems in the user exit and in IMS.

The /DISPLAY TRACE EXIT Command

Use the `/DISPLAY TRACE EXIT` command to display the DFSMSCE0 trace status.

To display the DFSMSCE0 trace status, issue the following `/DISPLAY` command:

```
/DISPLAY TRACE EXIT
```

The display will show ON, OFF, or N/A for each DFSMSCE0 trace entry point.

Starting and Stopping the DFSMSCE0 Trace

To start the DFSMSCE0 trace, issue one of the following `/TRACE` commands.

```
/TRACE SET (ON|OFF) EXIT (DFSMSCE0) (ALL|TRBT|TRVT|TR62|
                                     TROT|LRTR|LRLT|LRIN|
                                     LRDI|PRCH|PRIS)
```

Note: Any combination of TRBT, TRVT, TR62, TROT, LRTR, LRLT, LRIN, LRDI, PRCH, and PRIS is valid.

DFS081 Trace Exit Command Unsuccessful RSN=xxyy Message

This message is issued when one or more of the following scenarios occurs:

- IMS attempts to enqueue a message.
- The following user exits attempt to reroute a message:
  - The TM/MSC message routing exit, DFSMSCE0.
  - The Terminal Routing exit, DFSMSTR0.
  - The Input Message Routing exit, DFSNPRT0.
  - The Link Receive exit, DFSCMLR0.
  - The Program Routing exit DFSCMPR0.
- A `/FORMAT` command was entered.
- An error was encountered while routing the message.

The DFS070 Diagnostic Message: This is an example of the DFS070 diagnostic message.

```
DFS070 UNABLE TO ROUTE MESSAGE RSN=0104
```

The RSN code identifies the module that issued the message (01 = DFSICIO0) and the reason for the error (04 = Prefix buffer length is too large).

In this case DFSICIO0 called the message generator (DFSCLMR0) with R1 = 00680046.

```
Where x'00680046' = module identifier, reason code,message key
                         x'0068' = 0104 (decimal)
                                    01 = Module that issued message = DFSICIO0
                                    04 = Prefix buffer length is too large
                         x'0046  = 70 (decimal) = DFS070 MESSAGE KEY
```

The following table lists:

- The label used for the module identifier
- The identifier
- The module function or name

The labels can be used to scan the module source code to locate where the message was issued from.

Table 46. DFS081 Module Identifier Table

LABEL	MODULE IDENTIFIER (decimal)	FUNCTION (MODULE NAME)
ICLN	01	Trace Command Processor (DFSICLN0)

The following table lists:
- The label used for the reason code
- The reason code value
- The description of the error

The labels can be used to scan the module source code to locate where the message was issued from.

Table 47. DFS081 Reason (RSN) Codes Table

LABEL	REASON CODE DEC/HEX	DESCRIPTION
EXTIKW	01/01	Invalid keyword for trace exit.
EXTIPT	02/02	Invalid parameter type for trace exit command.
EXTNPT	03/03	No parameter type was specified for trace exit command.
EXTMPT	04/04	Multiple parameter types for trace exit command.
EXTMCB	05/05	Missing DFSMSCB control block for the trace exit DFSMSCE0 command.
EXTIPS	06/06	Invalid parameter subtype for the trace exit command.
EXTENS	07/07	Trace exit is not supported for this environment.
EXTENL	09/09	Required exit is not loaded for start trace command.
EXTSCF	10/0A	System command failure.
EXTIPL	11/0B	Invalid parameter length.

Contents of the DFSMSCE0 Trace Records

DFSMSCE0 records are type X'6701' with a trace ID of MSEA (entry) or MSEB (exit). Refer to the DFSMSCEB macro for contents of the MSCEB block.

PROGRAM ROUTING
- MSCEB (Message routing exit interface block)
  (CHNG/ISRT call)
- PCB (CHNG/ISRT call)
- MESSAGE PREFIX (CHNG/ISRT call)
- MESSAGE SEGMENT (ISRT call) maximum of 256 bytes

LINK RECEIVE
- MSCEB (Message routing exit interface block)
- MESSAGE PREFIX

TERMINAL ROUTING
- MSCEB (Message routing exit interface block)
- MESSAGE SEGMENT maximum of 256 bytes

Note: To assist in diagnosing DFSMSCE0 exit problems, the MSCEB block will maintain the following information:

- 8 bytes EYECATCHER 'DFSMSCEB'
- 4 bytes Routing exit type:
  `TRTB|TRVT|TR62|TROT|LRTR|LRLT|LRIN|LRDI|PRCH|PRIS`
- 4 bytes Address of ECB
- 4 bytes Address of interface block
- 4 bytes Address of DFSMSCE0 exit parameter list

Using the Transaction/Program Trace to Diagnose Routing Errors

The transaction or program trace can be used to diagnose routing error problems that are related to the user program routing exits DFSCMPR0 and DFSMSCE0. By setting this trace on for a transaction or program, IMS logs a 6701-LA3A record at entry to DFSDLA30, and a 6701-LA3B when DFSDLA30 returns to the application program. In addition, if the DFSMSCE0 exit is being used, IMS logs a 6701-MSEA record when the exit is entered, and a 6701-MSEB when the exit returns to IMS. IMS also logs a 6701-MSCE error record, for each DFSMSCE0 related routing error.

Module DFSDLA30 receives control for every user application program call to a TPPCB (such as I/O TPPCB or an alternate TPPCB). If the DFSCMPR0 routing exit is being used, DFSDLA30 receives control for every `CHNG` call to an alternate modifiable TPPCB. The DFSMSCE0 routing exit can be tailored to receive control for the first ISRT call of each new message to a I/O TPPCB or alternate TPPCB, or for each `CHNG` call to a alternate modifiable TPPCB.

For example, if the transaction trace is active for TRANA, and a TRANA message is processed and the user application program issues a ISRT to an alternate TPPCB, and the DFSMSCE0 exit is being used to route ISRT calls, IMS will trace the following records with this command:

```
/TRACE SET ON TRANSACTION transaction_name
```

```
   6701-LA3A - DFSDLA30 called to process ISRT call
   6701-MSEA - DFSMSCE0 called to process ISRT route
   6701-MSEB - DFSMSCE0 returns
   6701-MSCE - Logged if routing error detected, even if tran/prog trace
               is not active
   6701-LA3B - DFSDLA30 returns (ISRT/route processed)
```

To trace the DL/I portion of data communication for a specific program, enter this command:

```
/TRACE SET ON PROGRAM  program_name
```

Refer to the DLA3LOG trace in this manual for samples of the 6701-LA3A/LA3b records and the DFSMSCE0 6701-MSEA/MSEB records.

Note: For program routing exit (DFSMSCE0) call errors, TPPCB status, AIBRETRN, and AIBREASN codes are set. For DFSCMPR0, only TPCB status (A1) code is set.

TPCB STATUS, AIBRETRN, and AIBREASN Codes for DFSDLA30 Routing Errors

TPCB STATUS, AIBRETRN, and AIBREASN codes for DFSDLA30 routing errors are as follows:

```
  TPCBSTAT AIBRETRN AIBREASN       COMMENTS

  A1       00000104 MSERQINV(0560) EXIT ROUTE REQUEST INVALID (DFSCMPR0/DFSMSCE0)

  A1       00000104 MSEREJA1(0564) EXIT REJECTED CALL WITH A1 STATUS
                                   (DFSCMPR0/DFSMSCE0)

  A1       00000104 MSER3303(0568) EXIT REJECT CALL WITH U3303
                                   ABEND (DFSCMPR0/DFSMSCE0)

  A4       00000104 MSEREJA4(056C) EXIT REJECT CALL WITH A4 SECURITY
```

```
                                  ERROR (DFSMSCE0)

  E1       00000104 MSEREJE1(0570) EXIT REJECT CALL WITH E1 USER
                                  STATUS (DFSMSCE0)

  E2       00000104 MSEREJE2(0574) EXIT REJECT CALL WITH E2 USER
                                  status (DFSMSCE0)

  E3       00000104 MSEREJE3(0578) EXIT REJECT CALL WITH E3 USER
                                  STATUS (DFSMSCE0)

  QH OR XF 00000104 MSEDIRRO(057C) EXIT DIRRECT ROUTE OVERRIDE ERROR
                                  (DFSCMPR0/DFSMSCE0)
```

Using the DC LINE/NODE/LINK TRACE to Diagnose Routing Problems

The DC trace traces: line, node, and MSC link activity. It can be used in conjunction with (or without) the DFSMSCE0 exit trace, to diagnose message routing problems in the terminal routing, input message routing, and link receive exits. These traces log 6701 log records with a variety of trace IDs (such as: 6701-A01). If any of these traces is active, then IMS will log a 6701-MSEA record when the message routing exit is called and a 6701-MSEB log record when the exit returns. For example, if the node trace is active, the following trace records will be logged:

```
   6701-A01  - DC analyzer (DFSICIO0) is called to process the message
               LINK the DFSMSCE0 trace will log X'6701' records with a
               trace ID of MSEA (entry) or MSEB (exit) for terminal
               routing or link receive.  Refer to DFSMSCEB macro for
               the contents of the MSCEB block.

   6701-MSEA - DFSMSCE0 called to process the message

   6701-MSEB - DFSMSCE0 returns

   6701-MSCE - Logged if routing error detected, even if the line, node,
               or link trace is not active

   6701-A03  - DC Analyzer determines what to do next
```

Using 01/03 Log Record Trace

A double word trace to reflect the user routing request is included in the Transaction Management Router Segment of the 01/03 log records. The trace reflects the user exit routines called and the user options requested by the varies user exits. The trace reflects:

```
     BYTE 1   -  user parameter list (DFSMSCEP) flag 1
                 indicates the user routing exits called.

     BYTE 2-3 -  User Terminal Routing flags 2 and 3
                 (DFSMSCEP MSTRFL2 and MSTRFL3) indicates
                 the user Terminal Routing options.

     BYTE 4-5 -  User Link Receive Routing flags 2 and 3
                 (DFSMSCEP MSLRFL2 and MSLRFL3) indicates
                 the user LINK Routing options.

     BYTE 6-7 -  User Program Routing flags 2 and 3
                 (DFSMSCEP MSPRFL2 and MSPRFL3) indicates
                 the user Program Routing options.

     BYTE 8   -  Currently unused
```

DLA3LOG Trace

The DLA3LOG trace writes entries to the IMS log at entry to and exit from the DC call analyzer (DFSDLA30).

Starting the Trace

To start the trace, issue one of the two following `/TRACE` commands.

To trace the DL/I portion of data communication for a specific transaction, enter:

```
/TRACE SET ON TRAN transaction name
```

To trace the DL/I portion of data communication for a specific program, enter:

```
/TRACE SET ON PROGRAM program name
```

Content of the Trace Records

DFSDLA30 records are type X'6701' with a trace ID of LA3A (entry) or LA3B (exit). They contain:

- PCB
- Maximum of 64 bytes of the I/O area
- MODNAME
- PST
- SMB of the transaction (if the program in the IMS control region is an MPP or a message driven BMP)

The PCB and PST areas are always logged. The I/O area, MODNAME, and SMB are additional areas that are logged when available and applicable to the call type:

- The I/O area can be logged only on entry or exit. For example, a GN call logs the I/O area on exit, while an ISRT call logs the I/O area on entry. Depending on the call type, the I/O area can be logged on both entry and exit.
- The MODNAME is logged only on an entry trace.
- The SMB is logged on both the entry and exit traces.

Field PSTSYNFC in the PST contains the following calls:

04 ABTERM IN PROGRESS

08 SYNC POINT PHASE 1

0C SYNC POINT PHASE 2

10 PURGE TP PCBS

14 PHASE 1 SYNC POINT ENQ OUTPUT TO TEMP DEST

18 ROLB CALL

1C INVALID ABENDU0820

20 ABORT

Field PSTFUNCT in the PST contains the following calls:

01 GU

03 GN

41 ISRT

50 SETO

67 INQY

83 CHNG

85 CHKP

87 CMD

88 GCMD

89 ROLB

8A ROLS

8C SETS

8F AUTH

90 PURG

Figure 112 on page 289 is an example of a DLA3LOG trace.

Example of DLA3LOG Trace Records

```
INTERNAL TRACE RECORD          ID = LA3A  SEGNO=00  RECNO = 0000009A  TIME  07.45.06.42   DATE  93.014
PCB
 0271B084 000000   00300038 00010018  40404040 40404040   006DD054 00000000  00009F58 C9D6D7C3   *........        ._..........IOPC*
 0271B0A4 000020   C2404040 00000000  00000000 00000000   00000000 0271B084  E6E3D6D9 40404040   *B   ...................DWTOR    *
 0271B0C4 000040   12004040 0093014F  0745063F 00000006   40404040 40404040  40404040 40404040   *..  .L.|........                *
 0271B0E4 000060   40404040 40404040                                                             *                                *
I/O AREA
 02825000 000000   00340000 C3E4E2E3  D6D4C5D9 40D9C5D8   E4C5E2E3 E240C9D5  C6D6D9D4 C1E3C9D6   *....CUSTOMER REQUESTS INFORMATIO*
 02825020 000020   D540D6D5 40D7C1F2  F860F1F6 F140D4D6   C4C5D3E2 00000000  00000000 00000000   *N ON PA28-161 MODELS............*
MODNAME
 82825850 000000   D4D6C4F4 F0F0F4F2                                                             *MOD40042                        *
SMB
 027CA754 000000   00000000 00000000  00000000 00000000   00000000 00810075  00020002 D7C1D9E3   *.....................A......PART*
 027CA774 000020   40404040 41416000  0700A704 FFFFFFFF   00000002 FFFFFFFF  00001D1D 027D5410   *    ..-...X..................'..*
 027CA794 000040   00000000 0100FFFF  0000FFFF 00000000   027CA7C8 00000000  C4C6E2E2 C1D4F0F2   *................@XH....DFSSAM02*
 027CA7B4 000060   40404040 40404040  00000000                                                   *        ....                    *
PST
 0271B060 000000   00000000 82801A39  02978C04 02CB51DC   00000000 00000000  00000000 00000000   *....B....P......................*
 0271B080 000020   02819040 00300038  00010018 40404040   40404040 006DD054  00000000 00009F58   *.A. ........        ._..........*
 0271B0A0 000040   C9D6D7C3 C2404040  00000000 00000000   00000000 00000000  0271B084 E6E3D6D9   *IOPCB   ...................DWTOR*
 0271B0C0 000060   40404040 12004040  0093014F 0745063F   00000006 40404040  40404040 40404040   *    ..  .L.|........            *
 0271B0E0 000080   40404040 40404040  40404040 00000000   00000000 00000000  02C5A758 00000000   *            .............EX.....*
 0271B100 0000A0   00000000 00000000  04000002 02CB5148   00000000 027CA754  0094000E 01420080   *.....................@X..M......*
 0271B120 0000C0   02CB5138 04000002  00000000 00000001   00000001 00000000  02825840 0280E610   *.........................B. ..W.*
 0271B140 0000E0   006D3D08 00000000  00000000 02825000   0275DC40 00000000  00000080 00000000   *._...........B&;... ............*
 0271B160 000100   00000000 028BB020  01020304 00000000   00000000 00000000  D4D7D740 40404040   *........................MPP     *
 0271B180 000120   D4D7D740 40404040  00000000 00000000   00000000 00000000  00000000 00000000   *MPP     ........................*
 0271B1A0 000140   00000000 00000000  00000000 00000000   00000001 00000000  00000001 00000000   *................................*
 0271B1C0 000160   00000000 00000000  00000000 00000000   00000000 00000000  00000000 00000000   *................................*
 0271B1E0 000180   00000000 00000000  00000000 00000000   00000000 00000000  00000000 028258A8   *..............................B.Y*
 0271B200 0001A0   00000000 00000000  C5C0FFFF 8A000000   C9E2D9E3 4140BA07  0271B084 00000000   *........E.......ISRT. .....D....*
 0271B220 0001C0   00000002 00C53D20  00000000 00000000   00000000 00000000  00000000 00000000   *.....E..........................*
 0271B240 0001E0   00000000 00000000  000000A0 02000000   00000000 00000000  00000000 00000000   *................................*
 0271B260 000200   00000000 00000000  8299C762 00000000   00000000 00000000  00000000 00000000   *........BRG.....................*
 0271B280 000220   00000000 00000000  00000000 0290B210   00000000 00000000  00000000 00000000   *................................*
 0271B2A0 000240   00000000 00000000  00000000 00000000   00000000 00000002  000E0300 02825000   *.............................B&;*
 0271B2C0 000260   02707540 00000000  027573A4 00000000   00000000 00000000  00000000 00000000   *... .......U....................*
 0271B2E0 000280   C9E2D9E3                                                                      *ISRT                            *
INTERNAL TRACE RECORD          ID = LA3A  SEGNO=01  RECNO = 0000009B  TIME  07.45.06.42   DATE  93.014
CONTINUE
 0271B2E4 000284   00000000 00000000  00000000 0271B084   00000000 00000000  00000000 00000000   *...............D................*
 0271B304 0002A4   00000000 00000000  00000000 00000000   00000000 00000000  00000000 00000000   *................................*
 0271B324 0002C4   00000000 00000000  00000000 00000004   00F76180 00000000  00000840 00000000   *.................7/........ ....*
 0271B344 0002E4   00011C00 0271B3D8  10000000 00000000   00000000 00008000  C0808000 24008000   *.......Q........................*
 0271B364 000304   00000000 00000000  00000000 0275DC54   00000000 00000000  00000000 00000000   *................................*
 0271B384 000324   00000000 00000000  00000000 00000000   00000000 00000000  00000000 00000000   *................................*
 0271B3A4 000344                     SAME AS ABOVE
 0271B3C4 000364   00000000 00000000  00000000 00000BC8   08000000 0271B060  00000000 00000000   *...............H.......-........*
 0271B3E4 000384   00000000 00000000  00000000 00000000   00000000 00000000  00000000 00000000   *................................*
 0271B404 0003A4   00000000 00000000  00000000 00000000   00000000 028225A8  00000000 00000000   *....................B.Y........*
 0271B424 0003C4   00000000 00000000  026DE040 00004B00   000E15E6 00196FF2  00000000 00000000   *.........._. .......W..?2........*
 0271B444 0003E4   00000000 00000000  00000000 00000000   00000000 00000000  028BB000 00000000   *................................*
 0271B464 000404   00000000 00000000  00000000 00000000   00000000 00000000  00000000 00000000   *................................*
 0271B484 000424   00000000 00000000  00000000 00000000   00000000 00000000  00000000 0275D83B   *..............................Q.*
 0271B4A4 000444   0275D73C 00000000  07004040 40404040   00000000 00000000  00000000 0275DA3C   *..P.......      ................*
 0271B4C4 000464   00000000 0275D83C  00000000 00000000   00000000 00000000  00000000 00000000   *......Q.........................*
 0271B4E4 000484   00000000 028FCA40  02759040 00000000   00C16190 00000000  0271BC28 00000000   *....... ... .....A/.............*
 0271B504 0004A4   00000000 00000000  00000000 0280E450   0280E714 A6E4A497  78F98705 00F741B0   *.............U&;.X.WUUP.9G..7..*
 0271B524 0004C4   026EB048 006DD000  00340000 00000000   00800000 00000000  00000000 02CD2469   *.>..._..........................*
 0271B544 0004E4   AD2CD246 0275DD0C  00000000 00000001   00000000 00000000  0275DD38 0271BD18   *..K.............................*
 0271B564 000504   00000000 C4C6E2E2  C1D4F0F2 0280E610   00000000 00000000  000C4040 00000000   *....DFSSAM02..W...........  ....*
 0271B584 000524   00000000 00000000  00000000 00000000   00000000 00000000  00000000 00000000   *................................*
 0271B5A4 000544   00000000 00000000  02757040 00000000   00000000 00000000  569ABC9A 1D1D014C   *........... ...................<*
 0271B5C4 000564   00000000 00001D1D  0271B0BC 00000000   014C0000 E6E3D6D9  40404040 00000000   *.................<..WTOR    ....*
 0271B5E4 000584   000056E0 00000000  00196D3D 0280E524   00000000 00000001  0000FFFF 827BEC70   *.........._...V.............B#..*
 0271B604 0005A4   80000000 0271B648  829B8A1E 82978630   00000000 0271B060  827BEC70 028BB068   *........B...BPF........-B#......*
 0271B624 0005C4   0271B060 0280E610  02825840 829B891C   02825000 026EB048  00000064 00C53D20   *...-..W..B. B.I..B&;.>.......E..*
 0271B644 0005E4   029B81E8 00000000  0271B600 0271B690   82978774 0297C2FE  00000000 02707540   *..AY............BPG..PB........ *
 0271B664 000604   02825840 0271B084  02825000 82825850   02825840 829B891C  82978630 0271B060   *.B. ...D.B&;BB.&;B. B.I.BPF....-*
 0271B684 000624   82978698 00C53D20  82978630 00000000   0271B648 0271B6D8  8297C424 02C45B80   *BPFQ.E..BPF............QBPD..D$.*
 0271B6A4 000644   00000004 02707540  0000000C 0271B084   02825000 02707554  02707598 829B891C   *....... .......D.B&;.......QB.I.*
```

Figure 112. DLA3LOG Trace Records (Part 1 of 2)

```
INTERNAL TRACE RECORD                        ID = LA3A  SEGNO=02  RECNO = 0000009C  TIME  07.45.06.42   DATE  93.014
CONTINUE
 0271B6C4 000664   0297C488 0271B060  0297C4A0 00C53D20   0297C2FE 00000000  0271B690 0271B720   *.PDH...-.PD..E...PB............*
 0271B6E4 000684   82C45D79 82999D60  00C53D20 0271B060   00000410 00000584  0272E61C 02707598   *BD).BR.-.E.....-.......D..W....Q*
 0271B704 0006A4   02707554 0271B6C4  0272E5A4 0271B060   82C45E38 00C53D20  02C45B80 00000000   *.......D..VU...-BD;..E...D$.....*
 0271B724 0006C4   0271B6D8 0271B768  8299B341 0299BAEC   000E3E8D 00003E8D  0299AD60 000E3E8D   *...Q....BR...R...........R.-....*
 0271B744 0006E4   00000000 00000410  0271B068 8299A28A   00C19E00 0271B060  00C19000 00C53D20   *...........BRS..A.....-.A...E..*
 0271B764 000704   82999D60 00000000  0271B720 0271B7B0   8299B341 0299BAEC  00C53D20 027BED10   *BR.-............BR...R...E...#..*
 0271B784 000724   0299AD60 00C53D20  00000000 02707540   0272E594 829B891C  0272E078 0271B060   *.R.-.E......... ..VMB.I........-*
 0271B7A4 000744   00C19000 00C53D20  82999D60 00000000   0271B768 0271B7F8  8299BD25 829CE580   *.A...E..BR.-...........8BR..B.V.*
 0271B7C4 000764   00C53D20 027BED10  027BED10 00000024   00000004 02707540  0272E594 829B891C   *.E...#...#............. ..VMB.I.*
 0271B7E4 000784   0272E078 0271B060  00C19000 00C53D20   0299BC2C 00000000  0271B7B0 0271B840   *.......-.A...E...R............. *
 0271B804 0007A4   8299BF17 829CA578  00C53D20 027BED10   027BED10 00000028  0272E604 02707588   *BR..B.V..E...#...#........W....H*
 0271B824 0007C4   02707550 0271BC48  0272E5A4 0271B060   00C19000 00C53D20  0299BE12 00000000   *...&;.....VU...-.A...E...R......*
 0271B844 0007E4   0271B7F8 0271B888  8299BD25 829CE580   00C53D20 027BED10  027BED10 00000024   *...8...HBR..B.V..E...#...#......*
 0271B864 000804   00000004 02707540  0272E594 00000832   0272E078 0271B060  00C19000 00C53D20   *....... ..VM...........-.A...E..*
 0271B884 000824   0299BC2C 0271B600  0271B840 0271B8D0   82957237 829A19C8  00000000 0275F260   *.R......... ....BN..B..H......2-*
 0271B8A4 000844   C3D5E340 000001FF  02C5A758 02C5A758   00000000 00C34200  00C2A1C8 0271B060   *CNT .....EX..EX......C...B.H...-*
 0271B8C4 000864   0271B060 00C53D20  0295703E 00000000   0271B888 0271B918  829A1ACB 029A248E   *...-.E...N.........H....B.......*
 0271B8E4 000884   00000000 00F76180  C3D5E340 000001FF   02C5A758 02C5A758  0271B04C 00000000   *.....7/.CNT .....EX..EX....<....*
 0271B904 0008A4   027BEC70 0271B060  0275F260 00C53D20   829A19C8 00000000  0271B8D0 0271B960   *.#.....-..2-.E..B..H...........-*
 0271B924 0008C4   829A2579 829554F8  00C53D20 00C2A238   C3D5E340 000001FF  02C5A758 02C5A758   *B...BN.8.E...BS.CNT .....EX..EX.*
 0271B944 0008E4   00C2A238 00000000  027BEC70 0271B060   0275F260 00C53D20  029A248E 00000000   *.BS......#.....-..2-.E..........*
 0271B964 000904   0271B918 0271B9A8  8011566B 829CA578   000053E8 02766910  00000000 02B54000   *.......Y...,B.V....Y.......... .*
 0271B984 000924   000053E8 02766900  00000000 02766910   0271B0EC 0271B060  02B56260 00C53D20   *...Y...................-...-.E..*
 0271B9A4 000944   00115588 00000000  0271B960 0271B9F0   8011566B 829CA578  02B541D8 02766910   *...H.......-...0...,B.V....Q....*
 0271B9C4 000964   02B541D0 00000000  02B541D8 02766900   02B54000 02766910  0271B0EC 0271B060   *...........Q...... ...........-*
 0271B9E4 000984   00053CE0 00C53D20  00115588 00000000   0271B9A8 0271BA38  8011566B 829CA578   *.....E.....H.......Y.......,B.V.*
 0271BA04 0009A4   02B541D8 02766910  02B541D0 00000000   02B541D8 02766900  02B54000 02766910   *...Q...............Q...... .....*
 0271BA24 0009C4   0271B0EC 0271B060  00053CE0 00C53D20   00115502 00000000  0271B9F0 0271BA80   *.......-.....E.............0....*
 0271BA44 0009E4   00000000 00000000  00000000 00000000   00000000 00000000  00000000 00000000   *................................*
 0271BA64 000A04                     SAME AS ABOVE
 0271BA84 000A24   0271BA38 0271BAC8  00000000 00000000   00000000 00000000  00000000 00000000   *.......H........................*
INTERNAL TRACE RECORD           ID = LA3A  SEGNO=03  RECNO = 0000009D  TIME  07.45.06.42   DATE  93.014
CONTINUE
 0271BAA4 000A44   00000000 00000000  00000000 00000000   00000000 00000000  00000000 00000000   *................................*
 0271BAC4 000A64   00000000 00000000  0271BA80 0271BB10   00000000 00000000  00000000 00000000   *................................*
 0271BAE4 000A84   00000000 00000000  00000000 00000000   00000000 00000000  00000000 00000000   *................................*
 0271BB04 000AA4   00000000 00000000  00000000 00000000   0271BAC8 028041A8  00000000 00000000   *..................H...Y.........*
 0271BB24 000AC4   00000000 00000000  00000000 00000000   00000000 00000000  00000000 00000000   *................................*
 0271BB44 000AE4   00000000 00000000  00000000 00000000   00000000 00000000  028042C8 027579C8   *...........................H...H*
 0271BB64 000B04   00000000 00000000  00000000 00000000   00000000 00000000  00000000 00000000   *................................*
 0271BB84 000B24                     SAME AS ABOVE
 0271BBA4 000B44   C4C6E2E2 C1D4F0F2  00000000 00000000   80000000 00000000  00000000 80801000   *DFSSAM02........................*
 0271BBC4 000B64   002A2A00 00410000  00000000 00000000   00000000 00000000  00000000 0093014F   *.............................L.|*
 0271BBE4 000B84   0743506F 00000000  00000000 00000000   00000000 00000000  00000000 00000000   *..&?;...........................*
 0271BC04 000BA4   00000000 00000000  00000000 02707540   00000000 00000000  00000000 00000000   *............... ................*
 0271BC24 000BC4   00000000 00000000  00000000 00000000   00000000 00000000  00000000 00000000   *................................*
 0271BC44 000BE4   00000000                                                                      *....                           *
```

Figure 112. DLA3LOG Trace Records (Part 2 of 2)

Receive-Any Buffer Analysis

While talking with Level 1 or 2 support representatives, you might need to determine if you are out of receive-any (RECANY) buffers. Use the following procedure to help you make that determination. As you proceed through the steps, write down the information you gather.

Procedure

1. Find the address of the first RECANY buffer.

   | SCD+X'8A4' = pointer to the first RECANY buffer (SCDRECPT)

   SCD+X'890' = size of each RECANY buffer (SCDRCSIZ)

   SCD+X'892' = number of RECANY buffers (SCDRCANY)

2. Offset X'04' in the RECANY buffer points to the next RECANY buffer. You can follow the chain of RECANY buffers using the pointer at offset X'04'.
3. Examine offset X'90' in each RECANY buffer (4 bytes). This field contains either an address of a CLB or zeros. If it contains a CLB address, the buffer is in use. If it contains zeros, in most cases the buffer is available.

4. If the buffer is tied to a CLB, the data you find in the following fields in the CLB is helpful in problem diagnosis.

   CLB+X'00'-> Event Control Block (ECB) (4 bytes)

   CLB+X'20'-> VTAM CID of the session (CLBCID) (4 bytes)

   CLB+X'24'-> QE for queued receive-any buffers (CLBQE) (4 bytes)

   CLB+X'30' = Flag bytes (CLBFLAG1) (4 bytes)

   CLB+X'68'-> Input buffer (CLBINBUF) (4 bytes)

   CLB+X'6C'-> Output buffer (CLBOUTBF) (4 bytes)

   CLB+X'70' = QE for responses (CLBQERES) (4 bytes)

   CLB+X'74' = Flag bytes (CLBVFLAG) (4 bytes)

Finding the Active Save Set

To analyze data communication (DC) problems, you need to find the active save set at the time of abend. Use the following steps to locate the active save set.

1. Locate the registers at entry to abend (error registers). Register 13 points to the address of the active save set.
2. The active save sets begin under eye-catcher `*** SAVE AREA SET***`.
3. Find the save area (SA) address that matches the address in error register 13.

Example of a Save Area Set: If error register 13 contains 320548, you would analyze the save set flow as shown below in Figure 113. The registers in this save set are the registers saved on entry to each module.

```
***SAVE AREA SET****

     EP DFSICIO0
     SA 22FE930

     EP DFSCFEI0
     SA 22E930

     EP DFSCFEP0
     SA 22E990

     EP DFSCIOC0
     SA 229490

     EP DFSQMGR0
     SA 22D990

     EP DFSAOS80
     SA 320548
```

Figure 113. Example of Save Area Set

IMS-VTAM Interface

The basic functions of an IMS DC operation are establishing communications, sending and receiving messages, and terminating communications. The execution of these functions is shared among the elements that make up the network: the terminal, the controller, the VTAM system, the IMS system, and the application. The communications analyzer (DFSICI00) uses the request parameter list (RPL) block to communicate with VTAM, and VTAM returns its status to IMS in the RPL. Therefore, it is important to analyze the RPL. See *VTAM Messages and Codes* for a description of the RPL fields.

IBM 3270 Error Recovery Analysis

When the 3270 detects an error, it sends the processor a sense-status message. There are four categories of sense-status messages:

- Intervention required, such as printer out of paper
- DEVICE END, which indicates the end of an operation
- DEVICE BUSY, normally caused by an operational error
- Hardware I/O error within the 3270 complex, such as a data check, control check, or equipment check

If IMS receives a sense-status message other than a DEVICE END, it issues message DFS973I.

BTAM error recovery handles BTAM errors that result in IEA000I messages on the MVS console. These message indicate a TIME OUT, DATA CHECK, or lost data. Message DFS251I or DFS253I generally follows this message.

All 3270 BTAM device-dependent modules record errors on the log using log record X'6703' and ID=TRCE. The following blocks are logged: CLB, CTB, DCB, DEB, IOB, CTT, I/O buffers (called I TP BUF and O TP BUF), polling or selection list (remote 3270 only, called T-LIST) and FLAGS (CLBTEMP1). "Format of X'67' Log Record" on page 125 lists all log records and illustrates the format of the X'67' log record.

Message Format Service Normal BTAM Path

The diagrams in Figure 114 on page 293 show the normal path followed in processing an MFS-BTAM request. You can use these diagrams in your trace analysis of the problem.

The diagrams show only the simplest path. No error handling or paging is considered. IDs, such as A03 and D03, are the same as those in "Content of the Trace Records" on page 255.

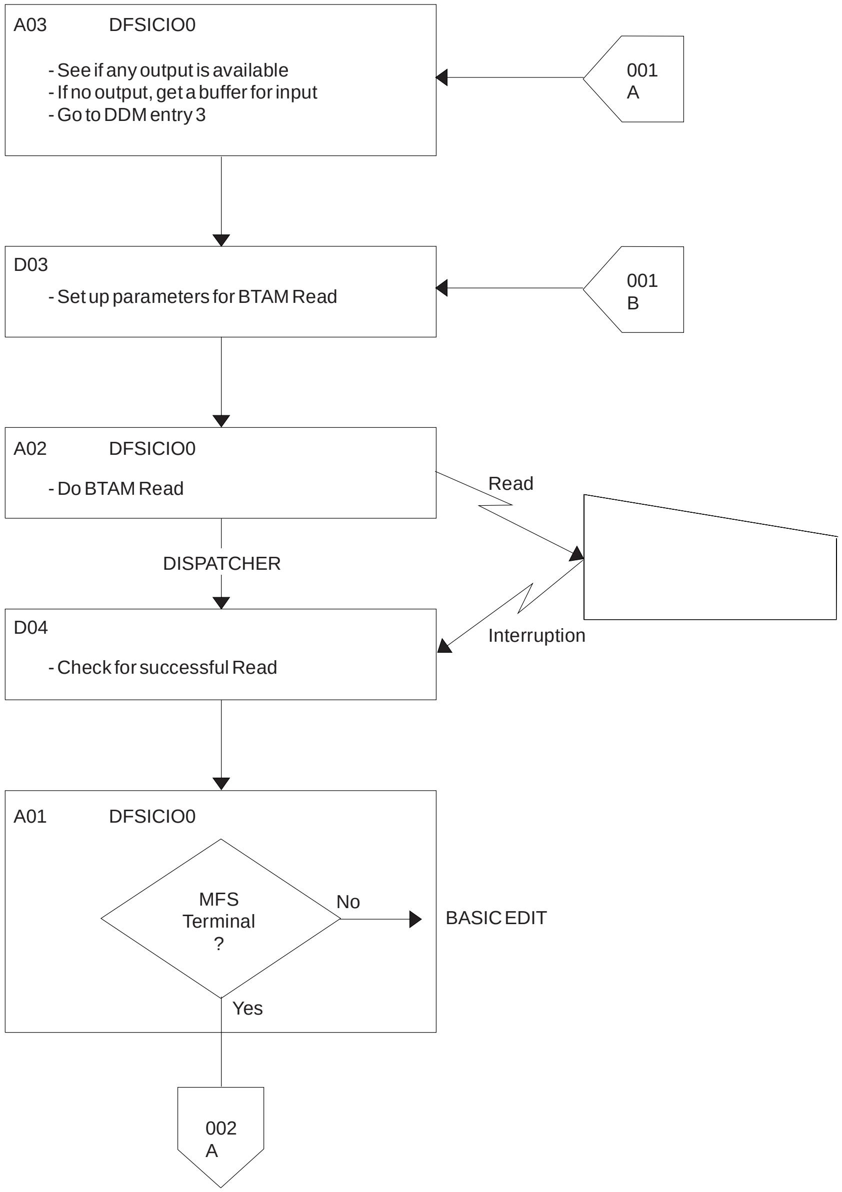


Figure 114. Message Format Service (MFS) Normal BTAM Path (Part 1 of 5)

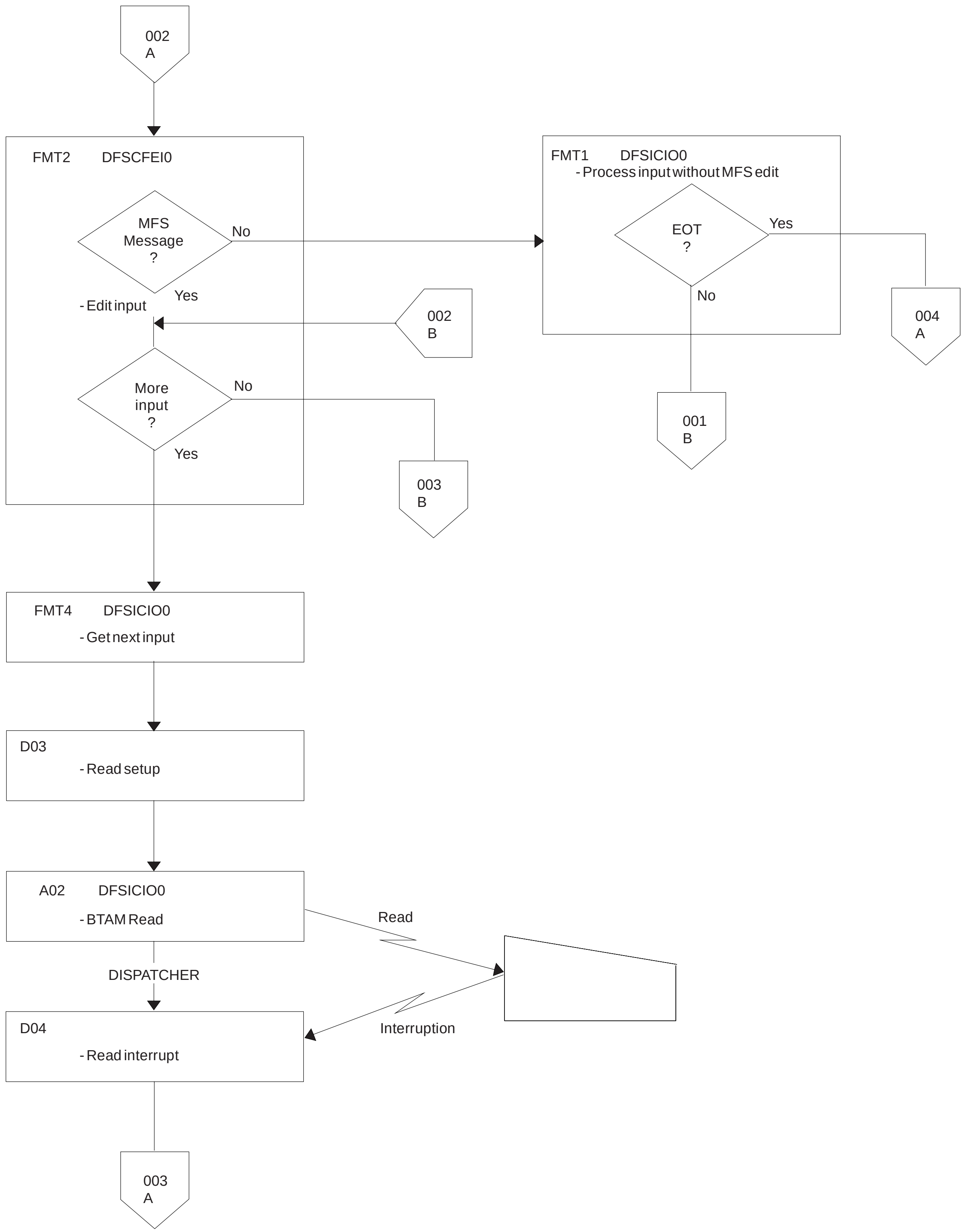


Figure 114. Message Format Service (MFS) Normal BTAM Path (Part 2 of 5)

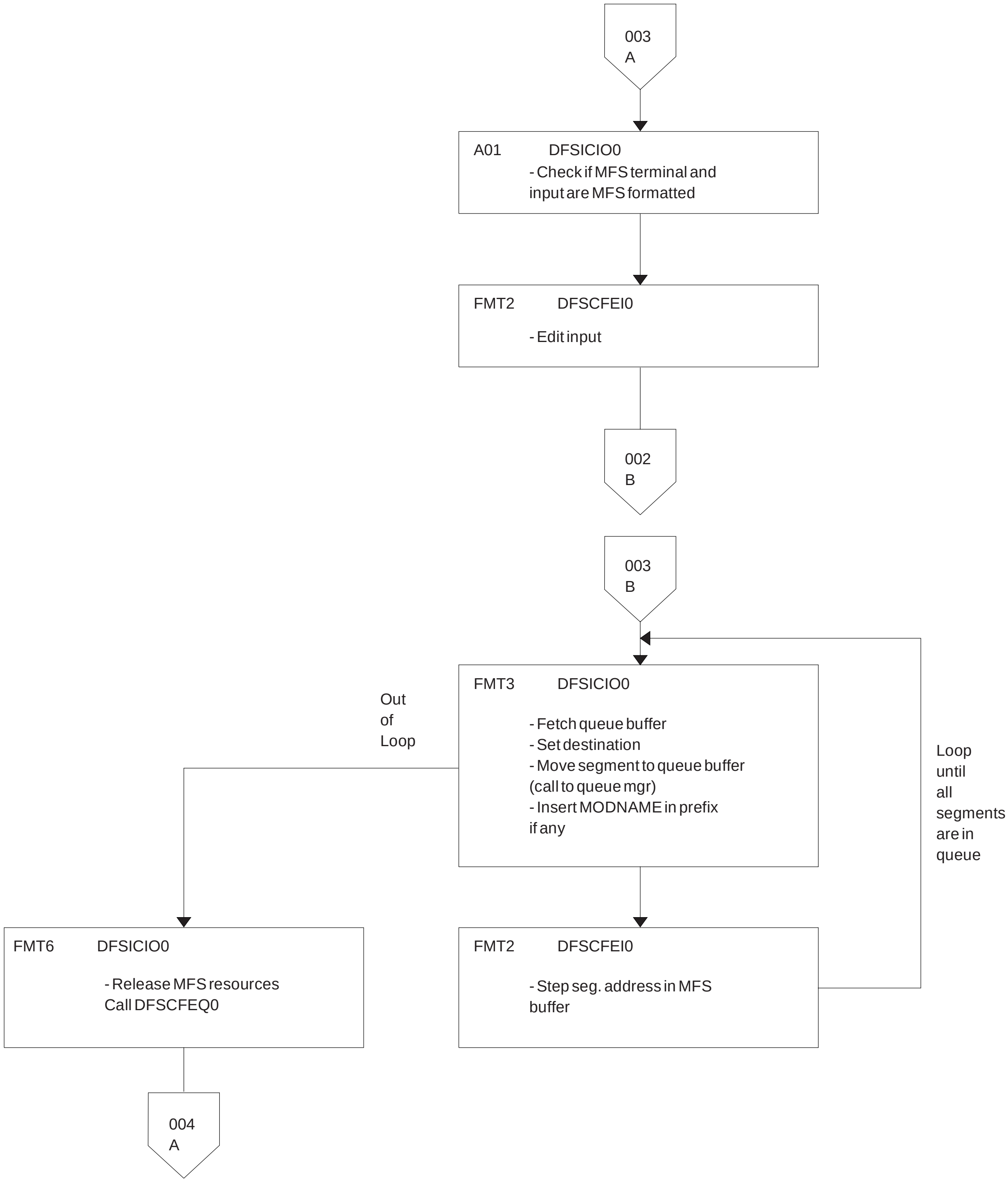


Figure 114. Message Format Service (MFS) Normal BTAM Path (Part 3 of 5)

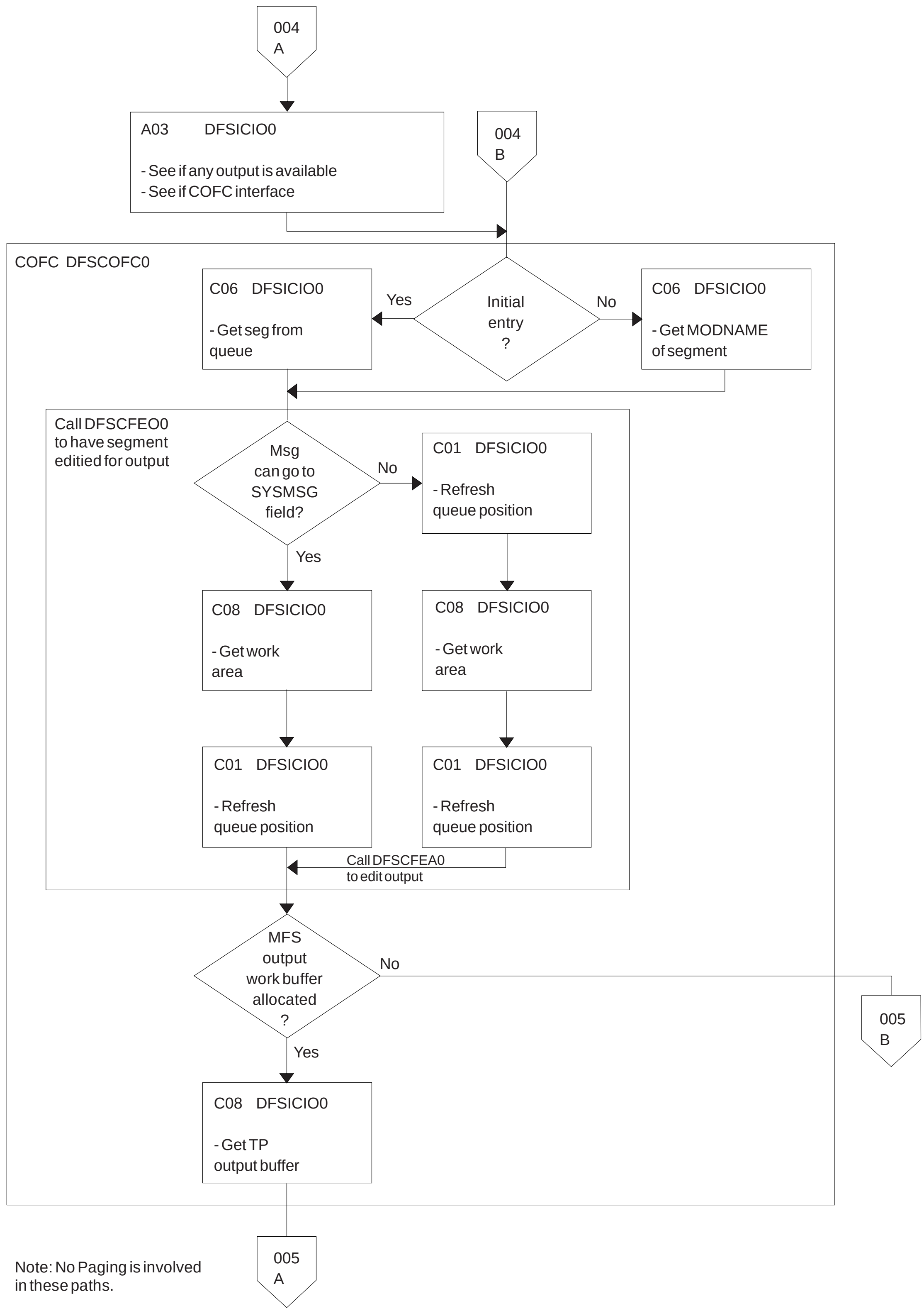


Figure 114. Message Format Service (MFS) Normal BTAM Path (Part 4 of 5)

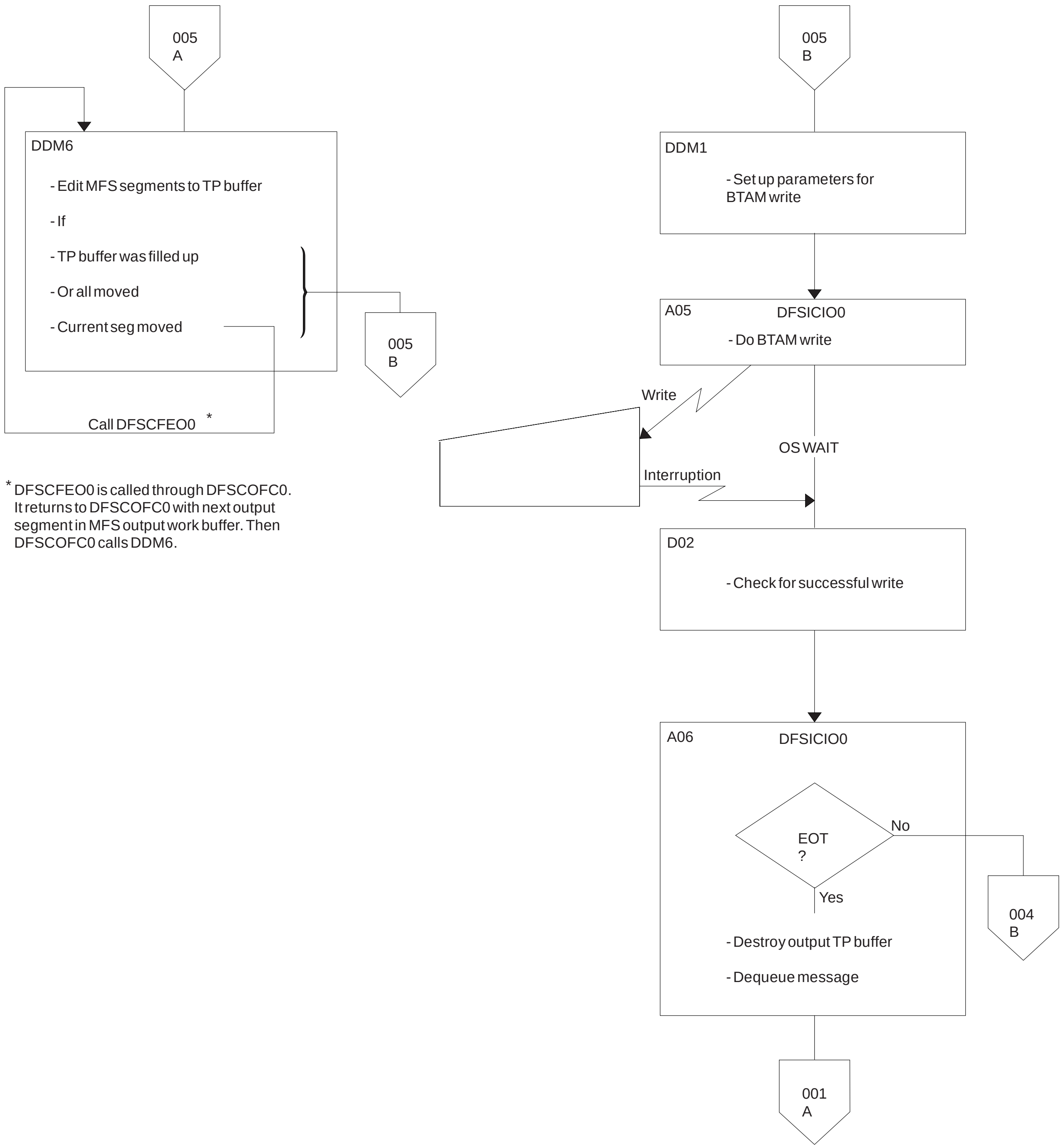


Figure 114. Message Format Service (MFS) Normal BTAM Path (Part 5 of 5)

Diagnosing Message Format Service Problems

For information about starting, stopping, and printing the DC trace, see "DC Trace" on page 253.

The number of physical terminals traced and the number of lines traced can affect completeness of trace records and sequence of trace entries.

- Completeness of the trace record, (that is, whether or not all module activity related to a particular I/O action is traced), is affected if only one PTERM is traced. The DDM occasionally can change the current

PTERM pointer before returning to the analyzer. Because the trace switch is kept in the CTB and is checked upon entry of a particular code, some module trace entries might be missing if the current CTB is not always maintained.

- Sequence of entries can be broken if more than one line is traced at a time. In this case, entries for a particular line have to be related by CLB.

Trace records with the following identifiers are useful in diagnosing MFS problems.

DD6M EDIT SEGMENT INTO TP BUFFER

CIB MOD/DOF name

MFS SEG
SEGMENT created by MFS from output message and MOD/DOF

D01/DDM1
PREPARE TO WRITE TO TERMINAL

CIB Offset X'00' contains 8-byte MOD name.

Offset X'0C' contains 8-byte DOF name.

A05 PRIOR TO ISSUING BTAM OR VTAM I/O REQUEST (NORMALLY A WRITE)

CLB For BTAM

Offset X'04' contains operation type. See BTAM documentation.

Offset X'06' contains the data length.

Offset X'0C' contains the address of the data in the output buffer.

O TP BUF
Contains the data to be written to the terminal and the RPL for VTAM devices. Refer to the previous A05 record.

A01 TERMINAL INPUT READY FOR IMS PROCESSING

I TP BUF
Contains input “device segment” 6 to 36 bytes from the beginning of the buffer. The data is preceded by a 2-byte length and 2 bytes of zeros.

FMT2 ENTRY TO MFS INPUT PROCESSING

CIB Offset X'00' contains MID name.

Offset X'22' indicates if PFK or PA key is used.

X'80' PA key

X'40' PFK key

X'21' PA or PFK number

FMT1 MESSAGE TO BE EDITED BY BASIC EDIT, NOT MFS

FMT3 MFS HAS COMPLETED A MESSAGE SEGMENT

MFS SEG
Shows input segment created by MFS.

MFS I WK
Shows complete input message (all segments) and internal segment control information used by DFSCFEI0.

ICLR A message satisfied MSGDEL=NONIOBCB for its destination PTERM and was deleted. The relevant control blocks are traced:

- Destination CTT

- Telecommunication processing program communication block (TP PCB)
- Destination CLB
- Destination CTB

This trace record is produced when any trace level is active for the destination PTERM.

Note: To examine the segments placed in the message queue, see X'01' and X'03' log records. X'01' log records contain input message segments. X'03' log records contain output message segments.

Message Format Service Module Traces

The Communications Interface Block (CIB) contains two module traces: CIBSTRAC and CIBTRACE. These are described below.

CIBSTRAC Trace

CIBSTRAC is located in the CIB + X'50'. This 4-byte trace entry contains information indicating which MFS modules received control and in what order. Figure 115 shows the format.

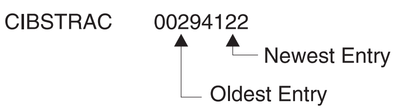


Figure 115. Example of CIBSTRAC Trace

The leftmost nonzero digit shows the oldest entry and the high-order 4 bits of the rightmost byte show the newest. You can ignore the rightmost digit because it is always the same as the digit to its left. The trace entries are described in the following list.

Value (Hex)	Meaning
1	Entry to DFSCFEQ0 (MFS resource cleanup).
2	Entry to DFSCFEI0 (MFS input editing occurred).
3	See value 8. Value 3 usually follows value 8 and is obtained by ORing 1 and 2.
4	INIT or DDFIN entry to DFSCFEO0 (either initial entry or after DDM6 finished current segment).
5	CONT entry to DFSCFEO0 (4 ORed with 1; after successful WRITE, next output segment was requested).
6	PAGEPOS entry to DFSCFEO0 (4 ORed with 2; entry after paging request).
7	DDNEXT entry to DFSCFEO0 (4 ORed with 3; DDM6 wanted next segment).
8	Entry to DFSCFEP0 (3 in the next slot; DFSCFEP0 flushed input message by calling DFSCFEQ0. After returning to DFSCFEP0, page position was established and exit to analyzer D was made. (Entry 8 was shifted left by DFSCFEQ0 entry and entry 1 was written. After returning to DFSCFEP0 1 was ORed with 2.) 5 in the next slot; DFSCFEP0 flushed input message by calling DFSCFEQ0. After returning to DFSCFEP0, message dequeue routine was entered. Entry 8 was shifted and entry 1 was written by calling DFSCFEQ0. After returning to DFSCFEP0, DEQ routines ORed 1 with 4 resulting in 5.
9	Entry to DFSCFEP0 and exit to analyzer 3 entry. (8 ORed with 1).
A	Entry to DFSCFEP0 (page position established) (8 ORed with 2).
C	Entry to DFSCFEP0 and message dequeue requested. (8 ORed with 4).

F Noninitial entry to DFSCFEI0

CIBTRACE Trace

CIBTRACE is located in the extended CIB at CIB+X'70'.If the CIBSEXT flag is on (X'80'), then an extended CIB exists. Figure 116 shows the format.

```
CIBTRACE     0000002C  7412A388
                    ▲         ▲
                    │         └─ Newest Entry
                    │
                    └─ Oldest Entry
```

Figure 116. Example of CIBTRACE Trace

The leftmost nonzero digit shows the oldest entry and high-order 4 bits of the rightmost byte show the newest. You can ignore the rightmost digit since it is always the same as the digit to its left. The trace entries are described in the following list.

Value (Hex)	Meaning
0	ENDMSG entry to DFSCFEI0 (Tests for EOT and spanned operation). If spanned, ENQWORK; if not, set EOM and setup for spanned operation.
1	CPP100 entry to DFSCFEI0. Data was moved to message field.
2	CPP10 entry to DFSCFEI0. Field was padded with fill character or literal has been moved into field.
3	GETLBUF entry to DFSCFEI0. Acquire next line buffer. Return at entry GETLBUF2 with address of line buffer segment in register 1.
4	NOFIT entry to DFSCFEI0. Sets up for spanned operation.
7	GETWORK entry to DFSCFEI0. Acquire work buffer and initialize work buffer header. Moved data from QBUF to work buffer.
8	REFRESH2 entry to DFSCFEI0. DIF table was cleared and setup.
9	ENQWORK entry to DFSCFEI0. Segment in work buffer was moved to QBUF for processing.
A	FINQBUF entry to DFSCFEI0. Compress nulls out of segmenting work buffer.
B	NULLFDE entry to DFSCFEI0. Process all NULLFDEs.
C	PROCQBUF entry to DFSCFEI0. Return to analyzer to process QBUF.
D	GETQBUF entry to DFSCFEI0. Branches to analyzer entry C0 to acquire a QBUFFER.
F	ISRTNULL entry to DFSCFEI0. Inserts all null segments and processes them for move data.

APPC/IMS Diagnostic Aids

This section details the following diagnostic aids:

- LU Manager Trace
- LU 6.2 Module-to-Code Cross-Reference Table
- APPC/MVS Verb-to-Code Cross-Reference Table
- DFS1959E Message Information
- SNAPs and Dumps

LU Manager Trace

The LU manager trace records the flow of control through the IMS LU 6.2 components. Analyzing the trace entries together with the MVS/ESA APPC trace entries is useful in determining the problem.

Starting the LU Manager Trace

The `/TRACE SET ON TABLE LUMI` command activates the trace and sends the entries to an internal table. You can format the table using the Offline Dump Formatter under IPCS, using either the VERBX command or the Interactive Dump Formatter panels. For information about using the Offline Dump Formatter, see “Formatting IMS Dumps Offline” on page 129.

If a SNAP dump is taken, the table is formatted as part of the IMS dump.

If you add the OPTION LOG parameter to the `/TRACE` command, IMS sends the output to an external data set. You can use the File Select and Formatting utility (DFSERA10) with exit DFSERA60 to format the trace entries.

Formatting the LU Manager Trace

Figure 117 shows the general format of an LU manager trace record. Each record is 8 words long. Word 0 holds standard information for each record.

WORD 0		WORD 1	WORD 2	WORD 3	WORD 4	WORD 5	WORD 6	WORD 7
ID	SEQ NUM							

Figure 117. LU Manager Trace Record Format

where **represents**

ID Two-byte trace ID.

SEQ NUM Two-byte trace sequence number assigned by the IMS trace component.

Words 1 through 7 contain data specific to each trace entry, as described below:

TRACE ID = X'7B01' LUM module entry

Word 1 **byte 0:**Module number **bytes 1-3:** Reserved

Word 2 A(ECB)

Word 3 Register 1

Words 4-5 Optional user data

Words 6-7 Time stamp (STCK)

TRACE ID = X'7B02' LUM module exit

Word 1 **byte 0:**Module number **bytes 1-3:** Reserved

Word 2 A(ECB)

Word 3 Return code

Words 4-5 Optional user data

Words 6-7 Time stamp (STCK)

TRACE ID = X'7B03' IMS internal LUM error

Word 1 **byte 0:**Module number **bytes 1-3:** 0

Word 2 A(ECB)

Word 3 Error code

Word 4 Optional user data

Word 5 0

Words 6-7 Time stamp (STCK)

TRACE ID = X'7B04' IMS APPC Status Change

Word 1

byte 0: Module number

byte 1: AWE function requested code

- **X'01':** Initialization request
- **X'02':** Dependent region connected
- **X'03':** Start APPC
- **X'04':** Stop APPC
- **X'05':** Purge APPC
- **X'06':** Cancel APPC
- **X'07':** Terminate APPC
- **X'08':** Attach request
- **X'09':** APPC initialized
- **X'0A':** APPC stopped
- **X'0B':** LU activated
- **X'0C':** LU deactivated
- **X'0D':** XRF takeover
- **X'0E':** Clear TIBs
- **X'0F':** Build LU6.2 descriptors

byte 2: Current APPC status

- **X'C1':**Starting
- **X'C3':**Cancelled
- **X'C4':**Disabled
- **X'C5':**Enabled
- **X'C6':**Failed
- **X'D6':**Outbound
- **X'D7':**Purging
- **X'E2':**Stopped

byte 3: Desired/requested APPC status

- **X'C1':**Starting
- **X'C3':**Cancelled
- **X'C4':**Disabled
- **X'C5':**Enabled
- **X'C6':**Failed
- **X'D6':**Outbound
- **X'D7':**Purging
- **X'E2':**Stopped

Word 2 A(ECB)

Word 3

byte 0: Last APPC status

X'C1': Starting
X'C3': Cancelled
X'C4': Disabled
X'C5': Enabled
X'C6': Failed
X'D6': Outbound
X'D7': Purging
X'E2': Stopped

byte 1: Last Desired/requested APPC status

X'C1': Starting
X'C3': Cancelled
X'C4': Disabled
X'C5': Enabled
X'C6': Failed
X'D6': Outbound
X'D7': Purging
X'E2': Stopped

bytes 2-3: 0

Word 4 0

Word 5 0

Words 6-7 Time stamp (STCK)

| **TRACE ID = X'7B05'** LUM module IWAIT

| **Word 1** **byte 0:**Module number **bytes 1-3:** Reserved

| **Word 2** A(ECB)

| **Word 3** TIB_SYNC_PTR

| **Words 4** A(TIB)

| **Words 5** 0

| **Words 6–7** Time stamp (STCK)

| **TRACE ID = X'7B06'** LUM module IPOST

| **Word 1** **byte 0:**Module number **bytes 1-3:** 0

| **Word 2** A(ECB)

| **Word 3** TIB_SYNC_PTR

| **Words 4** A(TIB)

| **Words 5** 0

| **Words 6–7** Time stamp (STCK)

TRACE ID = X'7C01' Normal return from APPC/MVS

Word 1

byte 0: Module number - See Table 48 on page 308.
byte 1: ATB call number - See Table 49 on page 309.
byte 2: ATB flags
bit 0: Verb issued for asynchronous processing
bit 1: Return code is from asynchronous processing
bit 2: CID given and all zeros
bit 3: TPID field has user data
bit 4: CID field has user data
byte 3: Optional user data

Words 2-3 TPID or user data

Words 4-5 CID or user data

Word 6 Return code

Word 7 A(ECB)

TRACE ID = X'7C02' Unexpected return code from APPC/MVS

Word 1

byte 0: Module number
byte 1: ATB call number
byte 2: ATB flags
bit 0: Verb issued for asynchronous processing
bit 1: Return code is from asynchronous processing
bit 2: CID given and all zeros
bit 3: TPID field has user data
bit 4: CID field has user data
byte 3: Optional user data

Words 2-3 TPID or user data

Words 4-5 CID or user data

Word 6 Return code

Word 7 A(ECB)

TRACE ID = X'7C03' APPC/MVS asynchronous verb entry

Word 1

byte 0: Module number
byte 1: ATB call number
byte 2: ATB flags
bit 0: Verb issued for asynchronous processing
bit 1: Return code is from asynchronous processing
bit 2: CID given and all zeros
bit 3: TPID field has user data
bit 4: CID field has user data
byte 3: Optional user data

Words 2-3 TPID or user data

Words 4-5 CID or user data

Word 6 Reserved (FFFFFFFF)

Word 7 A(ECB)

TRACE ID = X'7F01' APPC Attach from APPC/MVS

Word 1 Reserved

Word 2 XCF message type

Words 3-4 TPID for XCF message

Words 5-6 Local LU to which ATTACH request was directed

Word 7 Time stamp (STCK)

TRACE ID = X'7F02' IMS LU activating or deactivating

Word 1 Reserved

Word 2 XCF message type

Word 3 XCF message LU flags **bit** 0: LU is base LU

Words 4-5 LU name

Word 6 0

Word 7 Time stamp (STCK)

TRACE ID = X'7F03' APPC/MVS starting or stopping

Word 1 Reserved

Word 2 XCF message type

Words 3-6 0

Word 7 Time stamp (STCK)

TRACE ID = X'7F04' CPOOL storage shortage

Word 1 Reserved

Word 2 XCF message type

Word 3 XCF message length

Words 4-5 TPID from XCF message

Word 6 0

Word 7 Time stamp (STCK)

TRACE ID = X'7F05' CPOOL block too small for XCF message

Word 1 Reserved

Word 2 XCF message type

Word 3 XCF message length

Word 4 Cell size

Words 4-5 TPID from XCF message

Word 6 0

Word 7 Time stamp (STCK)

TRACE ID = X'7F06' Invalid request from XCF

Word 1 Reserved

Word 2 XCF message type

Word 3 0

Words 4-5 MEPLSRCE map

Word 6 0

Word 7 Time stamp (STCK)

TRACE ID = X'7F07' APPC/MVS not enabled for Attach

Word 1 Reserved

Word 2 XCF message type

Word 3

byte 0: LSCD status (disabled, failed, stopped)
byte 1: LSCD IN flags (LSCD - APPC/IMS global control block)
byte 2: LSCD OUT flags
byte 3: LSCD flags

Word 4 0

Words 5-6 TPID from XCF message

Word 7 Time stamp (STCK)

TRACE ID = X'7F09' TP deallocate failed

Word 1 Reserved

Word 2 XCF message type

Word 3 Return code

Words 4-6 0

Word 7 Time stamp (STCK)

An Example of the LU Manager Trace

The LU Manager trace in Figure 118 on page 307 shows:

- Some calls to DFS62FD0 caused by `/DISPLAY` commands
- A clean address space caused by a non-LU 6.2 transaction ending
- A synchronous LU 6.2 transaction being executed

It has been formatted by the File Select and Formatting utility (DFSERA10) with exit DFSERA60, which places the module number after word 7.

```
OPTION PRINT O=5,V=67FA,EXITR=DFSERA60
END
 FUNCTION        WORD 0    WORD 1    WORD 2    WORD 3    WORD 4    WORD 5    WORD 6    WORD 7
* LU1 TRACE TABLE - DATE 91323 TIME 11323667 SKIP 0000 TOTAL SKIP 00000000 RECORD NUMBER 00000167
Module Exit     7B023DD8  20000000  03080330  00000004  10800000  00000000  A4D224D2  27C7AB05  32
Module Exit     7B023E22  20000000  03080330  00000004  10800000  00000000  A4D224D2  34020504  32
Module Exit     7B023E2B  20000000  03080330  00000004  10400000  00000000  A4D224D2  340ACC04  32
Module Entry    7B01554C  0B000000  028E0060  02942244  00020080  00000000  A4D22B41  9C54DB04  11
APPC/MVS Exit   7C01554F  0B120000  FFFFFFFF  FFFFFFFF  FFFFFFFF  FFFFFFFF  00000004  028E0060  11-ATBCMAS
Module Exit     7B025552  0B000000  028E0060  00000000  00000000  00000000  A4D22B41  9C5EEA04  11
APPC ATTACH     7F01AC63  00000000  00000001  037AE648  00000002  D3F6F2C9  D4E2F140  48CE0D51
Module Exit     7B02AC8D  20000000  02D02020  00000000  40100000  0310E2B0  A4D24448  CEE7BE05  32
Module Entry    7B01AC97  06000000  0310E2B0  0294D538  00000000  00000000  A4D24448  CEF77405  06
Module Entry    7B01AC9C  10000000  0310E2B0  03036334  01000000  0310E5B2  A4D24448  CF163105  16
Module Exit     7B02AC9D  10000000  0310E2B0  00000000  404008C1  00000000  A4D24448  CF169505  16
Module Entry    7B01ACA2  10000000  0310E2B0  03036334  04020000  0310E5B2  A4D24448  CF1AA305  16
Module Exit     7B02ACA3  10000000  0310E2B0  00000000  404008C1  00000000  A4D24448  CF1B0905  16
APPC/MVS Entry  7C03ACA8  060D8040  037AE648  00000002  037B6018  00000002  FFFFFFFF  0310E2B0  06-ATBRCVW
APPC/MVS Exit   7C01ACB0  060DC000  037AE648  00000002  037B6018  00000002  00000000  0310E2B0  06-ATBRCVW
APPC/MVS Entry  7C03ACB7  060D8040  037AE648  00000002  037B6018  00000002  FFFFFFFF  0310E2B0  06-ATBRCVW
APPC/MVS Exit   7C01ACBF  060DC001  037AE648  00000002  037B6018  00000002  00000000  0310E2B0  06-ATBRCVW
Module Entry    7B01ACC4  22000000  0310E2B0  03035E98  C1D7D6D3  F1F14040  A4D24448  E8BD6C05  34
Module Exit     7B02ACC5  22000000  0310E2B0  00000000  00000000  00140014  A4D24448  E8C11D05  34
Module Exit     7B02ACEF  06000000  0310E2B0  00000000  00000000  00000000  A4D24448  E9427C05  06
Module Entry    7B01AD41  0A000000  028E0060  02942C78  80000080  028E00F8  A4D24448  F43CDC04  10
Module Exit     7B02AD48  20000000  028E0060  00000000  00100000  0310E2B0  A4D24448  F44ABE04  32
Module Exit     7B02AD4B  0A000000  028E0060  00000000  028E00F8  028E00AC  A4D24448  F44D9404  10
APPC/MVS Exit   7C01AD59  3E110000  037AE648  00000002  00000000  00000000  00000000  028E0060  62-ATBASOC
Module Entry    7B01AD5B  10000000  028E0060  02938040  01000000  02CF9AFE  A4D24448  F9BF9F04  16
Module Exit     7B02AD5C  10000000  028E0060  00000000  00000000  00000000  A4D24448  F9C01704  16
Module Entry    7B01AD78  0A000000  028E0060  02942240  00800080  028E00F8  A4D24449  5C418704  10
Module Entry    7B01AD7B  01000000  028E0060  02B921A8  80000000  028E00EC  A4D24449  5C4E4D04  01
Module Entry    7B01AD9A  22000000  028E0060  02B929C0  C1D7D6D3  F1F14040  A4D24449  5D101404  34
Module Exit     7B02AD9B  22000000  028E0060  00000000  04000000  00270027  A4D24449  5D10D104  34
APPC/MVS Entry  7C03ADA0  010F8000  037AE648  00000002  037B6018  00000002  FFFFFFFF  028E0060  01-ATBSEND
APPC/MVS Exit   7C01ADA8  010FC000  037AE648  00000002  037B6018  00000002  00000000  028E0060  01-ATBSEND
Module Entry    7B01ADAD  22000000  028E0060  02B929C0  C1D7D6D3  F1F14040  A4D24449  5E1F7B04  34
Module Exit     7B02ADAE  22000000  028E0060  00000000  04000000  00260026  A4D24449  5E202704  34
APPC/MVS Entry  7C03ADB3  010F8000  037AE648  00000002  037B6018  00000002  FFFFFFFF  028E0060  01-ATBSEND
APPC/MVS Exit   7C01ADBB  010FC000  037AE648  00000002  037B6018  00000002  00000000  028E0060  01-ATBSEND
APPC/MVS Entry  7C03ADC0  01068000  037AE648  00000002  037B6018  00000002  FFFFFFFF  028E0060  01-ATBFLUS
APPC/MVS Exit   7C01ADC8  0106C000  037AE648  00000002  037B6018  00000002  00000000  028E0060  01-ATBFLUS
Module Exit     7B02ADDB  01000000  028E0060  00000000  00010000  00000000  A4D24449  5E828004  01
Module Exit     7B02ADDE  0A000000  028E0060  00000000  028E00F8  00000000  A4D24449  5E855A04  10
Module Entry    7B01ADEC  0B000000  028E0060  02942240  00400080  028E00F8  A4D24449  5E9D0E04  11
Module Exit     7B02ADED  0B000000  028E0060  00000000  028E00F8  00000000  A4D24449  5E9E4C04  11
Module Entry    7B01ADF8  0A000000  028E0060  02942240  00040080  00000000  A4D24449  5EAAA104  10
Module Exit     7B02ADF9  0A000000  028E0060  00000000  00000000  028E00AC  A4D24449  5EABB204  10
Module Entry    7B01AE09  0A000000  028E0060  02942240  00200080  028E00F8  A4D24449  5EB48D04  10
APPC/MVS Entry  7C03AE0C  0A048000  037AE648  00000002  037B6018  00000002  FFFFFFFF  028E0060  10-ATBDEAL
APPC/MVS Exit   7C01AE14  0A04E000  037AE648  00000002  037B6018  00000002  00000000  028E0060  10-ATBDEAL
Module Exit     7B02AE19  20000000  028E0060  00000000  80100000  00000000  A4D24449  5EF81604  32
Module Exit     7B02AE1C  0A000000  028E0060  00000000  028E00F8  00000000  A4D24449  5F104504  10
Module Entry    7B01AE3F  0B000000  028E0060  02942244  00020080  00000000  A4D24449  5F2BD704  11
APPC/MVS Exit   7C01AE42  0B150000  037AE648  00000002  FFFFFFFF  FFFFFFFF  00000004  028E0060  11-ATBCMTP
Module Exit     7B02AE45  0B000000  028E0060  00000000  00000000  00000000  A4D24449  D2E40205  11
Module Entry    7B01AE5A  0B000000  028E0060  02942244  00020080  00000000  A4D24449  D5D0AD05  11
APPC/MVS Exit   7C01AE5D  0B120000  FFFFFFFF  FFFFFFFF  FFFFFFFF  FFFFFFFF  00000004  028E0060  11-ATBCMAS
Module Exit     7B02AE60  0B000000  028E0060  00000000  00000000  00000000  A4D24449  D5DB1205  11
DFS707I END OF FILE ON INPUT
DFS708I OPTION COMPLETE
DFS703I END OF JOB
```

Figure 118. Example of an LU Manager Trace

LU 6.2 Module-to-Code Cross-Reference Table

You can use Table 48 to associate code xx in message DFS1959E and the module number in trace records X'7Bxx' and X'7Cxx' with a module.

Table 48. LU 6.2 Module-to-Code Cross-Reference Table

Mod Num (Dec)	Mod Num (Hex)	Module	Description
01	01	DFSSLUM0	Synchronous output LU manager
02	02	DFSAPPC0	DFSAPPC message switch processor
03	03	DFSCMD00	LU 6.2 command interface
04	04	DFSALM00	Asynchronous output LU manager
05	05	DFSRLM00	Receive LU manager server
06	06	DFSRLM10	Receive LU manager receiver
08	08	DFSAPP10	DFSAPPC keyword parser
09	09	DFSATB00	APPC/MVS verb execution/trace
10	0A	DFS6LUS0	LU 6.2 services interface 1
11	0B	DFS6LUS1	LU 6.2 services interface 2
16	10	DFSRAC60	RACF interface module
21	15	DFS6RST0	LU 6.2 restart processor
22	16	DFS6CKP0	LU 6.2 checkpoint processor
24	18	DFS6ICD0	Read and build LU 6.2 descriptors
31	1F	DFS6ECT0	LU 6.2 XCF message processor
32	20	DFS62FD0	LU 6.2 Find destination routine (QABs/TIBs)
33	21	DFSLUDI0	LU 6.2 User Destination exit
34	22	DFSLIEE0	LU 6.2 User Data Edit exit
35	23	DFSHCI00	XRF takeover processing
36	24	DFS6QFX0	LU 6.2 Nonrecoverable message cleanup
37	25	DFSHAV70	XRF termination/takeover
38	26	DFS62FD1	LU 6.2 Find destination routine (LUBs/DESCs)
50	32	DFSXLUM0	LUM TCB Initialization routine
51	33	DFSLUM00	LUM ITASK manager
52	34	DFSXXCF0	XCF TCB initialization
53	35	DFSXRL00	RLUM TCB initialization
54	36	DFSXALM0	ALUM TCB initialization
55	37	DFSXALC0	ALUM allocate TCB initialization
56	38	DFSFLUM0	LUM TCB ESTAE routine
60	3C	DFSICM20	LU 6.2 command processor
61	3D	DFSTMR00	TM ABEND retry eligibility module
62	3E	DFSTMAS0	TM ASSOCIATE TPI and create ACEE
63	3F	DFSTMCD0	CONNECT/DISCONNECT support

APPC/MVS Verb-to-Code Cross-Reference Table

You can use Table 49 to associate the ATB call number in trace records X'7Cxx' with an APPC/MVS verb.

Table 49. APPC/MVS Verb-to-Code Cross-Reference Table

Verb Num (Hex)	Verb Name	Verb Description
01	ATBALLC	Allocate a conversation
02	ATBCFM	Send a confirmation request
03	ATBCFMD	Send a confirmation reply
04	ATBDEAL	Deallocate a conversation.
05	ATBDFTP	Define TPID
06	ATBFLUS	Empty the local LU’s send buffer
07	ATBGETA	Get conversation attributes
08	ATBGETC	Accept conversation
09	ATBGETP	Get TP properties
0A	ATBGETT	Get conversation type
0B	ATBPTR	Enter receive state
0C	ATBRCVI	Receive data, if available
0D	ATBRCVW	Wait to receive data
0E	ATBRTS	Enter send state
0F	ATBSEND	Send data
10	ATBSERR	Send error
11	ATBASOC	Associate TPID
12	ATBCMAS	Clean address space
13	ATBMIGRP	Join XCF message group
14	ATBSASA	Set address space attributes
15	ATBCMTP	Clean TPID
16	ATBCNTL	APPC/MVS control call
17	ATBCONN	Connect address space to scheduler
18	ATBDCON	Disconnect address space from scheduler
19	ATBEXAI	Extract conversation information
1A	ATBIDEN	Identify scheduler to APPC/MVS
1B	ATBSRN	Set Receive notification
1C	ATBUNID	Unidentify scheduler from APPC/MVS
1D	ATBLEAVE	Leave XCF message group

DFS1959E Message Information

APPC/IMS issues message DFS1959E when a severe internal error occurs. The message format is:

```
DFS1959E SEVERE IMS INTERNAL FAILURE, REASON CODE=xxyy
```

Variable xx is a decimal number that identifies the module. To determine the module associated with the code, see Table 48 on page 308. Variable yy is an internal reason code.

If you receive this message, contact the IBM Support Center with the module number and reason code supplied in the message, and, if requested, output from the LU manager trace.

The following tables provide an explanation of the reason codes listed in the DFS1959E message. Contact the IBM Support Center for action in response to these IMS internal failures.

The following two reason codes are module INDEPENDENT. xx denotes the specific IMS module performing the macro call:

RC	Description
xx98	Failure in DFSPOOL to acquire storage for PL/AS variables via the DFSLUMGT macro.
xx99	Failure in DFSPOOL to release storage for PL/AS variables via the DFSLUMRL macro.

The following reason codes are module DEPENDENT.

DFSALM00

RC	Description
0401	Failure to clear asynchronous control block work pending bit.
0402	Failure to get LUMP pool buffer via DFSPOOL macro.
0403	Failure to free LUMP pool buffer via DFSPOOL macro.
0408	Missing LUNAME from LU 6.2 message prefix.
0409	Missing TPNAME from LU 6.2 message prefix.
0410	Unsupported sync level specified in asynchronous control block or LU 6.2 message prefix.
0411	Invalid conversation type specified in asynchronous control block or LU 6.2 message prefix.
0412	Invalid control data in message segment from GU call.
0413	Invalid control data in message segment from GN call.
0414	No data, redundant DFSQMGR Get Next call. RC=4.
0415	Unknown return code on DFSQMGR Get Next call.
0416	Missing LU 6.2 prefix on DFSQMGR Get Unique call.
0417	Queue already in read status on DFSQMGR Get Unique call. RC >= x'C'.
0418	Failure to dequeue output message. ″No message on queue status″ is indicated. DFSQMGR Dequeue call, RC=8.
0419	Unknown return code from dequeue call. DFSQMGR Dequeue call, RC is other than 0 or 8.
0421	Unknown return code from DFSLIEE0 LU 6.2 user edit exit. RC is other than 0, 4, or 8.

DFSAPPC0

RC	Description
0201	DFSQMGR Get Unique call failure, RC not 0.
0202	DFSQMGR Get Next call failure, RC not 0 and QTP1EOM=0.
0203	DFSQMGR Enqueue call failure, RC not 0.
0204	DFSQMGR Dequeue call failure, RC not 0.
0205	DFSQMGR Insert Move call failure, RC not 0.
0206	DFSQMGR Insert Move call failure, RC not 0.
0207	DFSQMGR Cancel Input call failure, RC not 0.
0208	Failure to read DFSAPPC message from shared queues.

0250 Failure to find or create asynchronous control block.

0260 Router call failure. DFSICLR0 call, RC not 0.

0270 DFSUSE FUNC=NOUSE call failure, RC not 0.

DFSATB00

RC **Description**

0901 Calling module requesting unsupported APPC/MVS verb name.

DFSCMD00

RC **Description**

0301 DFSQMGR Get Unique call failure, RC not 0.

0302 DFSQMGR Get Next call failure, RC not 0.

0304 DFSQMGR Dequeue call failure, RC not 0.

0306 DFSQMGR Insert Move call failure, RC not 0.

0321 Failure to get LUMP pool buffer via DFSPOOL macro.

0322 Failure to free LUMP pool buffer via DFSPOOL macro.

DFSCMLC0

| **RC** **Description**

| **4001** Failure in LUMIF GU call through DFSCMAP0. Type 6701-MSS1/MSS2 records were logged.

| **4002** Failure in processing a remote keyed message. Type 6701-MSS1/MSS2 records were logged.

| **4003** Failure in an INSERT call. Type 6701-MSS1/MSS2 records were logged.

| **4004** Failure in DFSICLR0 message router. Type 6701-MSS1/MSS2 records were logged.

| **4005** DFSCOND0 was called to process an error scratch pad segment for a APPC or OTMA client in conversation mode and an error (RC=08) was returned. Type 6701-MSS1/MSS records were logged.

| **4006** Conversation scratch pad (SPA) message did not have the correct SPA message flags in the message prefix MSGMSFL1 and MSGMSFL2 flags. Type 6701-MSS1/MSS2 records were logged.

| **4007** DFSCONM0 was called to process a normal scratch pad segment for a APPC or OTMA client in conversation mode and an error (RC=0C) was returned. Type 6701-MSS1/MSS2 records were logged.

DFSCMS00

RC **Description**

4101 Failure in LUMIF GU call via DFSCMAP0.

4102 Failure in LUMIF GU call via DFSCMAP0.

4103 Failure in LUMIF GU call via DFSCMAP0.

DFSHCI00

RC **Description**

3501 Failure to get AWE storage via DFSBCB.

DFSRLM00

RC **Description**

0501 AWE extension not a FMH5 Attach request.

0502 Synchronous control block creation failure via DFS62DST FUNC=FIND.

0503 Error freeing XAWE. Unknown storage pool.

0504 Error freeing XAWE via STORAGE macro.

0505 AWE not an FMH5 Attach request.

0506 Error posting DFSRLM10 via DFSSERVR macro.

DFSRLM10

RC **Description**

0601 Failure in DFS62FD0 releasing a synchronous control block (DFS62DST FUNC=RELEASE).

0602 Failure in DFSICLF0 FindDest routine looking up trancode. RC >= x'10'.

0603 Failure in DFSRAC60. DFSRAC6 FUNC=RACINIT RC not 0.

0604 Failure in DFSRAC60. DFSRAC6 FUNC=FRACHECK RC>=x'44'.

0605 Failure in DFSTM0 building a CPI-C dynamic SMB RC not 0.

0606 Failure in DFSICLR0 message router. Enqueue to SMB RC not 0.

0607 Failure to get LUMP pool buffer via DFSPOOL macro.

0608 Failure to free LUMP pool buffer via DFSPOOL macro.

0609 Failure in DFSQMGR updating message to non-recoverable RC not 0.

0610 Failure in DFSTM0 to ENQ prefix to CPIC dynamic SMB RC not 0.

0611 Failure in DFSQMGR to insert Data for SMB or DFSAPPC DFSQMGR Insert Move call failure, RC not 0.

0612 Failure in DFSCMD00 processing IMS command. RC not 0.

0613 Failure in DFSAPPC0 processing Message Switch RC not 0.

0614 Failure in DFSQMGR to cancel a message in progress. RC not 0.

0615 Failure in DFSQMGR to enqueue message for Cmd or DFSAPPC. RC not 0.

0616 Failure in DFSQMGR to update APPC Message Prefix. RC not 0.

0617 Failure in DFSHEIL0 unrecognized return code from Fast Path RC other than 0, 4, 8, or 12.

| **0618** Conversation-id zero when DFSRLM10 has been posted.

0619 Failure in DFS6LUS0 RLUM reposted and not running conversational transaction.

0620 Failure in DFSQMGR to update modname RC not 0.

0621 Failure in DFSQMGR to update a message to response mode.

DFSSLUM0

RC **Description**

0101 Failure in DFSQMGR Get Unique or GN call. RC not 0 and QTP1EOM=0.

0103 Failure in DFSQMGR Dequeue or Cancel call. RC not 0.

0121 Failure to get LUMP pool buffer via DFSPOOL macro.

0122 Failure to free LUMP pool buffer via DFSPOOL macro.

DFS6CKP0

RC	Description
2201	Invalid checkpoint type specified in parameter list. Should be ALL or STATUS.
2202	Data block too large for log record.

DFS6ECT0

RC	Description
3101	Error freeing XAWE via DFSBCB macro.
3102	Error freeing XAWE via STORAGE macro.
3104	Invalid AWE request.
3105	Failure in DFSTM0 to connect all dependent regions FUNC=CONALL.
3107	Failure in DFSBCB to get AWE storage
3109	Error detected in DFS6IDC0 building user descriptors.
3110	Error getting CIOP storage via DFSPOOL macro.
3111	Error freeing CIOP storage via DFSPOOL macro.
3112	VTAM MODIFY USERVAR failed during activation of XRF alternate.
3113	VTAM VARY NET TERM failed for termination of primary system.
3114	Error Posting asynchronous control block via DFSSERVR macro.
3115	Error Checking synchronous control block via DFSSERVR macro.
3116	VTAM MODIFY USERVAR failed for activation of primary system.

DFS6IDC0

RC	Description
2401	Unable to obtain storage for BPAM buffer via STORAGE macro.
2402	Unable to release storage for BPAM buffer via STORAGE macro.
2403	Unknown DFS™ warning message number.
2404	Failure to get LUMP pool buffer via DFSPOOL macro.
2405	Failure to free LUMP pool buffer via DFSPOOL macro.

DFS6LUS0

RC	Description
1007	TIB was released while the task was waiting to synchronize.
1008	TIB_SYNC_PTR was changed, but not to zero.
1010	Unknown service call in main program.
1012	Unable to get storage for LU 6.2 message prefix via DFSBCB macro.
1013	Unable to create an asynchronous control block via DFS62DST FUNC=FIND.
1015	No LUM block given in BLDPRE service call.
1016	Unable to find asynchronous control block or create a new one in CHNG service call. DFS62DST FUNC(FIND).
1018	Conversation-id zero at send time.

1020	Return Code X'1C' from Queue Manager Get Unique call.
1022	Unable to free storage for LU 6.2 message prefix via DFSBCB macro.
1027	Expect input LU 6.2 msg prefix in COPYPF62 service call.
1029	Expect input synchronous/asynchronous control block in COPYPF62 service call.
1032	Unable to find LU 6.2 descriptor entry in BLDPRE service call via DFS62DST macro.

DFS6LUS1

RC	Description
1110	Unknown service call in main program.
1117	No message prefix or synchronous/asynchronous control block given in INQY service call.
1125	No synchronous control block is given in TIBINFO service call
1126	Unable to find the asynchronous or restart synchronous control block in GETQABTIB service call.
1130	Unable to post RLM back in CONVCONT service call.
1133	Unable to find LU 6.2 descriptor entry in INQY service call.
1134	No message prefix supplied in GETQABTIB service call.
1140	DFSQMGR Get Unique or Insert Move call failed in MSGROUTE service call.

DFS6LUS2

RC	Description
1201	No PCB given in READSQ service.
1202	No control block given in READSQ service.
1203	Invalid control block type in READSQ service.
1204	DFSQMGR Get Unique failure in READSQ service.
1205	DFSQMGR Enqueue failure in READSQ service.
1206	DFSQMGR Dequeue failure in READSQ service.
1224	QMGR detected CQS is not available in READSQ service.

DFS6QFX0

RC	Description
3601	Failure in creating a restart control block.
3602	Failure in DFSCIR to create restart ITASK.
3603	Failure in IXCTL to run under restart ITASK.
3604	Failure in DFSCIR to delete restart ITASK.
3682	Issue /STO APPC if APPC/IMS was started; then issue /STA APPC.

DFS6RST0

RC	Description
2101	Log record type not X'22', X'23', or X'24'.
2102	Log record code not X'40'.

DFS62FD0

RC	Description

3201 Failure in DFSBCB to release LU block.

3202 Failure in DFSBCB to release asynchronous control block.

3203 Failure in DFSBCB to get asynchronous control block.

3204 Failure in DFSBCB to release asynchronous control block. (Second location within module.)

3205 Failure in DFSTCBTB FUNC=LOCATE.

3206 Failure in DFSCIR to create ITASK.

3207 Failure in DFSBCB to get synchronous control block.

3208 Failure in DFSCIR to delete ITASK for asynchronous message.

3209 Failure in DFSCIR FUNC=DTASK to release duplicate ITASK for asynchronous message.

3210 Synchronous control block to be released not found in chain.

3211 Input parameter list is invalid, unknown type.

3212 DFSCS failed for LSCD_LOCK.

3213 DFSCS failed adding synchronous control block to chain.

3215 DFSCS failed for LSCD_LOCK while releasing synchronous control block.

3216 IMODULE DELETE failed while releasing asynchronous control block.

3217 Blank LUNAME or nonblank SIDENAME with TPNAME='DFSSIDE'.

3220 Invalid parameters on module entry.

3221 Invalid parameters on module entry.

DFS62FD1

RC	Description
3801	Input parameter list is invalid, unknown type.
3802	Failure in DFSBCB FUNC=GET to get LU block.
3803	Failure in DFSBCB FUNC=REL to release LU block.
3804	Failure in DFSBCB FUNC=GET to get descriptor.
3805	Failure in DFSCS for inserting descriptor into table.
3806	IMODULE DELETE failed for delete of restart synchronous control block hash table.
3807	Failure in DFSBCB FUNC=GET to get synchronous control block.
3808	Failure in DFSBCB FUNC=REL to release restart asynchronous control block.

DFSLUM00

RC	Description
5102	Failure in DFS62FD0 finding an asynchronous control block for notify message.
5109	Unknown return code from MVS clean address space call.
5110	Unknown return code from MVS unidentify call.
5111	IXCLEAVE unsuccessful.

DFSHAV70

RC	Description
3709	Unknown return code from MVS clean address space call.

3710 Unknown return code from MVS unidentify call.

3711 IXCLEAVE unsuccessful.

DFSXLUM0

RC	Description
5009	Unknown return code from MVS clean address space call.
5010	Unknown return code from MVS unidentify call.
5011	IXCLEAVE unsuccessful.

DFS1965 APPC/MVS Call Failure

A call to APPC/MVS had an unexpected return code. The call for FUNCTION=*aaaaaaaa* was issued, and a return code xx from APPC/MVS was the result. Return code xx denotes the specific IMS module performing the APPC call. Refer to the *MVS/ESA Authorized Callable Services* for the meaning of positive values for this return code. Error return codes that represent anticipated conditions are handled by IMS, and do not result in this message. This message is produced when an unexpected result is encountered, which might represent an abnormal condition in some system component.

RC	Description
xx90	Synchronous call failure
xx91	Asynchronous call failure

SNAPs and Dumps

For errors that do not result in an abend, IMS writes a X'67D0' log record or produces an SDUMP, depending on the error. The minimum data dumped for LU 6.2 problems are the control blocks associated with the task in error and the appropriate trace tables.

Tracing Errors in Module DFSCNXA0

DFSCNXA0 is the interface module between IMS and VTAM for all logon processing and abnormal session termination processing. It is often the first module to be notified when a failure occurs on a session and is always the first to get control when a node connects to IMS. The session attributes are verified and the IMS session control blocks are built before the connection request is passed on to signon processing in IMS. The module consists exclusively of VTAM exits.

Location Codes for DFSCNXA0 Error Messages

Message DFS3672I contains the location codes listed in Table 50 on page 317. The message also identifies the exit in which the error occurred.

Session failures might occur that do not cause any DFS messages to be issued by DFSCNXA0. In these cases, only message DFS3672 appears.

The format of the DFS3672I message is as follows:

```
DFS3672I SESSION ERROR. TYPE=aaa CODE=bb QUAL.=cc MSG=dddd
```

where

aaa is the VTAM exit which was driven when the error occurred.

bb is the location code of the error.

cc is the location qualifier of the error.

| Table 50. Location Codes for DFSCNXA0 Error Messages

Location Code (Dec)	Location Code (Hex)	Msg# (DFS)	Exit	Explanation
19	13	3862	LOG	Non-master terminal initiating a session on the alternate system.
20	14	3645	LOG	Generic Resource name used, but VGR for ISC was disabled.
21	15	3645	SCIP	Generic Resource name used, but VGR for ISC was disabled.
1	1	N/A	LOST	No CID in VTAM parameter list.
2	2	N/A	LOST	CLB not found.
3	3	N/A	LOST	Stacked logon chaining error.
4	4	N/A	LOST	CLBs do not match (stacked logon situation).
5	5	N/A	LOST	CLBs do not match (nonstacked situation).
1	1	N/A	NSXT	No CLB in USERFLD of NIB (Cleanup RU).
2	2	N/A	NSXT	No CID.
3	3	N/A	NSXT	CLB not found (Cleanup RU).
4	4	N/A	NSXT	CLB addresses do not match.
5	5	N/A	NSXT	IMS APPLID not found in RID vector list.
6	6	N/A	NSXT	
7	7	N/A	NSXT	Polarity mismatch on MSC link.
8	8	N/A	NSXT	Polarity mismatch on MSC link.
9	9	N/A	NSXT	
10	A	N/A	NSXT	Not Cleanup, NSPE, or Notify—RU is invalid.
11	B	N/A	NSXT	Invalid session key for NSPE.
12	C	N/A	NSXT	Invalid vector key for NOTIFY.
13	D	N/A	NSXT	Invalid session key for NOTIFY.
21	15	2061	NSXT	NSPE/NOTIFY processed.
22	16	2061	NSXT	NSPE/NOTIFY processed, AHDR not cleaned up.
23	17	2061	NSXT	CLB not found (NOTIFY RU).
1	1	N/A	RELQ	VTCB not found.
2	2	N/A	RELQ	Terminal defined with NORELRQ option.
3	3	N/A	RELQ	No CID in nonparallel-session VTCB.
4	4	N/A	RELQ	No CID in any parallel-session VTCBs.
1	1	1915	SCIP	No pointer to RPL.
2	2	1917	SCIP	Node not found.
3	3	3862	SCIP	VTCB not found (XRF Alt.).
4	4	3862	SCIP	Invalid temporary VTCB (XRF Alt.).
5	5	3862	SCIP	BIND not on surveillance link (XRF Alt.).
6	6	3101	SCIP	BIND not from same APPLID.
7	7	3101	SCIP	BIND rejected after setting VLGFF.
8	8	2104	SCIP	Non-LU 6.1 node.

Table 50. Location Codes for DFSCNXA0 Error Messages (continued)

Location Code (Dec)	Location Code (Hex)	Msg# (DFS)	Exit	Explanation
9	9	3111	SCIP	Node stopped.
10	A	3101	SCIP	Logoff requested.
11	B	3101	SCIP	SPQB already allocated. Another 3672 (code=2D) is sent, after the -resp is sent.
12	C	3101	SCIP	BIND not from same APPLID.
13	D	3101	SCIP	BIND rejected after setting CLBVLGFF flag.
14	E	2104	SCIP	CLEAR for non-ISC node.
15	F	970	SCIP	UNBIND entry message sent (after posting).
16	10	1931	SCIP	ASR processing begins.
17	11	2104	SCIP	SDT for non-ISC node.
18	12	1915	SCIP	Invalid command in RPL.
22	16	79	SCIP	Queues not available.

Codes Related to ISC Processing: The codes in Table 51 deal with ISC processing—either as a result of LOGON or SCIP exits being driven. This is reflected in the DFS3672 message via the appending of 'I' to the exit type.

Table 51. Codes Related to ISC Processing

Location Code (Dec)	Location Code (Hex)	Msg# (DFS)	Exit	Explanation
1	1	79	ISC	IMS shutting down.
2	2	1914	ISC	Bad INQUIRE return code.
3	3	1914	ISC	Bad INQUIRE feedback.
4	4	2066	ISC	USERFLD is zeros.
5	5	2066	ISC	1st structured field not 0.
6	6	2066	ISC	User field length = 0.
7	7	2066	ISC	Primary Session Qualifier length = 0.
8	8	2066	ISC	Primary Session Qualifier length > 8.
9	9	2066	ISC	Secondary Session Qualifier length = 0.
10	A	2066	ISC	Secondary Session Qualifier length > 8.
11	B	3107	ISC	SPQB found but allocated.
12	C	3107	ISC	SPQB CRB pointer <> 0.
13	D	2049	ISC	VTCB not found and no dynamic terminals.
14	E	3101	ISC	No available VTCBs.
15	F	3107	ISC	Session initialization already begun.
16	10	3101	ISC	2nd SCIP entry for same session.
17	11	3105	ISC	No CNTs on SPQB.
18	12	3107	ISC	Nonzero CID for existing session.
19	13	3111	ISC	Session blocked (3STOP).

Table 51. Codes Related to ISC Processing (continued)

Location Code (Dec)	Location Code (Hex)	Msg# (DFS)	Exit	Explanation
20	14	3111	ISC	Session stopped.
21	15	3107	ISC	Ran out of CLBs.
22	16	3101	ISC	SPQB CRB pointer = 0.
23	17	1916	ISC	LOGON, but previous session was secondary.
24	18	1916	ISC	SCIP, but previous session was primary.
25	19	2066	ISC	User data length from INQUIRE = 0.
26	1A	3663	ISC	LU type in BIND = '0602' (LU 6.2)
27	1B	3107	ISC	SPQB found but allocated.
28	1C	3107	ISC	SPQB CRB pointer <> 0.
29	1D	3101	ISC	2nd logon entry for same session

The codes in Table 52 may occur during ISC BINDRACE processing.

Table 52. Codes Related to ISC BINDRACE Processing

Location Code (Dec)	Location Code (Hex)	Msg# (DFS)	Exit	Explanation
41	29	N/A	ISC	SESSIONC not issuable—VTAM terminating.
42	2A	N/A	ISC	SESSIONC issued.
43	2B	N/A	ISC	SESSIONC not issuable—VTAM terminating.
44	2C	N/A	ISC	BIND not received.
45	2D	N/A	ISC	SESSIONC issued.

Codes Related to MSC Errors: The codes in Table 53 deal with MSC errors.

Table 53. Codes Related to MSC Errors

Location Code (Dec)	Location Code (Hex)	Msg# (DFS)	Exit	Explanation
51	33	3101	MSC	CID already present.
52	34	3213	MSC	3213 message issued. Code = 4.
53	35	3213	MSC	3213 message issued. Code = 8.
54	36	3213	MSC	3213 message issued. Code = 24.
55	37	3213	MSC	3213 message issued. Code = 32.

The codes in Table 54 deal with MSC SCIP errors.

Table 54. Codes Related to MSC SCIP Errors

Location Code (Dec)	Location Code (Hex)	Msg# (DFS)	Exit	Explanation
71	47	N/A	MSC	CID already present.

Table 54. Codes Related to MSC SCIP Errors (continued)

Location Code (Dec)	Location Code (Hex)	Msg# (DFS)	Exit	Explanation
72	48	N/A	MSC	No USERFLD provided.
73	49	N/A	MSC	RPL not initialized.

Codes Related to Dynamic Logon: The codes in Table 55 deal with dynamic logon errors.

Table 55. Codes Related to Dynamic Logon Errors

Location Code (Dec)	Location Code (Hex)	Msg# (DFS)	Exit	Explanation
81	51	2264	LOG	Do not accept logons.
82	52	3862	LOG	Nonexistent VTCB trying to logon to alternate system.
83	53	2037	LOG	/STA DC not done.
84	54	2104	LOG	Invalid temporary VTCB exists.
85	55	3862	LOG	Invalid temporary VTCB exists.
86	56	3862	LOG	Logon not for XRF link.
87	57	3111	LOG	Node stopped.
88	58	2264	LOG	Logons not accepted and SIMLOG not in effect.
89	59	3862	LOG	In backup but not preopen.
90	5A	3862	LOG	In backup preopen but backup session not allowed.
91	5B	2037	LOG	/STA DC not done.
92	5C	79	LOG	Queues not available.
93	5D	3111	LOG	Node not started.
94	5E	79	LOG	Shutting down and not MTO logging on.
95	5F	3111	LOG	Node stopped.
96	60	3101	LOG	Node logging off.
97	61	3101	LOG	Session terminating.
98	62	3101	LOG	CID already exists.
99	63	3111	ISC	Node stopped on temporary VTCB.

Codes Related to Existing ISC Session Errors: The codes in Table 56 deal with existing ISC session errors.

Table 56. Codes Related to Existing ISC Session Errors

Location Code (Dec)	Location Code (Hex)	Msg# (DFS)	Exit	Explanation
111	6F	3645	ISC	QSAVE could not be gotten.
112	70	3645	ISC	Parsing failed.
113	71	3645	ISC	Dynamic terminals not allowed.

Codes Related to User-Logon-Exit Processing: The location codes in Table 57 deal with user-logon-exit processing.

Table 57. Codes Related to User-Logon-Exit Processing

Location Code (Dec)	Location Code (Hex)	Msg# (DFS)	Exit	Explanation
121	79	3645	LOG	Could not get QSAVE for signon parameters.
122	7A	3645	LOG	Parsing failed.
123	7B	3645	LOG	User logon exit rejected logon.
124	7C	3645	LOG	User logon exit rejected logon.
125	7D	3645	LOG	Invalid ALOT/ASOT value from user logon exit
126	7E	3645	N/A	User logon exit wiped out all descriptors.
127	7F	3645	LOG	A dynamically created logging-on STSN VCTB must have user data.
128	80	3645	LOG	Existing dynamic logging-on STSN VTCB must have user data.

Codes Related to Logon Errors: The codes in Table 58 deal with logon-related errors.

Table 58. Codes Related to Logon Errors

Location Code (Dec)	Location Code (Hex)	Msg# (DFS)	Exit	Explanation
141	8D	3645	N/A	Dynamic terminals not allowed.
142	8E	3646	N/A	Inconsistent attributes—see Table 63 on page 323.
143	8F	3646	N/A	Inconsistent attributes—see Table 63 on page 323.
144	90	3645	N/A	Could not get SOPB storage.
145	91	3645	N/A	Parsing of userdata failed. See Table 61 on page 322.
146	92	3645	N/A	Terminal is the primary or secondary master terminal for the alternate system in an XRF environment.
148	94	3644	N/A	Could not get SOPB storage.
149	95	3644	N/A	Could not get SOPB storage.
150	96	2066	LOG	LUtype in BIND/CINIT conflicts with static ISC block LUtype.
161	A1	3671	N/A	Invalid descriptor specified in userdata.
162	A2	3651	N/A	No default descriptor found.
163	A3	3671	N/A	User logon exit returned invalid descriptor.
164	A4	3644	N/A	Could not get SOPB storage.

Codes Related to Logon Descriptor Processing: The location codes in Table 59 on page 322 deal with logon descriptor processing.

Table 59. Codes Related to Logon Descriptor Processing

Location Code (Dec)	Location Code (Hex)	Msg# (DFS)	Exit	Explanation
181	B5	3663	LOG	LU type must be < 7.
182	B6	3663	LOG	LU type must be >= 0.
183	B7	3663	LOG	Invalid LU type specified.
184	B8	3663	LOG	Invalidly-specified non-SNA 3270 VTAM device. Make sure mode-table is properly defined and referenced.
185	B9	3663	LOG	Invalid LU1/NTO device type.

Codes Related to Logging-on Device Characteristics: The location codes in Table 60 deal with logging-on device characteristics and their compatibility with the logon descriptor being requested.

Table 60. Codes Related to Logging-on Device Characteristics

Location Code (Dec)	Location Code (Hex)	Msg# (DFS)	Exit	Explanation
191	BF	3646	LOG	Invalid SLU1 device logging on.
192	C0	3646	LOG	Device LU type does not match descriptor.
193	C1	3646	LOG	Non-SNA 3270 VTAM logon descriptor invalid for the logging-on device.
194	C2	3646	LOG	Invalid SLU P/3600 type device mismatch with the logon descriptor.
195	C3	3646	LOG	TS type or LU type mismatch.

Qualifier Codes

Codes Related to ETO Parsing Errors: The QUALIFIER codes in Table 61 deal with ETO-related parsing errors (associated with a 3645 message).

Table 61. Qualifier Codes Related to ETO Parsing Errors

Location Code (Dec)	Location Code (Hex)	Msg# (DFS)	Exit	Explanation
1	1	N/A	N/A	Invalid logon descriptor name—no name specified.
2	2	N/A	N/A	Invalid logon descriptor name—name is greater than 8 characters.
3	3	N/A	N/A	Invalid logon descriptor name—no name specified.

Codes Related to VTCB-Creation Errors: The QUALIFIER codes in Table 62 deal with VTCB-creation errors (associated with a 3644 message).

Table 62. Qualifier Codes Related to VTCB-Creation Errors

Location Code (Dec)	Location Code (Hex)	Msg# (DFS)	Exit	Explanation
1	1	N/A	N/A	QSAVE not gotten.

Table 62. Qualifier Codes Related to VTCB-Creation Errors (continued)

Location Code (Dec)	Location Code (Hex)	Msg# (DFS)	Exit	Explanation
2	2	N/A	N/A	VTCB could not be created.
3	3	N/A	N/A	Could not put VTCB into hash table.

Codes Related to Screen-Attribute Errors: The QUALIFIER codes in Table 63 deal with screen-attribute errors (associated with a 3646 message).

Table 63. Qualifier Codes Related to Screen-Attribute Errors

Location Code (Dec)	Location Code (Hex)	Msg# (DFS)	Exit	Explanation
1	1	N/A	N/A	No Device Characteristics Table. MFS DCT (DFSUTB00) utility must be run.
2	2	N/A	N/A	No match on screen size and feature. Update MFS DCT (DFSUTB00) for the missing entry.
3	3	N/A	N/A	Screen size control byte incorrectly specified. The byte itself might be invalid. If 7F is specified, then a valid screen size must also be specified.

IDC0 Trace Table Entries

Error Messages Issued by DFSCNXA0

Table 64 lists codes that identify error messages issued by DFSCNXA0. The code is placed in the MsgID field of an IDC0 trace entry.

Table 64. Codes that Identify Error Messages Issued by DFSCNXA0

Code (Dec)	Code (Hex)	Msg# (DFS)
0	00	2104
4	04	3111
8	08	2037
12	0C	79
16	10	1915
20	14	1917
24	18	1931
28	1C	3862
32	20	970
36	24	1916
40	28	1914
44	2C	2066
48	30	3107
52	34	3105
56	38	3101
60	3C	N/A
64	40	2049

Table 64. Codes that Identify Error Messages Issued by DFSCNXA0 (continued)

Code (Dec)	Code (Hex)	Msg# (DFS)
68	44	3213
72	48	2264
76	4C	3644
80	50	3645
84	54	3646
88	58	3651
92	5C	3663
96	60	N/A
100	64	3671
104	68	2061

The following internal trace formats map IDC0 trace table entries:

Format 1 (″IDC0″):

XL1 Function Code = X'B8' (set by 'DFSTRACE')

XL1 Subcode

XL2 Unusable

XL1 RPLRTNCD - RPL return code

XL1 RPLFDB2 - RPL feedback

XL1 Reserved

XL1 Error type
X'80' = 2061 error
X'40' = 2062 error
X'20' = 970 error

CL8 Nodename

CL8 Mode-table entry name

CL8 Applid (if applicable)
or

CL8 Timestamp

Format 2 (″CNXA″):

One event can span two entries.

First Entry::

XL1 Function Code = X'B9' (set by 'DFSTRACE')

XL1 Subcode

XL2 Unusable

XL1 VTAM-exit indicator

```
00 --> You are looking at the '2nd' entry
04 --> LOGON EXIT ENTERED
08 --> SCIP EXIT ENTERED
```

0C --> NSEXIT EXIT ENTERED
10 --> LOSTERM EXIT ENTERED
14 --> RELREQ EXIT ENTERED

XL1 Error location code

XL1 Location code qualifier

XL1 Processing flag at error time

80 VTCB LATCH HELD
40 LOGON DESCRIPTOR NAME IN CINIT/BIND
20 VTCB DOES NOT YET EXIST
10 VTCB ATTEMPTING CONNECTION FOUND
08 SPQB FOUND
04 IMS CORRELATION ID IN USERDATA
02 ISC PROCESSING ENTERED
01 EXISTING VTCB IN LOGOFF PROCESS

CL8 Nodename

XL4 LOSTERM reason code

XL4 CLB address

XL4 CID

XL1 LU type

XL1 TS profile

XL1 MSG ID of error message

XL1 Reserved

2nd Entry (in the Case of LOGON or SCIP Exits Being Driven)::

XL1 Function Code = X'B9' (set by ’DFSTRACE’)

XL1 Subcode

XL2 Unusable

XL4 Reserved

CL8 Nodename

CL8 Descriptor name or subpool name

XL8 Time stamp

OTMA Diagnostic Aids

This section describes the following diagnostic information to help you analyze problems in OTMA.

- OTMA trace
- OTMA module-to-code cross-reference table
- XCF/MVS verb-to-code cross-reference table
- DFS1269E message information
- Log records
- SNAPs and dumps

OTMA Trace

The OTMA trace records the flow of control through IMS OTMA. Turn on the OTMA trace only if the IBM support representative requests it.

Starting the OTMA Trace

The `/TRACE SET ON TABLE OTMT` command activates the trace and sends the entries to an internal table. You can format the table using the offline dump formatter under IPCS, using either VERBX command or the interactive dump formatter panels. For information about using the offline dump formatter, see “Formatting IMS Dumps Offline” on page 129.

If a SNAP dump is taken, the table is formatted as part of the IMS dump. If you add the `OPTION LOG` parameter to the `/TRACE` command, IMS sends the output to an external data set. You can use the File Select and Format utility (DFSERA10) with exit routine DFSERA60 to format trace entries.

Format of OTMA Trace Records

Figure 119 shows the format of OTMA trace records. Each record is eight words long. Word 0 holds standard information.

WORD 0		WORD 1	WORD 2	WORD 3	WORD 4	WORD 5	WORD 6	WORD 7
ID	SEQ NUM							

Figure 119. OTMA Trace Record Format

where **represents**

ID 2-byte trace ID

SEQ NUM 2-byte trace sequence number assigned by the IMS trace component

Words 1 through 7 contain data specific to each trace entry, as described below:

Trace ID = X'5A01'OTMA module entry

Word 1 Byte 0: Module number
Bytes 1-3: Reserved

Word 2 A(ECB)

Word 3 Register 1

Words 4-5 Optional user data

Words 6-7 Time stamp (STCK)

TRACE ID = X'5A02'OTMA module exit

Word 1 Byte 0: Module number
Bytes 1-3: Reserved

Word 2 A(ECB)

Word 3 Return code

Words 4-5 Optional user data

Words 6-7 Time stamp (STCK)

TRACE ID = X'5A03'IMS internal OTMA error

Word 1 Byte 0: Module number

Bytes 1-3: 0

Word 2 A(ECB)

Word 3 Error code

Word 4 Optional user data

Word 5 0

Words 6-7 Time stamp (STCK)

TRACE ID = X'5A04'XCF state change

Word 1 Byte 0: Module number

Byte 1: XCF call number

Word 2 A(ECB)

Word 7 Time stamp (short)

TRACE ID = X'5B01'XCF/MVS entry

Word 1 Byte 0: Module number

Byte 1: XCF call number

Words 2-7 Control message

TRACE ID = X'5B02'XCF/MVS exit

Word 1 Byte 0: Module number

Byte 1: XCF call number

Word 2 A(ECB)

Word 3-4 XCF token

Word 5 Return code

Word 6 Reason code

Word 7 Time stamp (short)

TRACE ID = X'5CX'OTMA AWE function

Word 1 Byte 0: Module number

Words 2-6 Reserved

Word 7 Time stamp (short)

OTMA Module-to-Code Cross-Reference Table

You can use Table 65 to associate code *xx* in message DFS1269E and the module number in trace records X'5A'*xx*, X'5B'*xx* and X'5C'*xx* with a module.

Table 65. OTMA Module-to-Code Cross-Reference Table

Mod Num (Dec)	Mod Num (Hex)	Module	Description
19	13	DFSYLUS0	OTMA fast services
20	14	DFSYSTO0	OTMA storage manager

Table 65. OTMA Module-to-Code Cross-Reference Table (continued)

Mod Num (Dec)	Mod Num (Hex)	Module	Description
21	15	DFSYRR00	OTMA destination reroute setup routine
22	16	DFSYIO00	OTMA input/output setup routine
23	17	DFSYCM20	OTMA command processor
24	18	DFSYDP40	OTMA `/DIS TRAN`
25	19	DFSYCLH0	OTMA `/TRA` services
26	1A	DFSYRAC0	OTMA security
27	1B	DFSYMGX0	OTMA XCF message exit
28	1C	DFSYGRX0	OTMA XCF group exit
29	1D	DFSYXMO0	OTMA attach member OIM TCB
30	1E	DFSYC480	OTMA `STA/STO` (join/leave) interface
31	1F	DFSYFND0	OTMA FINDDEST processor
32	20	DFSYFD00	OTMA control block processor
33	21	DFSYFD10	OTMA control block processor
34	22	DFSYMOM0	OTMA AWE server DFSYMOM0
35	23	DFSYMEM0	OTMA member AWE server DFSYMEM0
36	24	DFSYIMI0	OTMA getting storage for new member
37	25	DFSYPSI0	TPIPE input AWE server DFSYPSI0
38	26	DFSYPSOO	TPIPE output AWE server DFSYPSO0
39	27	DFSYSND0	OTMA XCF interface
40	28	DFSYTIB0	OTMA synchronous processor DFSYTIB0
41	29	DFSYQAB0	OTMA asynchronous processor DFSYQAB0
42	2A	DFSYLUS0	OTMA service module number 0
43	2B	DFSYCMD0	OTMA command service
44	2C	DFSYCKP0	OTMA check point
45	2D	DFSYSLM0	OTMA synchronous send module
46	2E	DFSYRST0	OTMA restart
47	2F	DFSYIDC0	OTMA descriptor builder
48	30	DFSYQFXO	OTMA queue fixer
49	31	DFSYPRX0	OTMA pre-routing exit routine DFSYPRX0
50	32	DFSYDRU0	OTMA default DRU exit routine DFSYDRU0
51	33	DFSYJL00	OTMA join/leave-DFSYJL00

OTMA Verb-to-Code Cross-Reference Table

You can use Table 66 to associate the XCF call number in trace record X'5B'*xx* with an XCF/MVS verb.

Table 66. XCF/MVS Verb-to-Code Cross-Reference Table

Verb Num (Hex)	Verb Name	Verb Description
01	IXCCREAT	Defines a member to XCF
02	IXCJOIN	Enables a member to join a group

Table 66. XCF/MVS Verb-to-Code Cross-Reference Table (continued)

Verb Num (Hex)	Verb Name	Verb Description
03	IXCQUERY	Return information about groups and members
04	IXCMSGO	Sends a message to another active member
05	IXCMSGI	Receives a message on an active member
06	IXCLEAVE	Disassociates a member from XCF

DFS1269E Message Information

OTMA issues message DFS1269E when a severe internal error occurs. The message format is:

```
DFS1269E SEVERE IMS INTERNAL FAILURE, REASON CODE=xxyy
```

Variable *xx* is a decimal number that identifies the module. To determine the module associated with the code, see Table 65 on page 327. Variable *yy* is an internal reason code.

If you receive this message, contact the IBM Support Center with the module number and reason code supplied in the message, and, if requested, output from the OTMA trace.

The following two reason codes are module independent. Variable *xx* represents the specific IMS module issuing the macro call.

Reason Code	Description
***xx*98**	Failure in DFSPOOL to acquire storage for a variable with the DFSYMAGT macro.
***xx*99**	Failure in DFSPOOL to release storage for a variable with the DFSYMARL macro.

Other reason codes are module dependent.

Log Records

To activate OTMA logging, enter one of the following trace commands from the master terminal or the MVS console.

```
/TRA SET ON tmember client1.
/TRA SET ON tmember client1 tpipe tpipe1.
```

SNAPs and Dumps

For errors that do not result in an abend, IMS writes log record X'67D0', or produces an SDUMP, depending on the error. The minimum data dumped for OTMA problems are the control blocks associated with the task in error and the appropriate trace tables.

Diagnosing Errors Related to Print Data Set Options: IMS Spool API Support

IMS provides an expansion of the DL/I application program interface that allows applications to interface directly to JES and create print data sets on the JES spool. These print data sets can then be made available to print managers and spool servers to serve the needs of the application.

Understanding Parsing Errors

The IMS Spool API support provides feedback to the application program when IMS detects errors in the print data set options included on either the CHNG or SETO calls. The intent of this section is to give a better understanding of high level processing of the parameters associated with the CHNG and SETO calls, including some examples of errors and the types of feedback information that can be expected.

"Error Codes" provides a summary of the error codes that can be expected to be returned if the application provides a feedback area. It might be desirable for the application to develop ways to display these errors by sending a message to an IMS printer or some other technique that allows examination of the parameter lists and feedback area without having to look at a dump. This chapter discusses each error code and provides some examples of when the error code might be expected. This discussion applies to these calls when used with the IMS Spool API support.

When diagnosing multiple parsing error return codes, the first code returned should be the most meaningful. Errors detected with incorrect length fields or previously invalid keywords can result in valid keywords being reported as errors.

Keywords

The parameter lists used with CHNG and SETO calls contain two types of keywords. The two types are those keywords valid for the calls (that is, IAFP, PRTO, TXTU, and OUTN), and the keywords provided as operands of the PRTO keyword (for example, CLASS, FORMS). This separation of keywords is used to determine what type of keyword validation IMS should perform. When looking for valid keywords on the calls, one set of keywords is valid, and when looking at keywords following the PRTO keyword, another set of keywords are valid. For this reason, incorrectly specified length fields may cause one scan to terminate prematurely and keywords to be invalid because they are incorrectly positioned in the call list.

Status Codes

We can also obtain some hint as to what might be the source of the error code by looking at the status code returned for the call. As a general rule, a status code of **AR** is given when the keyword is associated with the call and a status code of **AS** is given when the keyword is invalid as a PRTO option. There might be exceptions to this rule, but in general this will hold true.

Error Codes

The following sections contain examples of mistakes and the resultant error codes provided to the application. Some length fields are omitted from the examples when not necessary to illustrate the example. Consider feedback and options lists that are shown on multiple lines to be contiguous the same way they would be found in the application's working storage.

Error Code (0002): This code indicates an invalid keyword was discovered within the call options. The error code of (0002) tells us that the keyword scan being performed is associated with keywords that are valid for the call. For example,

```
CALL = SETO
                    01
OPTIONS LIST = PRTO=04DEST(018),CLASS(A),TXTU=SET1

FEEDBACK = TXTU(0002)

STATUS CODE = AR
```

In this example, the options list contains both the keywords PRTO and TXTU. The keyword, TXTU, is not valid for the SETO call.

Another example of an error code of (0002) in the feedback is created when the length field representing the PRTO options is specified as shorter than the actual length of the options. For example,

```
CALL = CHNG
                             01
OPTIONS LIST = IAFP=N0M,PRTO=0FDEST(018),LINECT(200),CLASS(A),
                               COPIES(80),FORMS(ANS)

FEEDBACK = COPIES(0002),FORMS(0002)

STATUS CODE = AR
```

In this example, the length field of the PRTO options (that is, 001F) is too short to contain all of the options. The result of this incorrect length is that IMS finds the keywords of COPIES and FORMS outside of the PRTO options list area and indicates that these keywords are not allowed as keywords on the CHNG call.

Error Code (0004): This error code indicates that an option variable following a keyword in the options list for the CALL is not within the length limits for the option. An example of this type of error is the OUTN keyword. The name of the OUTPUT JCL statement must be from 1 to 8 characters long. For example,

```
CALL = CHNG

OPTIONS LIST = IAFP=N0M,OUTN=OUTPUTDD1

FEEDBACK = OUTN(0004)

STATUS CODE = AR
```

The operand for the OUTN keyword is 9 bytes long and exceeds the maximum value.

Error Code (0006): This error occurs when IMS is doing the scan looking for valid keywords associated with the call. IMS has encountered the PRTO keyword. Upon interrogation of the length field associated with the PRTO keyword, IMS discovers that the total length of the options list for the call is too short to contain all of the operands within the PRTO keyword. For example,

```
CALL = CHNG
               0400              05
OPTIONS LIST = 0800IAFP=N0M,PRTO=0ADEST(018),LINECT(200),CLASS(A),
                                 COPIES(3),FORMS(ANS)

FEEDBACK = PRTO(0006),LINECT(0002),CLASS(0002),COPIES(0002),
                      FORMS(0002)

STATUS CODE = AR
```

This example provides an options list that is hexadecimal, 48 (decimal 72) bytes long and the correct length for the options list. The length field of the PRTO keyword incorrectly indicates a length of hexadecimal 5A. The length of the PRTO options exceeds the length of the entire options list so the PRTO keyword is ignored and the rest of the options list scanned for valid keywords. The feedback area contains the PRTO(0006) as we would expect to indicate a length error for this keyword, but we also find that the PRTO keywords are reported to be in error (0002). This is because the keywords beyond the first PRTO keyword, up to the length specified in the options list length field have been scanned in search of valid keywords for the call. The status code of AR tells us that the keywords are considered invalid for the call and not the PRTO keyword.

Error Code (0008): This error is returned when IMS finds that one of the options for the IAFP keyword has not been specified correctly. For example,

```
CALL = CHNG
                             00
OPTIONS LIST = IAFP=N0Z,PRTO=0BDEST(018)

FEEDBACK = IAFP(0008) INVALID VARIABLE

STATUS CODE = AR
```

The message option of the IAFP keyword has been incorrectly specified as ’Z’. This results in the error code of (0008).

Error Code (000A): This error indicates that not all of the necessary keywords have been specified for this call. For example,

```
CALL = CHNG

OPTIONS LIST = TXTU=SET1

FEEDBACK = TXTU(000A)

STATUS CODE = AR
```

For this call, a valid keyword of TXTU was specified but the call also requires that the IAFP keyword be specified if the TXTU keyword is used. Since the IAFP keyword is missing, the error code of (000A) is given when the TXTU keyword is found.

Error Code (000C): The error code is reporting a condition where a set of mutually exclusive keywords have been used in the same call options list. Again, a clue to the problem being with the call options and not the PRTO options is given by issuing of the status code of **AR** and not the status code of **AS**. For example,

```
CALL = CHNG
                               00
OPTIONS LIST = IAFP=A00,PRTO=0BCOPIES(3),TXTU=SET1

FEEDBACK = TXTU(000C)

STATUS CODE = AR
```

Here we have a case where the call options list contains both the keywords of PRTO and TXTU. These options are mutually exclusive and cannot be used in the same options call list. The result is error code of (000C) returned along with status code of **AR**.

Error Code (000E): This error code indicates that while parsing the actual print data set descriptors, an error was detected with one or more of the operands. For the most part, IMS does not do any checking for these print descriptors. Instead IMS utilizes MVS/ESA services (SJF) to do the validation of the print descriptors. When SJF is called, the validation requested is the same as for the TSO OUTDES command. For this reason, IMS is insensitive to changes in output descriptors and the valid descriptors for your system are a function of the MVS/ESA release level.

You can obtain a list of the valid descriptors and the proper syntax by using the TSO HELP OUTDES command or by referring to the appropriate TSO documentation such as the *TSO Command Language Reference*.

IMS must first establish that the format of the PRTO options is in a format such that SJF services can be requested. If not, IMS returns status code **AS** and error code of (000E) and a descriptive error message. If the error has been detected during the SJF process, the error message from SJF includes information of the form, (R.C.=xxxx,REAS.=yyyyyyyy) and an error message indicating the error. The return codes and reason are further identified in the *Authorized Assembler Programming Guide*.

The range of some variables are controlled by the JES initialization parameters. Values for the maximum number of copies, allowable remote destination, classes, and form names are examples of variables influenced by the JES initialization parameters.

The following are some examples of parsing errors and the resulting error messages.

```
CALL = CHNG
                               01
OPTIONS LIST = IAFP=A00,PRTO=0BCOPIES((3),(8,RG,18,80))

FEEDBACK = PRTO(000E) (R.C.=0004,REAS.=00000204) COPIES/RG VALUE
                      MUST BE NUMERIC CHARACTERS
STATUS CODE = AS
```

For this example, the COPIES parameter has the incorrect value ’RG’ specified as one of its operands. The error message indicates that the values for these operands must be numeric.

```
CALL = CHNG
                             00
OPTIONS LIST = IAFP=A00,PRTO=0AXYZ(018)

FEEDBACK = PRTO(000E) (R.C.=0004,REAS.=000000D0) XYZ

STATUS CODE = AS
```

This example includes an invalid PRTO operand. The resulting reason code of X'000000D0' indicates the operand shown (that is, XYZ) is invalid.

This section has attempted to provide some examples of all the possible error codes that might be received by an application program. Some length fields are omitted from the examples when not necessary to illustrate the example. Consider feedback and options lists that are shown on multiple lines to be contiguous the same way they would be found in the application’s working storage.

Debugging and Diagnostic Aids Provided by IMS Spool API

In addition to providing feedback related to parsing errors, the IMS Spool API also provides other aids you can use in your diagnosis, such as the following:

- Internal trace table
- Log records
- Diagnostic information in the dependent region dump

These diagnostic aids are explained in this section.

While debugging suspected problems with either the IMS Spool API or the application using the support, keep in mind that multiple services are involved in providing the total environment. Certain JES specifications might affect which options and specifications can be used by the IMS Spool API on behalf of an application program.

Internal Trace Table

Each dependent region that uses the IMS Spool API creates a trace table that is used to trace module flow and significant events during IMS Spool API processing. This trace table is of the internal wrap around type, is always active for IMS Spool API functions, and cannot be written to an external device. It appears in any dumps produced by the dependent region. The first four words of the trace table are the header and contain the following information.

Word One
This is the trace table eye-catcher. The eye-catcher is **IWB**.

Word Two
This is the offset from the beginning of the trace table (that is, trace table header) to the last entry traced. Since the entry is an offset, relocation of the trace table does not affect the use of this word to obtain the address of the last trace entry. The offset value is added to the relocated trace table address to obtain the last trace entry. If the value is zero, no entries have been traced.

Word Three
This is the offset from the beginning of the trace table (the header) to the last trace entry in the table.

Word Four
Reserved.

Log Records Produced

The IMS Spool API produces log records to record the significant events during IMS Spool API processing. A log record of the type X'68' is written for each data set that is opened. This log record contains the information necessary for identification of the data set. If any significant event occurs during spool processing, a diagnostic log record, 67D0 is produced to record diagnostic information about the error or event. The writing of the 67D0 records is normally associated with the DFS0013E message sent to the IMS MTO for these errors.

Special Abend Processing

The IMS Spool API places control blocks in both extended common storage area (ECSA) and dependent region private storage. When a dependent region dump is produced, and IMS abnormal termination routines are allowed to execute, the following control block relocation is performed to provide diagnostic information in the dependent region dump.

The master control block for the dependent region and any active data set control blocks in ECSA are copied to the dependent region. These control blocks are copied without modification and the ECSA address of each print data set control block, IAFPDCB, is appended to the front of each relocated block.

A dummy module, DFSIAFD0, is loaded into the dependent region to serve as a place holder for the addresses of the relocated IMS Spool API control blocks. Module DFSIAFD0's address is obtained by inspecting the dependent regions Job Pack Queue for the Contents Directory Entry (CDE) that represents module DFSIAFD0. The first three words of this dummy module contain the address of the relocated control blocks as follows.

Word One
This is the address of the relocated master control block (IAFPMCB) for the dependent region. The ECSA address of the master control block is appended in front of the relocated control block area. The eye-catcher for the block is **IAFPMCB**.

Word Two
This is the address of the first relocated IMS Spool API data set control block for a print data set (IAFPDCB). When this block is copied to the dependent region, the ECSA address of the original block is appended to the front of the relocated block. This is so that the chaining of the blocks can be verified. Any additional IAFPDCB control blocks are relocated following the first relocated block with the ECSA address of each block appended to the front of each relocated block. The eye-catcher for the block is **IAFPDCB**.

Word Three
This is the address of the trace table for the IMS Spool API. The eye catcher for the trace table is **IWB**.

Service Error Log Record 67D0

The IMS Spool API creates Service Error log records, log record type 67D0, whenever a service error or unexpected condition is encountered. The 67D0 log record contains the service in error and detailed information about the system status at the time the error is detected. When problem determination is being attempted for suspected IMS Spool API errors, obtain the 67D0 log records from the IMS systems log. If the IMS Spool API issues message DFS0013E, a service error log record is also written.

In addition to the errors reported via message DFS0013E, service error log records are written if the IMS Spool API code encounters inconsistent control block structures or is unable to properly process print data sets during abend processing. These service error log records are printed using the File Select and Formatting Print Utility, DFSERA10. See the *IMS Version 7 Utilities Reference: System*, for more information on this utility program.

Some examples of events that cause Service Error log records, 67D0, to be produced are:

- Error during storage obtain/free
- Open or Close errors
- Allocation or deallocation errors
- Errors during Output Descriptor processing
- BSAM write errors
- Invalid IAFP Control Block encountered
- Unable to process print data sets due to abending dependent region

The writing of these Service Error Log Records occurs automatically.

Chapter 10. IRLM Service Aids

This chapter describes the service aids that can help you analyze internal resource lock manager (IRLM) problems. These service aids are:

- IRLM dumps
- Software LOGREC records
- MVS Component Trace

In addition, the IRLM generates diagnostic messages that begin with the prefix DXR. documents these messages.

IRLM Dumps

The IRLM uses the SDUMP system services of MVS whenever failures occur in the following situations:

- Within the IRLM address space
- While executing IRLM code or IMS code within the IMS address space
- While executing IRLM code for exits from SLM within the IMS address space

SDUMP dumps the IRLM address space to a SYS1.DUMPxx data set without formatting it. When dump processing completes, you can format the dump offline by specifying IRLM on the VERBEXIT subcommand in IPCS. If more than one IRLM is active in the system at the time the dump was taken, you must also specify the MVS subsystem name (IRLMNM in the IRLM procedure).

To access MVS component trace entries for IRLM, use the IPCS CTRACE or VERBX command. To see the syntax of the VERBX command for displaying traces, enter: IPCS VERBX IRLM 'help'.

Examples:

- If only one IRLM is in the dump, this command formats the IRLM address space:

```
VERBX IRLM 'SUBsys=IRLM'
or
VERBX IRLM
or
VERBX IRLM 'SUB=IRLM'
```

- If more than one IRLM is in the dump, this command formats the KRLM address space:

```
VERBX IRLM 'SUBsys=KRLM'
or
VERBX IRLM 'SUB=KRLM'
```

If you want to format dumps online during the abnormal termination process, you must change the FMTO= parameter to request a SNAP dump. For more information about the SDUMP support job stream and the FMTO parameters, see *IMS Version 7 Installation Volume 2: System Definition and Tailoring*.

Note: Under the direction of IBM Service, you can use the Modify DIAG command to take diagnostic dumps.

SYS1.LOGREC

The IRLM generates a software LOGREC record when the IRLM detects a program error. You can use the IFCEREP1 service aid described in *MVS/ESA Diagnosis: Procedures* to obtain a listing of the SYS1.LOGREC data set containing the LOGREC entries for the IRLM.

MVS Component Trace

Use the MVS TRACE CT command to start, stop, or modify an IRLM diagnostic trace. IRLM does not support all the options available on the TRACE command. The MVS *TRACE CT* command is described in *IMS Version 7 Command Reference* and *MVS/ESA System Commands*.

This command can only be entered from the master console. The command requires an appropriate level of MVS authority, as described in *MVS/ESA System Commands*.

The TRACE CT command lets you run the following types of sublevel traces:

DBM Trace interactions with the identified DBMS.

EXP Trace any exception condition.

INT Trace member and group events other than normal locking activity.

SLM Trace interactions with the MVS locking component.

XCF Trace all interactions with MVS cross-system coupling services.

XIT Trace just asynchronous interactions with the MVS locking component.

For EXP, INT, and XIT sublevel traces, the OFF parameter stops the traces from writing to the external writer. However they continue to write to buffers.

Example of MVS Component Trace Output

The following example shows trace output for a lock request using the DBM and SLM sublevel traces.

The command that produced this output is: `CTRACE COMP(IRLE) SUB((DBM)) FULL`

The command that produced this output is: `CTRACE COMP(IRLE) SUB((SLM)) FULL`

```
COMPONENT TRACE FULL FORMAT
COMP(IRLE)    SUBNAME((DBM))
**** 02/10/94
MNEMONIC  ENTRY ID    TIME STAMP     DESCRIPTION
--------  --------  ---------------  -----------
DBM       00000002  18:42:05.816178  RLPL format
   +0000  ID....... DXRRL100-01: START A REQUEST
   +0020  TLA1..... 000100C8  07166220
   +0028  RLPL..... 00000000  06545768  00000000  80000000  00000000
   +003C            00000000  006B12C8  008FBBC0  0090B000  00906048
   +0050            00316545  06545060  00000000  00316545  06545060
   +0064            00000000  00000000  00000000  0423AD20  09000058
   +0078            C8806D01  D7000000  00000000  00000000  00000000
   +008C            00000000  00000000  80000000  00000000  00000000
   +00A0            006B12C8  008FBBC0  02060000  8A000000  00000000
   +00B4            00000000  006B5BE4  00000000  00000000  00000000
   +00C8            00000000  00000000  00000000  00000000  00000000
   +00DC            00000000  00000000  00000000  00000000  00000000
DBM       00000002  18:42:05.816406  RLPL format
   +0000  ID....... DXRRL100-02: REQUEST COMPLETED
   +0020  TLA1..... 000100C8  07166220
   +0028  RLPL..... 00000000  06545768  00000000  80000000  00000000
   +003C            00000000  006B12C8  008FBBC0  0090B000  00906048
   +0050            00316545  06545060  00000000  00316545  06545060
   +0064            00000000  00000000  00000000  0423AD20  09000058
   +0078            C8806D01  D7000000  00000000  00000000  00000000
   +008C            00000000  00000000  80000000  00000003  00000000
   +00A0            006B12C8  008FBBC0  02060000  8A000000  00000000
   +00B4            00000000  006B5BE4  00000000  00000000  00000000
   +00C8            00000000  00000000  00000000  0067027C  A743B4E5
   +00DC            09010080  00000000  00080000  00000000  00000000
```

```
COMPONENT TRACE FULL FORMAT
COMP(IRLE)    SUBNAME((SLM))
**** 02/10/94
MNEMONIC  ENTRY ID    TIME STAMP     DESCRIPTION
--------  --------  ---------------  -----------
SLM       00000010  18:42:05.816193  RNA, RTE and UDB format
   +0000  ID....... DXRRL120-01: IXLLOCK OBTAIN
   +0020  TLA1..... 00060020  00670238
   +0028  RNA...... 09000058  C8806D01  D7000000  00000000  00000000
   +003C            00000000  00000000  00000000
   +0048  TLA2..... 000C0040  07166418
   +0050  RTE...... 0423AD20  09000058  C8806D01  D7000000  00000000
   +0064            00000000  00000000  00000000  00000000  00000008
   +0078            C9D4E2C5  40404040  0423AD20  00000000  00000000
   +008C            00000000
   +0090  TLA3..... 000B0040  071663D8
   +0098  UDB...... C9D4E2C5  40404040  00000000  00000000  00080000
   +00AC            00000000  00000000  00000000  00000000  40000000
   +00C0            08000000  00000000  A8D1A743  B4D7B281  A8D1A743
   +00D4            B4D7B281
SLM       00000020  18:42:05.816397  RNA and reason code
   +0000  ID....... DXRRL120-03: IXLLOCK RETURN
   +0020  TLA1..... 00060020  00670238
   +0028  RNA...... 09000058  C8806D01  D7000000  00000000  00000000
   +003C            00000000  00000000  00000000
   +0048  TLA2..... 00060004  0716637C
   +0050  REAS..... 00000000
```

Chapter 11. FP—Fast Path Service Aids

This chapter describes diagnostic information to help you analyze problems in Fast Path. This includes:
- Guidance information on diagnosing Fast Path problems
- DEDB CI problem assistance aids
- Descriptions of the Fast Path control blocks and tables
- A summary of Fast Path Transaction Retry

Diagnosing Fast Path Problems

Before diagnosing problems in Fast Path, you must understand the structure of its dumps, especially the dependent region dumps. When a dependent region abends, the structure of the dump varies, depending on a number of conditions. For example, if you requested and were able to perform offline dump formatting, the structure of the dump is different than if you had not requested offline dump formatting. Furthermore, if the abending dependent region was an MPP executing in mixed mode, the structure of the dump might be different from that of an IFP region. The recommended approach is to request and use the offline dump formatting option.

ABENDU1026 Analysis

Several modules issue ABENDU1026 to indicate conditions that should not occur. The dependent region abends, but the IMS control region continues processing. Message DFS2712I accompanies ABENDU1026.

This section describes an approach to analyzing ABENDU1026 failures. It tells you what documentation to obtain and guides you in finding and interpreting diagnostic data from the documentation. It is important to gather the necessary data before searching an IBM software support database or calling the IBM Support Center.

This analysis is based on using a dump that you can format with the Offline Dump Formatter (ODF). Table 67 shows you where to find ODF information.

Table 67. Locating Information about the Offline Dump Formatter (ODF)

For Information About	Refer to
Obtaining dumps suitable for input to the ODF	"Input for the Offline Dump Formatter" on page 130
Running the ODF	*IMS Version 7 Utilities Reference: System*
Using the ODF to solve problems	"Formatting IMS Dumps Offline" on page 129

Before beginning the analysis, you need:
- A copy of the DFS2712I message
- A dump formatted by the ODF
- A copy of *IMS Version 7 Failure Analysis Structure Tables (FAST) for Dump Analysis*

If an authorized program analysis report (APAR) is necessary, you might also need the following:
- The last successful image copy of the database encountering the problem
- The IMS logs from the time of the last successful image copy to the point of failure
- A copy of the Fast Path trace, if Transaction Retry was invoked

Procedure

The following example takes you through the analysis of an actual ABENDU1026 until you have collected enough data to search an IBM software support database or call the IBM Support Center.

This example uses the sample message DFS2712I in Figure 120. DFS2712I is sent to the console. Be sure to save a hard copy of the message.

```
DFS2712I    MODULE NAME:    DBFMRCU0
DFS2712I    ABEND SUBCODE:  0053
DFS2712I    AREA NAME:      DB21AR0

DFS2712I    MLTE:
DFS2712I    02A923BC  02919E60 00000000 00000000 00001008
DFS2712I    02A923CC  02903310 00005A08 00001008 00040400
DFS2712I    02A923DC  03018000 001C0008 029328B4 00060000
DFS2712I    02A923EC  00000000 00000000 00000000 02A92178
DFS2712I    02A923FC  02A92470 0072F70A 00000000 40800000
DFS2712I    02A9240C  00000000 00000000 00000000 00000000
DFS2712I    02A9241C  00000000 00000000 00060000 00000000
DFS2712I    02A9242C  00000000

DFS2712I    BUFFER CONTENTS:
DFS2712I    02919E58  016C0802 40000000 99000000 5C08015E
DFS2712I    02919E68  C1C140E3 C8C9E240 C9E240E3 C8C540C6
DFS2712I    02919E78  C9D9E2E3 40F3D9C4 40D3C5E5 C5D340E2
DFS2712I    02919E88  C5C7D4C5 D5E34040 40404040 40404040
DFS2712I    02919E98  40404040 40404040 40404040 40404040
  .    .   .    .    .    .    .    .    .    .    .    .    .    .

DFS2712I    R0-R3   00000008 00000053 02919E60 02A92010
DFS2712I    R4-R7   02A923BC 008138D4 00000008 00005A00
DFS2712I    R8-R11  00000004 02903310 0070B040 0086DF20
DFS2712I    R12-R15 00818BA0 0070767C 80818C62 00000018
```

Figure 120. Example of Message DFS2712I

Use the following steps to analyze ABENDU1026:

1. Locate the module name and subcode associated with the abend. This information appears in the first few lines of message DFS2712I.

   In the example in Figure 120, the module name is DBFMRCU0 and the subcode is 0053.

2. To find the meaning of the subcode, look up ABENDU1026 in *IMS Version 7 Failure Analysis Structure Tables (FAST) for Dump Analysis*. Find module DBFMRCU0 and subcode 0053.

   The description of subcode 0053 is:

   > MLTE segment code (Reg4 + X'1E') is not equal to the DSEGCODE of the segment pointed to by register 2.

   This means that the segment code in field MLTESGCD in MLTE (a Fast Path control block) does not match the segment code of the segment in the buffer (DSEGCODE). Therefore, your next step is to determine what the mismatched values are.

3. Turn to *IMS Version 7 Failure Analysis Structure Tables (FAST) for Dump Analysis* again to determine which registers you must examine.

   The important registers are:

   Register 8 = MLTESGCD
   Register 2 = Address of the segment; DSEGCODE is the first byte

   In Figure 120, the register contents appear at the bottom of message DFS2712I.

4. Use the registers and the buffer contents in the message to compare the segment code in the segment in the buffer (DSEGCODE) with the segment code in field MLTESGCD in the MLTE. These codes must match.

   - Register 8 contains the segment code from field MLTESGCD in the MLTE. In the example, register 8 has a value of 00000004.

- Register 2 contains the address of the segment in the buffer. The first byte of the segment is the segment code (DSEGCODE). In the example, DSEGCODE has a value of 99.
- Because the segment code from the MLTE (04) does not match the segment code of the segment (99), ABENDU1026 occurred.

There are several ways to find this data. To find the segment code in field MLTESGCD in MLTE, you can also use register 4 + X'1E'. To find the DSEGCODE, you can also use register 6 (00000008), which is the offset in the buffer to the DSEGCODE.

5. You must now look at the module save area set to determine the module flow leading to the abend. You can use the Offline Dump Formatter (ODF) to format the save area set in a dump by specifying `FMTIMS DB,MIN`. Figure 121 shows an example of the save area set formatted by the ODF.
   - Register 13 in message DFS2712I contains the address of the save area for the PST that suffered the abend.
   - In the example message in Figure 120 on page 342, register 13 contains the address 0070767C.
   - In the `**DPST` section of the formatted dump in Figure 121, search for a save area (SA) with address 0070767C. If you are searching online, the second occurrence you find is the actual save area.

```
***SAVE AREA SET***

EP   DBFMCLX005/06/8804.27PL24768 ABCD
SA  0070755C       WD1 8071B310     HSA 80000000     LSA 007075A4     RET 8088070E     EPA 00812FE0      R0 00000519
                    R1 8071B310      R2 C7D5D740      R3 02A92010      R4 0001A000      R5 00707050      R6 00000000
                    R7 8072F624      R8 00707050      R9 0072F6CC     R10 0070B040     R11 0086DF20     R12 00880042
EP   DBFMGNX003/03/8820.09PL22770 AB
SA  007075A4       WD1 00000000     HSA 0070755C     LSA 007075EC     RET 808131A0     EPA 00814528      R0 00000519
                    R1 8071B3AB      R2 C7D5D740      R3 02A92010      R4 02A92090      R5 008138D4      R6 FFFFFD80
                    R7 FEE06FD4      R8 00707050      R9 0072F6CC     R10 0070B040     R11 0086DF20     R12 00812FE0
EP   DBFMPUG005/11/8800.59PL26682 ABCDE
SA  007075EC       WD1 00000000     HSA 007075A4     LSA 00707634     RET 8081466A     EPA 00816900      R0 00000519
                    R1 8071B3AB      R2 00000000      R3 02A92010      R4 02A92178      R5 008138D4      R6 FFFFFD80
                    R7 FEE06FD4      R8 00707050      R9 0072F6CC     R10 0070B040     R11 0086DF20     R12 00814528
EP   DBFMRCU003/21/8618.02PTO1119 0
SA  00707634       WD1 00000000     HSA 007075EC     LSA 0070767C     RET 80816ABE     EPA 00818BA0      R0 00000519
                    R1 8071B3AB      R2 02A92178      R3 02A92010      R4 02A923BC      R5 008138D4      R6 FFFFFD80
                    R7 00005A08      R8 0291AE66      R9 0072F6CC     R10 0070B040     R11 0086DF20     R12 00816900
EP   DBFMPGO002/04/8617.58PP35272 1B
SA  0070767C       WD1 00000000     HSA 00707634     LSA 007076C4     RET 80818C62     EPA 00818FD8      R0 00000008
                    R1 8071B3AB      R2 02919E60      R3 02A92010      R4 02A923BC      R5 008138D4      R6 00000008
                    R7 00005A00      R8 00000004      R9 02903310     R10 0070B040     R11 0086DF20     R12 00818BA0
EP   DBFMSRB002/13/8716.56PP58251 AB
SA  007076C4       WD1 00000000     HSA 0070767C     LSA 0070770C     RET 80822377     EPA 008285F0      R0 FFFF4040
                    R1 02903310      R2 02932A08      R3 02903278      R4 808222E0      R5 00822638      R6 00005A00
                    R7 00BBCF78      R8 02932A08      R9 02903310     R10 0070B040     R11 0086DF20     R12 008221B8
EP   DBFXSL3007/08/8819.02PL28384 AB
SA  0070770C       WD1 00000000     HSA 007076C4     LSA 00707754     RET 808286D7     EPA 00823D38      R0 00000000
                    R1 0070B040      R2 02932A70      R3 02903278      R4 02903310      R5 0071A250      R6 00005A00
                    R7 00BBCF78      R8 02932A08      R9 02903310     R10 0070B040     R11 0086DF20     R12 008285F0
```

Figure 121. Example of a Save Area Set

6. In Figure 121, the module flow, reading from the top down, is DBFMCLX0, DBFMGNX0, DBFMPUG0, and DBFMRCU0, which is where the abend occurred. Notice that other modules follow DBFMRCU0 in the flow. You can ignore these modules now. However, they might be important later in the problem analysis.
7. Information from other sources might help you while searching the IBM software support database or talking with the IBM Support Center representative.

   If an MPP or an IFP received the ABENDU1026, the Transaction Retry function should have retried the transaction. (For information about this function, see "Fast Path Transaction Retry" on page 344.)

   Look in your MTO log for messages DFS0663I, DFS0784I, DFS0785I, DFS0787I, and other messages associated with a retry to find out what happened.

At this point you have most of the following information:

The abend code (ABENDU1026).

The subcode (SUBCODE053).

The module name (DBFMRCU0).

The save area flow leading to the abend.

The field in error (MLTESEGCD or DSEGCODE). You might not be sure which field is incorrect.

Any messages produced by a transaction retry (for example, MSGDFS0663I).

With this information you are ready to search the database or contact the IBM Support Center.

Fast Path Transaction Retry

Fast Path Transaction Retry (FPTR) is designed for IMS Fast Path users who cannot run the Fast Path trace permanently on their system because of its impact on performance, but want to have the trace turned on when Fast Path failures occur. Fast Path problems can be resolved much faster when trace information is available to show the logic flow of a call or transaction.

FPTR is activated only when certain Fast Path failures occur. FPTR automatically allocates a trace data set, turns on the trace, and retries the transaction. If no abend occurs on the retry, FPTR issues a message, turns off the trace, and the system continues processing. If an abend does occur on the retry of the transaction, Fast Path trace writes the trace data, FPTR turns off the trace, and the system continues with Fast Path trace inactive. FPTR is not invoked for abends in BMP regions.

When you report certain IMS Fast Path problems to the IBM Support Center, you will be asked if the Transaction Retry function failed. The following sections will help you determine what information to report.

Processing Flow

A summary of the processing flow of FPTR follows:

- The ESTAE exit of the dependent region controller receives control for abends U1026 and U1027, and all system abends except 122 and 222.
- The ESTAE exit provides debugging information including:
  - Name of abending module
  - Last applied APAR of the abending module
  - Date and time of assembly of module

  If the failing module cannot be identified, a message informs the operator.
- The ESTAE exit decides if the transaction can be retried. If so, the ESTAE requeues the failing input message for retry and produces a dump of the first abend.
- Message DFS554A is sent to the master terminal.
- The retry process starts in an eligible dependent region.
  - FPTR dynamically allocates a trace data set and starts Fast Path trace.
  - FPTR writes message DFS0785A to the master terminal and the JES2 job log. (See *IMS Version 7 Messages and Codes, Volume 1* for an explanation of the message.)
- When the retry of the transaction is complete, FPTR deallocates the trace data set and spools the contents of the trace data set to the SYSOUT class specified in the MSGCLASS parameter on the JOB statement of the dependent region.

What the System Programmer Should Do

The system programmer should:

- Print the job log.
- Print the spooled trace data set information.
- Save and analyze the above information.

- Contact the IBM Support Center for assistance, if needed.

DEDB Control Interval (CI) Problem Assistance Aids

After you have performed the analysis described in "ABENDU1026 Analysis" on page 341, you will need to review the contents of the various control blocks. Included in message DFS2712I is a dump of the control block that is related to the logical inconsistency. This control block is in the format of one of the control intervals (CIs) that are listed in this section. You can (maybe with help from the IBM Support Center) obtain the RBA of the affected CI from the buffer. You can then use this RBA:

- When you extract the CI from the image copy of the DEDB
- When you choose the criteria for selecting and printing the IMS log records (with DFSERA10)

Related Reading: For information about choosing which log records to analyze, see "Log Records" on page 113.

This section describes the structure of various CIs as they appear in a dump. When you print portions of the DEDB, the CIs have the identifying characteristics listed below.

Some of the acronyms used in this section are:

DOVF
Dependent overflow

IOVF
Independent overflow

RAP BLOCK
Root-anchor point block

SDEP
Sequential dependent

CI Type Identification

Each CI has an identifier at X'02' in the CI, with the exception of the first and second CIs. The first is the IMS control CI and the second contains the DMAC control block for this Area.

CI Type	Identifier
REORG CI	00
RAP	01
DOVF	02
IOVF (SPACE MAP)	04
IOVF	08
SDEP	10

DEDB CI Formats

This section first discusses the details of the various CI types, and then describes the data common to all CIs (except the SDEP CI).

CI 0

This is the IMS control CI.

```
0          8          10           18       1C   20     28      32
Creation   Restart    EREstart     RBA of   Characters  Cisize  Org
Date/Time  Date/Time  Date/Time    Last CI  DBF1.000    - 7     "D"
```

CI 1

The DMAC control block for this area is located here.

The Error Queue Element (EQE) list is also located in this CI. This list is 44 bytes long and immediately precedes the trailer information, (for example, CUSN, RBA, RDF and CIDF). The following diagram shows the EQE list format.

	FLG *	EQE CNT	EQE ENTRY	....	EQE ENTRY
bytes	1	3	4		4
			- - - - - - - - 10-Entries - - - - - - - -		

* X'80' means more than 10 EQEs or error in 2nd CI.

Figure 122. EQE list in CI 1

RAP CI

```
0        2       4                8
FSEAP    0102    RBA of 1st       Segments, FSEs and Scraps
                  root
          |
          v
       (02) - Indicates CUSN
              is in this CI
```

Figure 123. RAP CI

First DOVF CI

The first DOVF CI has this format.

```
0        2       4                8
FSEAP    0203    RBA of current   Segments, FSEs and Scraps
                 overflow CI
          |
          v
       (02) - Same as RAP CI      ---> these two bits combined
       (01) - Look here for space ---> make the 03 in byte 3.
```

Figure 124. First DOVF CI

Exception: From here on, the key bits are shown, but byte 3 is not shown.

Other DOVF CIs

All DOVF CIs except the first one have this format.

```
0       2     4              8
FSEAP   02    RBA of next    Segments, FSEs and Scraps
              DOVF CI with
              space, last
              contains zeros
```

Figure 125. Other DOVF CIs

First IOVF CI

This CI is a space map and is the first in each group of 120 CIs. The 119 CIs that follow are data CIs.

```
0       2    4   6          8 (119 words mapping next 119 CIs)
0000    04   8000xxxx offset  8000xxxx free and offset to next free
                      to 1st  4000uow# allocated
                      free    2000uow# used by reorg
             40000000 no free space in this space map CI
```

Figure 126. First IOVF CI

Other IOVF CIs

This is a data CI - 119 data CIs follow each space map CI.

```
0       2     4              8
FSEAP   0802  4000uow#       Segments, FSEs and Scraps (allocated,
                             to UOW number; 0 is the first UOW).
0008    0802  80000000       FSE (CI not allocated).

         (02) indicates CUSN is in this CI
```

SDEP CI

Exception: SDEP CIs do not contain FSEs and have no FSEAP or CUSN. User segments have a time stamp added at the end.

```
0         2 3     4              12
0000      1000    Partner name   Segments inserted sequentially and
                                  cannot be updated

          (01) - Time stamp exists.
          (04) - SDEP CI is full.
```

Figure 127. First DOVF CI

FSEAP

FSEAP is the offset of the first FSE in the CI. Fast Path FSEs are chained from the highest RBA, in order, to the lowest RBA in the CI.

```
FSE---X'8offssss'  off=offset of next FSE in CI
                   ssss=size (length) of the free space
                   including the FSE.

X'8000ssss'  indicates this is the last FSE on the chain in
             this CI.
```

If the CI is empty, the FSE is X'15' bytes less than the CI size, or X'13' less than the CI size if no CUSN exists. The RDF and CIDF are X'7' bytes less than the CI size. Here are some examples:

```
CI    512  X'200'   1024  X'400    2048  X'800'   4096  X'1000'

FSE   800001EB      800003EB       800007EB       80000FEB
RDF   0001F9        0003F9         0007F9         000FF9
CIDF  01F90000      03F90000       07F90000       0FF90000
```

Scraps

Scraps are less than 4 bytes. They begin with X'7n' if less than 8 segment types, or X'Fn' if more than 8. For example,

```
1 byte—X'71' or X'F1'
2 bytes—X'72' or X'F2'
3 bytes—X'73' or X'F3'
```

Data Common to All CIs

The last X'0D' bytes of a CI all have the same use. The last line of a CI looks like this in a dump.

```
data  data  data  data  data
                              -D -C-B-A-9 -8-7-6-5 -4-3-2-1
x-x   x-x   x-x   x-x   xxxxxxxx xxxxxxxx xxbbbbbb bbbbbbbb
```

The bytes with bbbbbs do not print and will show as blanks in the dump. The fields from -D to -1 are:

```
CUSN  -D,C      These 2 bytes represent updates to the CI. The 02
                bit in byte 3 of a CI indicates a CUSN exists in the CI.

RBA   -B,A,9,8  These 4 bytes are the beginning RBA of the CI.

RDF   -7,6,5

CIDF  -4,3,2,1
```

Recommendation: Use the RBA of the CI when you select log records to format and print with the DFSERA10 utility.

SDEP CIs do not contain FSEs and do not have a CUSN. SDEP CIs end at -B (the RBA). Data can occupy the space up to that location.

Analyzing Control Interval (CI) Contention

When CI contention occurs in a DEDB, Fast Path passes both lock requests to program isolation (PI) modules. The PI trace, if active, traces the locks. To format the PI trace records (log record type X'67FA'), use the File Select and Formatting Print utility (DFSERA10) with exit DFSERA40. For information about running this utility, see *IMS Version 7 Utilities Reference: System*.

Using the trace records, find the RBA field of the CI. The digits in the CI RBA field are shifted right 8 bits. For example, an RBA of 00468000 is displayed as 00004680.

You must translate the value in the DMB field to a relative DMAC number. (DMAC numbers are relative to the DATABASE definitions.)

For example, if the first DMAC is X'FFFE', then the second DMAC is X'FFFD', the third DMAC is X'FFFC', and so forth. Since databases are chained alphabetically in the DDIR, if the DMB field is X'FFF6', you would calculate the relative DMAC number as follows:

```
X'FFFF' – X'FFF6' = X'19' = 25 (decimal)
```

This means that X'FFE6' is the 25th Area relative to the first Area of the first DEDB in the DDIR.

Locating Fast Path Control Blocks and Tables

Many of the Fast Path control blocks are extensions of IMS full-function control blocks. The names of these Fast Path control blocks are the same as in full-function. The acronyms for these Fast Path control blocks start with “E”.

Example:

SCD System Contents Directory (full-function IMS)

ESCD Extended System Contents Directory (Fast Path)

| To view the layout of the Fast Path control blocks for your system, assemble DFSADSCT from
| IMS.ADFSSRC. Remember to use XREF(FULL).

Table 68 shows the Fast Path control blocks and work areas that appear as a load list in an IMS dump.

This information is especially relevant when you are working on an abend U1011 in module DBFINI20; message DFS2703A generally accompanies the abend. This abend results from either a GEN problem or a storage fragmentation problem.

At Fast Path initialization, module DBFINI20 calculates the amount of contiguous ECSA storage that is needed in order to load DBFCONT0, which contains the buffers, buffer headers, MSDBs, and other related control blocks. If DBFINI20 cannot obtain a large enough contiguous block of storage, abend U1011 is issued.

When this occurs, you can try doing an IPL, or you can stop other jobs and perhaps free up whatever was preventing DBFINI20 from obtaining the necessary storage.

You can look in register 8, which contains the amount of storage DBFINI20 was trying to obtain. This amount is the accumulated total sizes of the blocks needed by Fast Path. If you receive abend U1011 again, you can quickly perform the following calculation:

```
buffers x buffer size + MSDB_size
```

If the amount you calculate is close to the value in register 8, you can be fairly sure that IMS performed the calculations correctly; this means that the problem is with storage fragmentation.

Refer to Table 69 when you are figuring out which specific control blocks are needed in your Fast Path environment.

Table 68. Fast Path Control Blocks and Work Areas that Appear in IMS Dumps

Load List Name	Fast Path Block/Work Area	Appearance in Dump
DBFCONT0	Fast Path Global Control Blocks	IMS STM Task
DFSEPnnn	Fast Path EPSTs (nnn=000-999)	IMS STM task

The possible control block structure of DBFCONT0 appears in Table 69.

Table 69. Control Block Structure of DBFCONT0

Control Block/Table	With MSDB/DEDB	Without DEDB	Without MSDB	Without DEDB/MSDB
ECNT	X	X	X	X
BHDR	X	X		
MSDB	X	X		
DMHR	X	X	X	
BUFF	X	X	X	
DMCB	X		X	
OTHR	X		X	
BALG	X	X	X	X
MBUF	X	X	X	X
LBUF	X	X	X	X
FPAL	X		X	

If you use online formatting, only the first 16 MB of DBFCONT0 are dumped.

Chapter 12. MSC—Multiple Systems Coupling Service Aids

This chapter includes descriptions and diagnostic hints to help you diagnose multiple systems coupling problems. It does not apply to a Database Control (DBCTL) environment. Included are:
- A description of the various entry points in the device-dependent modules
- An MSC communication task trace
- A description of MSC coupling traces
- Diagnosing link problems
- A channel-to-channel access method trace stack (LXB trace)

Multiple Systems Coupling Communication Task Trace

The flow through an MSC communication task is very similar to that through the terminal communication task. The register 0 trace is read in exactly the same manner, and most of the MSC analyzer and MSC DDM entry points provide the same functions as the terminal communications analyzer and DDMs. The entry points for the MSC analyzer and DDMs are:

DDM Entry Point	ANALYZER
AM01	Process input from a link
AM02	Perform read or read of the link
AM03	Determine what to do next on the link
AM04	Not used
AM05	Perform write or send to the link
AM06	Dequeue the message after a good write or send
AM07	Not used
AM08	Return a message to the message queues for later transmission
AM09	Generate an error message
AM10	Quiesce the link
AM11	Not used
AM12	Wait for the completion of asynchronous I/O or the enqueue of a message

Multiple Systems Coupling Device-Dependent Module

An MSC device-dependent module (DDM) performs all of the functions unique to a type of link. The functions the DDM performs at each entry point are:

DDM Entry Point	MSC
DM01	Setup output buffer for a write or send operation
DM02	Error check last output operation
DM03	Setup to obtain input from the link
DM04	Error check an input operation
DM05	Not used
DM06	Not used

DM07 Connect or disconnect the link

DM0I An access method is entered from the DDM

Several entry points are not used to preserve a commonality between coupling communication and terminal communication functions.

Figure 128 summarizes the MSC communication task trace.

Traced By	Entry Point	Function	Trace Ident	When Traced	/TRACE Option												
DFSCMS00	DFSCIO01	Process Input	AM01		X	X		X	X		X	X	X				X
DFSCMS00	DFSCIO02		AM02														X
DFSCMS00	DFSCIO03	What Next?	AM03		X	X		X	X		X						X
DFSCMS00	DFSCIO05		AM05														X
DFSCMS00	DFSCIO06	After Good Write	AM06		X	X		X	X	X	X						
	DFSCIO08	Wash Message	AM08		X	X		X	X		X		X				
	DFSCIO09	Generate Message	AM09		X	X		X	X		X						X
	DFSCIO10	Quiesce Link	AM10		X	X		X	X		X						
	DFSCIO12	Wait for I/O or Message Enqueue	AM12		X	X		X	X		X		X				
	DFSCIOC0	Get Work Buffer	CM00		X			X	X		X				X		
		Reposition Queue Buffer	CM01		X			X	X		X				X		
		Get Next	CM02		X			X	X		X				X		
		Dequeue Message	CM03		X			X	X		X				X		
		Wash Output	CM04		X			X	X		X				X		
		Find Output	CM05		X			X	X		X				X		
		Get New Output	CM06		X			X	X		X				X		
		Free Input Queue Buffer	CM07		X			X	X		X				X		
		Free Work Buffer	CM08		X			X	X		X				X		
DFSCMS80	DFSCMS80	Abort Processing (First LTB)	MSS1	X			X	X	X	X	X	X	X	X			
DFSCMS80	DFSCMS80	Abort Processing (Second LTB)	MSS2	X				X						X		X	
DFSCMS81	DFSCMS81	Prior to DDM I/O	DM0I		X	X		X	X		X		X				
DFSCMS00	DFSCIO03;06	Write Setup	DM01		X	X		X	X	X	X	X	X				
	DFSCIO00	Write Interrupt	DM02		X	X	X	X	X		X		X		X		X
	DFSCIO01;03	Read Setup	DM03		X	X		X	X		X						
	DFSCIO00	Read Interrupt	DM04		X	X	X	X	X		X		X		X		X
	DFSCIO00;03	Connect/Disconnect I/O Interrupt	DM07				X	X	X		X		X		X		X
DFSCMEI0	DFSCMEI0	Message Control/Error exi processing	CMEI		X									X			
DFSCMEI0	DFSCMEI0	Before calling Message Control/ Error exit DFSCMUX0	CMEA		X									X			
DFSCMEI0		After calling Message Control/Error exit DFSCMUX0	CMEB		X									X			

Figure 128. Multiple Systems Coupling Communication Task Trace

Multiple Systems Coupling Traces

This section covers the following MSC traces:

- Message Processing Trace
- Main Storage Access Method Trace
- Main Storage-to-Main Storage Save Set Trace

MSC Message Processing Trace—BUFMSTRA

The MSC message processing trace records the SYSIDs of the last four IMS systems that processed the MSC message (that is, a BMP or MPP issued a GET UNIQUE to the message queue). The trace is located in the MSC message prefix at label BUFMSTRA within the BUFMS DSECT. The trace contains up to four 1-byte SYSID entries. The low-order byte contains the most recent entry. The initial entry contains the SYSID of the system to which the inputting terminal is attached. Each additional entry results in a shift left (the high-order byte is shifted out).

In Version 6, the SYSID is increased to two bytes and is traced in field MSGMETRA of the MSC extension in DSECT MSGMSCE. If the SYSID is less than 256, it is traced both in field BUFMSTRA and MSGMETRA for compatibility. If the SYSID is greater than 255, it is only traced in MSGMETRA; field BUFMSTRA contains zeros.

Main Storage-to-Main Storage Access Method Trace

The main storage-to-main storage access method trace records information related to the main storage-to-main storage access method, DFSMTMA0, and the main storage-to-main storage device-dependent module, DFSDN540. The trace is located in global storage pointed to by the "MTMWINDOW" and copied to module DFSMTMTR during abend processing. The following locates the trace:

TTOP—Table beginning

TPTR—Next entry to be used

TBOT—Table end

The trace is a wraparound trace. Each entry is 192 bytes long and contains information such as function, return code, and control blocks. The TRACEMAP DSECT contains further details on entry contents. TRACEMAP is embedded in macro INTFMTMA. Trace operation is controlled by a global SETC labeled within DFSMTMA0. The default assembly value is ON.

Main Storage-to-Main Storage Save Set Trace

DSECT SAVWORK describes a key work area used by DFSMTMA0. This work area is chained into the standard IMS save set chain with a SAVE ID of MTMWORKAREA. The trace appears in the save set chain even when the trace is set. The SAVWORK DSECT is embedded within macro INTFMTMA.

Diagnosing Link Problems

Set TRACE on for appropriate lines from the IMS master terminal. Trace all terminals on a line. For example, use:

```
/TRACE SET ON LEVEL 4 MODULE ALL LINK
/TRACE SET OFF LINK x
```

For diagnosing link problems, the trace records with the following identifiers are helpful.

AM01 RECEIPT OF DATA FROM PARTNER SYSTEM

Entry 1 is invoked when data other than a link level status message (that is, 'LINK STOPPED') is received.

Assemble a copy of DFSADSCT, and refer to the BUFMS DSECT in the listing.

I TP BUF
Contains the segments received.

BUFTFLAG
Indicates more about what was received (that is, first segment).

O TP BUF
Contains the data set last sent to the partner.

Q BUF
Contains the segments received so far.

I WP BUF
Contains the MSC prefix/work buffer.

O WP BUF
Contains the MSC prefix/work buffer.

AM02 ERROR - CHECK LAST OUTPUT OPERATION

I WP BUF
Contains the MSC prefix/work buffer.

O WP BUF
Contains the MSC prefix/work buffer.

AM03 MSC ANALYZER 'WHAT NEXT'

If this entry is invoked from a DDM, it is because the DDM has nothing else to do.

Example: EOT received to ACK. Neither side sending; therefore, let the analyzer decide what to do.

Example: A data block containing only the message prefix was received (no segment could fit in the remaining buffer space). DDM goes to AM03 because there might be output that can be sent. Data response to data is okay.

If this entry is invoked from another analyzer entry point, it is because that function is complete.

Example: After the dequeue of an output message, ENTRY 6 goes to AM03 to see if more output can be initiated.

CLBCNTQB
Is a QCB for a destination that has messages queued to be sent across the link.

CLB3INP and/or CTBAINP
Indicates that the DDM is not able to send any output data.

CTBAERR
Indicates that an error message is to be sent to the partner.

I WP BUF
Contains the MSC prefix/work buffer.

O WP BUF
Contains the MSC prefix/work buffer.

AM05 MSC ANALYZER ENTRY 5

This entry is invoked from DDM to send out a message.

O TP BUF
Contains the data last sent to the partner.

I WP BUF
Contains the MSC prefix/work buffer.

O WP BUF
Contains the MSC prefix/work buffer.

AM06 LAST OUTPUT OPERATION SUCCESSFUL

This entry is invoked from DDM when the previous output was successful.

CTBAEOM=1
Indicates that the previous output included the last piece of the message, and that the message is to be dequeued.

CTBAEOM=0
Indicates that the last piece of the message has not been sent. No dequeue is to take place. The DDM is dispatched at DM01 to attempt to continue transmitting.

AM08 CANCEL MESSAGE ENQUEUE OPERATION

There is a probable contention situation, and this partner must yield. The output message in progress is returned ("washed back") to the queues to be sent later.

O TP BUF
Contains the data that the DDM was attempting to transmit.

AM09 GENERATE AN ERROR MESSAGE

I WP BUF
Contains the MSC prefix/work buffer.

O WP BUF
Contains the MSC prefix/work buffer.

AM10 LINK SHUTDOWN: OPERATOR INTERVENTION REQUIRED

This entry is invoked because the link is PSTOPPED (either via /PSTOP or I/O error). If the entry is invoked from DDM it is because the DDM has detected a condition that prevents anything more from being done. Find the previous DDM interrupt entry (DM02, DM04 or DM07) to determine why the DDM went to AM10.

General cleanup is performed: Queue buffers and I/O buffers are released.

AM12 NORMAL 'LINK IDLE' CONDITION

This entry is invoked when DDM has nothing else to do under normal conditions.

Example: MTM link is attention driven. There is no outstanding READ as with BSC. When the DDM has no more to do (no more data to send and no pending acknowledgment), it becomes idle to wait for a POST by either the enqueue of output or an attention from the partner. This entry is different from AM10 in that the analyzer does not take it upon itself to perform a general cleanup.

CM00 GET A WORK BUFFER

This analyzer entry is called when the DDM needs additional space to perform message editing. An example is the collecting of all pieces of a SPA.

CM01 REPOSITION QUEUE BUFFER

This analyzer entry is called when the DDM wants to ensure that the queue buffer is in storage. This entry is currently not used.

CM02 GET NEXT

This analyzer entry is called when the DDM needs the next output segment of a message.

CM03 DEQUEUE MESSAGE

This analyzer entry is called when the DDM wishes to dequeue a message (rather than let the analyzer do it). An example is the emergency restart of a link. The DDMs exchange message sequence numbers. If one DDM determines that a message in its queues has already been received by the partner, the message is dequeued to prevent it from being sent twice.

CM04 WASH OUTPUT MESSAGE

This analyzer entry is called when the DDM wants to return an in-process message to the queues. An example is a permanent I/O error. The DDM washes any output in progress so that it will be resent after the error recovery sequence completes.

CM05 DETERMINE IF QUEUED OUTPUT IS PRESENT ON A LINK

This analyzer entry is called when it must be determined if there is any (more) queued output to be sent across the link emergency restart processing. If one DDM determines that a message in its queue has already been received by the partner, the DDM does a GU (for positioning) followed by a DEQUEUE (CM03) to get rid of the message.

CM07 FREE INPUT QUEUE BUFFER

This analyzer entry is called when the DDM wants to cancel an input queue buffer. An example is permanent I/O error. The DDM throws away all input segments that, up to the point of failure, have been collected in queue buffers. The message is lost on this system, and the ABORT sequence sent to the partner tells the partner that the message must be sent again later.

CM08 FREE A WORK BUFFER

This analyzer entry is called when the DDM wants to free an extra work buffer. This entry is currently not used because the buffer mentioned in the CM00 description is automatically freed by the analyzer.

DM01 WRITE SETUP

The DDM is entered here when the MSC analyzer finds output to be sent and the link is available (CLB3INP off).

Assemble a copy of DFSADSCT, and refer to the BUFMS DSECT in the listing.

Q BUF
Contains the segments to be sent.

O TP BUF
Contains the data stream ready to be sent.

I TP BUF
Contains any data received from the partner.

DM02 WRITE INTERRUPT

The DDM is entered here at the completion of a logical write operation.

DECSDECB
Contains the completion code.

BUFTYPE
Contains more information about the type of completion (MTM).

O TP BUF
Contains the data stream sent to the partner.

I TP BUF
Contains any data received from the partner.

I WP BUF
Contains the MSC prefix/work buffer.

O WP BUF
Contains the MSC prefix/work buffer.

DM03 READ SETUP

The DDM is entered here when the MSC analyzer determines there is no output that can be sent. MTM and CTC are attention driven, and no I/O is initiated here.

DM04 READ INTERRUPT

The DDM is entered here at the completion of a logical read operation.

DECSDECB
Contains completion code.

BUFTYPE
Contains more information about the type of completion (MTM).

DECTYPE
Indicates the type of the last operation.

I TP BUF
Contains the data just read.

O TP BUF
Contains any data sent to the partner in response to a previous read completion.

I WP BUF
Contains the MSC prefix/work buffer.

O WP BUF
Contains the MSC prefix/work buffer.

DM07 RESTART

The DDM is entered here from the MSC analyzer whenever the link is not active (CRB1ACT is not equal to X'11').

DECTYPE
Indicates the type of the last operation attempted.

DECSDECB
If I/O is completed, this indicates status.

I TP BUF
Contains the last data read.

O TP BUF
Contains the data to write or the data last written.

I WP BUF
Contains the MSC prefix/work buffer.

O WP BUF
Contains the MSC prefix/work buffer.

DM0I ENTRY TO ACCESS METHOD

This record is traced at entry to the access method from the DDM.

DECTYPE
Indicates the type of operation.

O TP BUF
If output, contains data to be written.

MSS1 and MSS2 Records

These records are created as a result of ABORT processing when an I/O error (either correctable or not) occurs. All available control blocks are SNAPed, regardless of any /TRACE options in effect on the link involved. These records are followed by a type 03 record containing the message that was sent to the master terminal as a result of the error.

Significant Fields:

Table 70. Significant Fields in MSS1 and MSS2 Records

Field	Description
BSC	POST code (first byte of LLB) DECTYPE DECFLAGS DECERRST DECRESPN IOB I/O buffers (data and response)
MTM	POST code (first byte of LLB) DECTYPE I/O buffers (data and response)
CTC	POST code (first byte of LLB) DECTYPE IOSB I/O buffers (data and response) LBX
VTAM	POST code (first byte of LLB) DECTYPE I/O buffers (including RPL)

Channel-to-Channel Access Method Trace Stack (LXB Trace)

The LXB trace stack is designed to be used in conjunction with the module listings to provide a detailed trace of instruction flow through the channel-to-channel (CTC) access method. The trace stack is located in the LXB at label LXBCTRAC, 288 (X'E4') bytes into the LXB, and is 50 bytes long. The only modules that manipulate the LXB trace stack are the CTC access method modules, DFSCMC00, DFSCMC10, DFSCMC40, and DFSCMC50. The code that manipulates the LXB trace stack is unconditionally operative. (That is, it is not conditionally assembled and the function is not controlled by the operator command.) If level 3 or 4 of the IMS trace command is in effect, the LXB is included among the areas traced to the log.

Most LXB trace stack entries are 2 bytes long; a few are 1 byte long. Usually, each invocation of one of the access method modules causes a trace entry to be placed in the LXB trace stack. In order to create a trace entry, the module first moves (pushes) the trace stack 2 (or 1) bytes backward (toward low storage), thereby deleting the oldest portions of the trace stack. The module then inserts the new entry at the high (storage address) end of the trace stack. In rare instances, when the asynchronous modules DFSCMC40 and DFSCMC10 interrupt execution of another CTC access method module, the trace entries might overlap and thus might not be meaningful.

The format and meaning of the possible LXB trace entries follow:

Byte 1, bit 0
If on, this is a 2-byte entry; otherwise it is a 1-byte entry.

Byte 1, bits 1-3
This identifies the module and, if applicable, the routine within the module that made the entry in the LXB.

Value	Meaning
1	DFSCMC40, attention DIE routine
2	DFSCMC10, channel-end appendage
3	DFSCMC10, abnormal-end appendage
4	DFSCMC40, I/O request DIE routine
5	DFSCMC10, shutdown appendage
6	DFSCMC50, shutdown processing routine
7	DFSCMC00, MSC analyzer

Byte 1, bits 4-7
This identifies what processing was performed. The meaning of the bits, as shown below, is dependent on the routine that made the entry in the LXB.

Byte 2
This is an input byte that the routine keys on. This is also dependent on the routine and is described below.

DFSCMC00 (MSC Analyzer)

Byte 1, bits 4-7

Value	Meaning
0	No I/O operation was queued; contention exists for the CTC adapter
1	WRITE channel program was queued
2	ACK channel program was queued
3	WRACK channel program was queued
4	READ channel program was queued; contention exists for use of the CTC adapter
5	STARTUP channel program was modified to be a WRITE channel program
6	Old STARTUP channel program was modified to be a WRITE channel program
7	WRITE channel program was not queued; write-pending switch was set
8	Error return was given

Byte 2
This contains the operation code (found in DECTYPE+1).

DFSCMC50 (Shutdown Processing Routine)

Byte 1, bits 4-7

Value	Meaning
1	Normal STACK operation was performed
2	Normal SHUTDOWN operation was performed

3 Abnormal SHUTDOWN occurred

Byte 2
This contains the operation code (found in DECTYPE+1).

DFSCMC40 (Attention DIE Routine)

Byte 1, bits 4-7
IOSB was passed to IOS to perform a read.

Value	Meaning
0	Error was previously posted
1	IOSB was passed to IOS
2	IOSB on queue was modified to perform a read
3	LLB was posted with ACK received
4	LXB was posted with STARTUP complete; the link is available for a WRITE operation
5	LXB was posted with an error
6	LLB was posted with an error
7	During STARTUP processing, a control command was received after this routine used a no-operation command
8	Attention interrupt was received during SHUTDOWN processing; UCB was already cleared
9	Attention interrupt was received during SHUTDOWN processing; this routine did not reset UCBQISCE switch
A	Attention interrupt was received during SHUTDOWN processing; this routine did not reset UCBQISCE switch
B	Attention interrupt was received during SHUTDOWN processing; this routine scheduled an IOSB
C	Attention interrupt was received during SHUTDOWN processing; this routine set LXBC2XS switch
D	LXBC2SD switch was set after an attention interrupt because a WRITE command was received; READ operation was not done
E	Read-pending or response-received switch was set
F	Attention interrupt was received during SHUTDOWN processing; SHUTDOWN channel program was aborted

Byte 2
The command byte is sensed from the channel-to-channel adapter (found at IOSCTCMD), except when an I/O error prevented retrieval of the command byte, in which case byte 2 is absent.

DFSCNC40 (I/O Request DIE Routine)

Byte 1, bits 4-7

Value	Meaning
0	Second entry into this routine was taken; nothing was done
1	LXBCLIB switch was reset
2	IOSB on queue was modified to perform a WRITE operation (this is always a 1-byte entry)

DFSCMC10 (Channel-End Appendage)

Byte 1, bits 4-7

Value	Meaning
0	Nothing was done
1	LXB was posted with STARTUP complete; the link is available for a WRITE operation
2	LXB was posted with STARTUP complete; STARTUP message was received
3	During STARTUP processing, no-operation command was scheduled
5	LXB was posted; message received
6	LLB was posted; message received
8	During STARTUP processing, control command was scheduled
9	LLB was posted; an error occurred on message that was written
A	LLB was posted; an error occurred on message that was received
B	LXB was posted; an error occurred on message that was received

Byte 2
This contains the first command code in the just-completed channel program (pointed to by IOSVST).

DFSCMC10 (Abnormal-End Appendage)

Byte 1, bits 4-7

Value	Meaning
2	Not a permanent error; control is given to an ERP
3	Error was declared permanent
4	Serial channel error
5	MIH detected error before retry

Byte 2
This contains the value in IOSCOD.

DFSCMC10 (Shutdown Appendage)

Byte 1, bits 4-7

Value	Meaning
1	Completion was normal; a new I/O operation was scheduled
2	Completion was normal; LLB was posted
3	Completion was abnormal; UCB was already cleared
4	Completion was abnormal; this routine has cleared UCB and posted LLB
5	Completion was abnormal; this routine will restart I/O
6	Completion was abnormal; this routine has restarted I/O
7	Completion was normal; UCB was already cleared

Byte 2
This contains the first command code in the just-completed channel program (pointed by IOSVST).

LXB Trace Stack Example

Figure 129 is a printout of the LXB portion of an internal trace record. The LXB trace stack begins at AE90E8, and it contains 29 entries. Following Figure 129 is a list of the meanings of the routines that made each entry.

```
DFSERA30 —— FORMATTED LOG PRINT
:
INTERNAL TRACE RECORD
:
LXB
 AE9004  000000   807F0BC9 00093660   00AE9350 00AE92B0   00091E90 00AE991C  17000000 7F0C0000
 AE9024  000020   80000000 520821CE   0008229C 000820C6   80082194 012141CE  60000054 0A000000
 AE9044  000040   30000005 022140C6   600000CE 09000000   30000005 47000000  20000001 00000000
 AE9064  000060   00000000 00000000   00000000 00000000   00000000 00000000  00000000 00000000
 AE9084  000080 TO AE90C4  0000C0   SAME AS ABOVE
 AE90E4  0000E0   00000000 0C419317   F1044193 17F10441   9337E218 D243F510  A314A8C3 419101A2
 AE9104  000100   02F30C41 93179101   A502F004 F30C4193   17F10441 93170000  00000000 00B66218
```

Figure 129. Printout of the LXB Trace Stack

Entry	Meaning
X'OC'	The first byte of this entry, the oldest entry in the trace stack, has been pushed off the trace stack. Ignore this entry.
X'41'	DFSCMC40 (I/O request DIE). LXBCLIB was reset.
X'9317'	DFSCMC40 (attention DIE). Operation code X'17' (ACK) was received from the other system. The LLB was posted X'7F1C0000' (ACK received).
X'F104'	DFSCMC00. Operation code X'04' (WRITE) was received. The WRITE channel program was queued.
X'41'	DFSCMC40. (I/O request DIE). LXBCLIB was reset. WRITE operation was completed.
X'9317'	DFSCMC40 (attention DIE). Operation code X'17' (ACK) was received from the other system. The LLB was posted X'7F1C0000' (ACK received).
X'F104'	DFSCMC00. Operation code X'04' (WRITE was received). The WRITE channel program was queued.
X'41'	DFSCMC40 (I/O request DIE). LXBCLIB was reset. WRITE operation was completed.
X'9337'	DFSCMC40 (attention DIE). Operation code X'37' (STACK) was received from the other system. The LLB was posted X'7F1C0000' (ACK received).
X'E218'	DFSCMC50 (SHUTDOWN processing). Operation code X'18' (SHUTDOWN) was received. Normal SHUTDOWN was performed.
X'D243'	DFSCMC10 (SHUTDOWN appendage). Channel command X'43' (enable compatibility) completed normally. The LLB was posted.
X'F510'	DFSCMC00. Operation code X'10' (STARTUP) was received. The start-link channel program was queued.
X'A314'	DFSCMC10 (channel-end appendage). Channel command X'14' (sense command byte) of the start-link channel program completed normally. The disable compatibility no-operation command was scheduled.
X'A8C3'	DSFCMC10 (channel-end appendage). Channel command C'X3' (disable compatibility no-operation) completed normally. The startup control command was scheduled.
X'41'	DFSCMC40 (I/O request DIE). LXBCLIB was reset. Channel end was received from the startup control.

X'9101' DFSCMC10 (attention DIE). Operation code X'01' (WRITE) was received from the other system. The IOSB was passed to IOS to initiate a READ.

X'A202' DFSCMC10 (channel-end appendage). Channel command X'02' (read) completed normally. The LXB was posted X'7F080000'(startup complete, startup message received).

X'F30C' DFSCMC00. Operation code X'0C' (WRACK) was received. ACK with data (WRACK) channel program was queued.

X'41' DFSCMC40 (I/O request DIE). LXBCLIB was reset. WRACK operation has completed.

X'9317' DFSCMC40 (attention DIE). Operation code X'17' (ACK) was received from the other system. The LLB was posted X'7F0C0000' (ACK received).

X'9101' DFSCMC40 (attention DIE). Operation code X'01' (WRITE) was received from the other system. The IOSB was passed to IOS to initiate a READ operation.

X'A502' DFSCMC10 (channel-end appendage). Channel command X'02' (read) was completed. The LXB was posted X'7F0C0000' (message received).

X'F004' DFSCMC00. Operation code X'04' (WRITE) was received. No I/O was scheduled. Contention exists between this WRITE operation and the WRITE operation received from the other system in the preceding 9101 entry. The DDM has not yet received control in response to the LXB post traced by the preceding A502 entry.

X'F30C' DFSCMC00. Operation code X'0C' (WRACK) was received. ACK with data (WRACK) channel program was queued.

The ACK acknowledges the data received from the other system in the preceding 9101 entry. The data is the data that was not sent in the preceding F004 entry.

X'41' DFSCMC40 (I/O request DIE). LXBCLIB was reset.

X'9317' DFSCMC40 (attention DIE). Operation code X'17' (ACK) was received from the other system. The LLB was posted X'7F1C0000' (ACK received).

X'F104' DFSCMC00. Operation code X'04' (WRITE) was received. The WRITE channel program was queued.

X'41' DFSCMC40 (I/O request DIE). LXBCLIB was reset. WRITE operation was completed.

X'9317' DFSCMC40 (attention DIE). Operation code X'17' (ACK) was received from the other system. The LLB was posted X'7F1C0000' (ACK received).

MSC Routine Trace—BUFMSVID

This trace records the MSVID (as specified in the IMSCTRL macro during system definition) of the last eight IMS systems through which messages were routed. It is initialized when a terminal sends a message or when an application program does an ISRT of a message, and it is updated for each intermediate system and the destination system. The MSC routing trace is located in the MSC message prefix at label BUFMSVID within the BUFMS DSECT. The low-order byte in the trace contains the most recent entry, and each additional entry results in a shift left (the high-order byte is shifted out).

In Version 6, this trace records the primary MTO’s local SYSID of the last eight IMS systems through which messages were routed. It is initialized when a terminal sends a message or when an application program does an ISRT of a message, and it is updated for each intermediate system and the destination system. The MSC routing trace is located in the MSC message prefix extension at label MSGMEVID in DSECT MSGMSCE. The low-order byte in the trace contains the most recent entry, and each additional entry results in a shift left (the high-order byte is shifted out). If the SYSID is equal to or greater than 255, it is traced both in field BUFMEVID and MSGMEVID. IF the SYSID is less than 255, it is only traced in MSGMEVID; BUFMEVID contains zeros.

Chapter 13. DBRC—Database Recovery Control Service Aids

This chapter describes diagnostic aids that help you analyze problems in DBRC. Included are:
- Diagnosing from a RECON list
- A description of the DBRC internal and external trace

Diagnosing from a RECON List

You can use the LIST command to list the contents of all or part of the RECON data set. You can list:
- The copy1 RECON data set
- RECON records for a particular change-accumulation group or for all change-accumulation groups
- RECON records for a particular log data set or for all log data sets
- RECON records for a particular database data set or for DBDS groups
- Databases
- Subsystems
- Interim log records

Because some information is not printed when you issue the `LIST.RECON` command, you can issue the access method services `PRINT` command to list all information in hexadecimal format.

Related Reading: For information about the use of the `LIST.RECON` command and RECON record types, see *IMS Version 7 Database Recovery Control (DBRC) Guide and Reference*.

RECON Record Types

The records in the RECON data set store information about logging activity and events that can affect the recovery of the database. This section describes the content of the keys in the RECON records. To view the layout of the entire RECON record, see Table 71. Consider these points as you examine the records:
- The RECON key size is 32 bytes.
- The last three bytes of the key are reserved, and contain zeros.
- Beginning with IMS version 6.1, time stamps have the following characteristics:
  – Time stamps are 12 bytes.
  – The symbolic UTC format is:
    YYYYYDDDFHHMMSSTHMIJUAQQS
    An example of the UTC format is: 199906F211432800000032D
  – DSPTIMES (DFSTIMES) contains time stamp structure information.

Table 71. Recon Record Types

Common Name	Part Name	List ID	Release	Key Fields
RECON Header	DSPRCNRC	RECON	R-1	DBD: hex zeros DDN: hex zeros Type: X'01' Time: hex zeros
RECON Header Extension	DSPRCR1	******	R-3	DBD: hex zeros DDN: hex zeros Type: X'01' Time: X'00000000008'

Table 71. Recon Record Types (continued)

Common Name	Part Name	List ID	Release	Key Fields
Time History Table	DSPTHTRC	THT	6.1	DBD: hex zeros DDN: hex zeros Type: X'01' Time: X'00000000010'
Audit Trail Record	DSPMUPHD	*****	2.1	DBD: hex zeros DDN: hex zeros Type: X'02' Time: sequence number
PRILOG	DSPLOGRC	PRILOG	R-1	DBD: hex zeros DDN: hex zeros Type: X'05' Time: timestamp
Interim PRILOG	DSPLOGRC	IPRI	R-2	DBD: hex zeros DDN: hex zeros Type: X'06' Time: timestamp
LOGALL	DSPLGARC	LOGALL	R-1	DBD: hex zeros DDN: hex zeros Type: X'07' Time: timestamp
SECLOG	DSPLOGRC	SECLOG	R-1	DBD: hex zeros DDN: hex zeros Type: X'09' Time: timestamp
Interim SECLOG	DSPLOGRC	ISEC	R-2	DBD: hex zeros DDN: hex zeros Type: X'0A' Time: timestamp
Change Accum Group	DSPCAGRC	CAGRP	R-1	DBD: hex zeros DDN: CA group name Type: X'0F' Time: hex zeros
Change Accum Execution	DSPCHGRC	CA	R-1	DBD: hex zeros DDN: CA group name Type: X'11' Time: timestamp
DBDS Group	DSPDGRC	DBDSGRP	2.1	DBD: X'0000000000000007' DDN: DBDS group name Type: X'16' Time: hex zeros
Database Header	DSPDBHRC	DB	R-2	DBD: DBD name DDN: DDN name Type: X'18' Time: hex zeros
Partition	DSPPTNRC	DB	7.1	DBD: DBD name DDN: Partition name Type: X'19' Time: hex zeros
Database Data Set	DSPDSHRC	DBDS	R-1	DBD: DBD name DDN: DDN name Type: X'20' Time: hex zeros
Area Recovery	DSPDSHRC	DBDS	R-3	DBD: DBD name DDN: area name Type: X'20' Time: hex zeros
Area Auth	DSPDBHRC	DBDS	R-3	DBD: DBD name DDN: area name Type: X'21' Time: hex zeros

Table 71. Recon Record Types (continued)

Common Name	Part Name	List ID	Release	Key Fields
ALLOC	DSPALLRC	ALLOC	R-1	DBD: DBD name DDN: DDN or area name Type: X'28' Time: timestamp
Image Copy	DSPIMGRC	IMAGE	R-1	DBD: DBD name DDN: DDN or area name Type: X'2D' Time: timestamp
Reorg	DSPRRGRC	REORG	R-2	DBD: DBD name DDN: DDN or area name Type: X'32' Time: timestamp
Recovery	DSPRCVRC	RECOV	R-1	DBD: DBD name DDN: DDN or area name Type: X'37' Time: timestamp
Backout	DSPBKORC	BACKOUT	4.1	DBD: X'FFFFFFFF00000035' DDN: subsystem name Type: X'35' Time: hex zeros
Global Service Group	DSPGSRC	GSG	5.0	DBD: X'FFFFFFFFFFFF0000' DDN: subsystem name Type: X'3A' Time: hex zeros
Tracking Subsystem	DSPSSRC	SSYS	5.0	DBD: X'FFFFFFFF0000003E' DDN: subsystem name Type: X'3E' Time: hex zeros
Subsystem	DSPSSRC	SSYS	R-2	DBD: X'FFFFFFFFFFFFFFFF' DDN: subsystem name Type: X'3F' Time: hex zeros
PRISLDS	DSPSLDRC	PRISLD	R-3	DBD: X'FFFFFFFF00000043' DDN: subsystem name Type: X'43' Time: timestamp
PRITSLDS	DSPSLDRC	PRITSLDS	5.0	DBD: X'FFFFFFFF00000044' DDN: subsystem name Type: X'44' Time: timestamp
Interim PRISLDS	DSPSLDRC	IPRISL	R-3	DBD: X'FFFFFFFF00000045' DDN: subsystem name Type: X'45' Time: timestamp
Interim PRITSLDS	DSPSLDRC	IPRITSLD	5.0	DBD: X'FFFFFFFF00000046' DDN: subsystem name Type: X'46' Time: timestamp
SECSLDS	DSPSLDRC	SECSLD	R-3	DBD: X'FFFFFFFF00000047' DDN: subsystem name Type: X'47' Time: timestamp
SECTSLDS	DSPSLDRC	SECTSLDS	5.0	DBD: X'FFFFFFFF00000048' DDN: subsystem name Type: X'48' Time: timestamp

Table 71. Recon Record Types (continued)

Common Name	Part Name	List ID	Release	Key Fields
Interim SECSLDS	DSPSLDRC	ISECSL	R-3	DBD: X'FFFFFFFF00000049' DDN: subsystem name Type: X'49' Time: timestamp
Interim SECTSLDS	DSPSLDRC	ISECTSLD	5.0	DBD: X'FFFFFFFF00000050' DDN: subsystem name Type: X'50' Time: timestamp
Available CA Execution	DSPCHGRC	CA	R-1	DBD: hex zeros DDN: hex zeros Type: X'51' Time: timestamp
PRIOLDS	DSPOLDRC	PRIOLD	R-3	DBD: X'FFFFFFFF00000053' DDN: subsystem name Type: X'53' Time: timestamp
Interim PRIOLDS	DSPOLDRC	IPRIOL	R-3	DBD: X'FFFFFFFF00000055' DDN: subsystem name Type: X'55' Time: timestamp
SECOLDS	DSPOLDRC	SECOLD	R-3	DBD: X'FFFFFFFF00000057' DDN: subsystem name Type: X'57' Time: timestamp
Interim SECOLDS	DSPOLDRC	ISECOL	R-3	DBD: X'FFFFFFFF00000059' DDN: subsystem name Type: X'59' Time: timestamp
Available Image Copy	DSPIMGRC	IMAGE	R-1	DBD: DBD name DDN: DDN or area name Type: X'6D' Time: timestamp

DBRC Internal Trace

The DBRC internal trace is a useful diagnostic tool when problems are suspected in DBRC. It is always enabled.

The DBRC trace can help diagnose many different types of problems, such as:

- RECON data set contention
- RECON errors that are indicated by messages
- System abends in which the PSW is pointing to DBRC
- DBRC abends
- Whether DBRC or some other IMS component is causing the problem

Sometimes a problem occurs as a result of the interaction between two different modules performing different tasks. Interpreting trace entries is the best way to determine what each module was doing and when. For example, for RECON data set errors, it's important to know which DBRC modules manipulated the RECON and when.

You generally look at the DBRC trace output under the direction of an IBM support representative, who will guide you in collecting data in specific trace fields and in interpreting that data. The DBRC trace entries that follow help you interpret trace data.

Example: A user receives abend code xxx. The PSW is pointing to DBRC. The user reports the problem to an IBM support representative. Some of the steps that the user diagnostician might take under the guidance of the IBM representative are:

1. Locate the DBRC trace in the dump using the TRACETBL eye catcher.
2. Use the sample trace (see “DBRC Unformatted Internal Trace Example” on page 380) to verify that you have found the trace and to help you navigate through the trace table entries.
3. Find DBRC and IMS control blocks and data areas by using addresses from selected trace table entries.
4. Determine the events that occurred before the abend.
5. Use the information in the trace and data areas to understand what caused the abend.

Some DBRC functions have the capability of generating additional trace entries that can aid in problem analysis. An IBM representative may assist you in enabling one or more of these expanded trace options through the use of the `CHANGE.RECON` command.

The `CHANGE.RECON` command supports a TRACEOPT parameter that allows you, under the direction of an IBM representative, to select expanded DBRC trace options.

CHANGE.RECON

```
►►──TRACEOPT──┬─────────────┬──────────────────────────────────────────────►◄
              └─(n(,m...))──┘
```

n,m,... DBRC TRACEOPT options

TRACEOPT is an optional parameter that you use only under the direction of an IBM representative for the purpose of gathering documentation for problem analysis. The IBM representative will provide the sub-options for the TRACEOPT parameter.

Trace Input

When called, DSPTRACE receives a 16-byte parameter list that consists of:

- An 8-character identifier that becomes the first 8 characters of the trace entry
- A 4-byte control block pointer that points to a DFSBRLSB or the DSPGDB
- A 4-byte block area pointer. 64 bytes of data from the block area are inserted in the trace entry. If the pointer is 0, the trace entry is 32 bytes long; otherwise it is 96 bytes long.

Locating the Trace

The DBRC trace is in the IMS-formatted portion of an IMS-formatted dump. You can locate the DBRC trace in these ways:

Method 1

Find the trace in the DBRC section of the IMS offline formatted dump.

Method 2

Find any DSPxxxxx module in the Save Area trace of the dump. For most DSPxxxxx modules marked ENTERED VIA CALL, register 5 contains the address of the Global Data Block (GDB). Offset X'38' in the GDB contains the address of router storage. Offset X'1C' in router storage contains the address of the DBRC trace.

In certain situations, register 5 does not point to the GDB. If this is the case, use method 3 or 4.

Method 3

The trace is in subpool 0. If the dump has an index, look in the index to locate subpool 0. Scan this portion of the dump for eye-catcher “TRACETBL”, which identifies the beginning of the trace.

Method 4

If you are looking at a dump online, search for either eye-catcher “TRACETBL” or “GETFEED”. If you search for “GETFEED”, you might first find it within DBRC modules. Keep searching until you find “GETFEED” within the DBRC trace. Scroll back to the beginning of the trace. To verify that you are looking at the trace, see the trace example in “DBRC Unformatted Internal Trace Example” on page 380.

Trace Output

Trace output normally resides in subpool 0 storage, but you can direct output to a Generalized Trace Facility (GTF) data set. To do this, see “DBRC External Trace” on page 384.

The DBRC internal trace is a wrap-around trace. That is, after the trace table is full, tracing starts at the beginning of the table, and each new entry overlays an old entry.

An entry with the identifier ***TRACENXT*** marks the next entry to be used, which is the logical end of the trace table.

The format of the header record and key trace entries are shown on the following pages.

Trace Header Record

Figure 130 shows the DBRC trace header record.

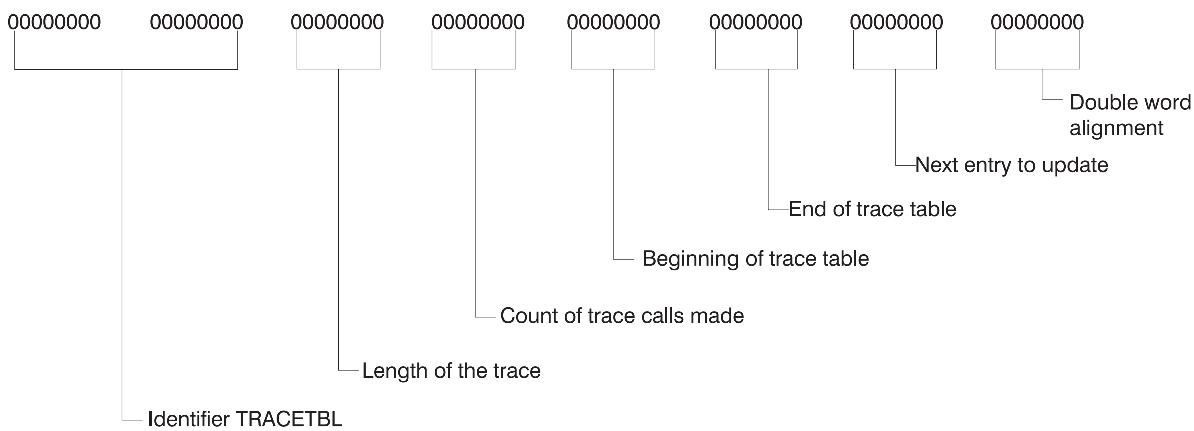


Figure 130. DBRC Trace Header Record

Module Call, Module Return, and DSPSTACK Trace Entries

A summary of the DBRC processing that produces the trace entries precedes the layout of the trace entries.

With few exceptions, DBRC modules call module DSPSTGET to obtain initial work space and additional temporary work space (with the DSPGFSTK macro). Upon exit, DSPSTFRE releases the space obtained for the module. This centralized temporary storage management allows DBRC to track the flow of modules, starting with the first call out of DSPCRTRO (entry point to DBRC). Three trace entries accomplish this:

- Words 1 and 2, which in previous releases only contained DSPSTGET or DSPSTFRE, now show the following things:
  - An arrow indicating whether the module is being called or is returning.
  - The nesting level of the module being called or returned to. Nesting levels are shown in one or two decimal digits up to 99. (Nesting level 0 is DSPUIN00)
  - The last five characters of the module name being called or returning.

- DSPSTACK–additional work space trace entry (the result of the currently active module issuing the DSPGFSTK macro that calls DSPSTGET)

Figure 131 illustrates the following processing flow:

1. Module A calls module B, which in turn calls DSPSTGET to obtain initial work space.
2. Module B issues macro DSPGFSTK to obtain additional work space.
3. Module B calls DSPSTFRE to release all temporary storage.
4. Module B returns control to module A.

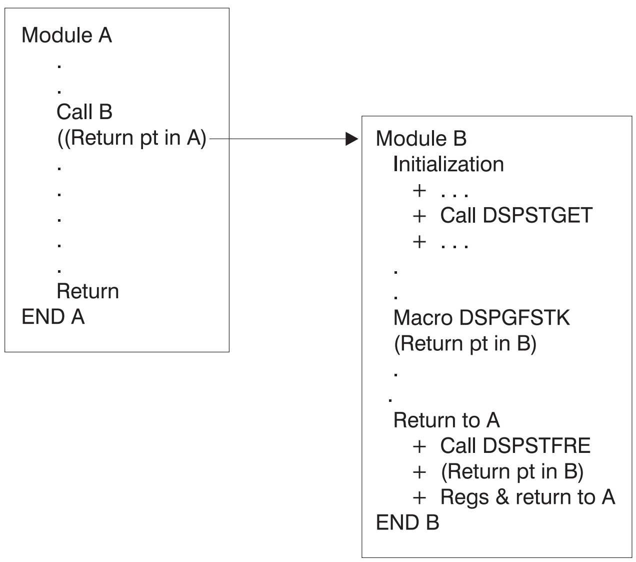


Figure 131. DBRC Trace Processing Flow

Figure 132 on page 372, Figure 133 on page 372, and Figure 134 on page 373 illustrate the format of the trace entries associated with this module flow. Each entry occupies one line (8 words) in the DBRC internal trace table. References to specific addresses and locations in modules A and B refer to the diagram in Figure 131.

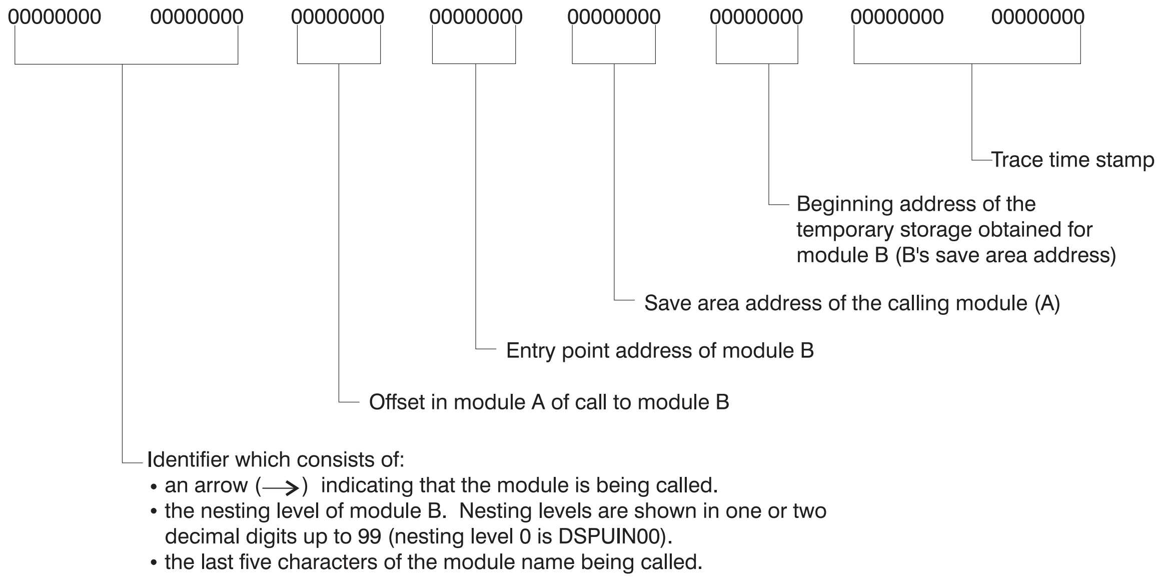


Figure 132. A one-line trace entry that is produced when module A calls module B. A one-line trace entry that is produced when module B calls DSPSTGET to obtain initial work space storage after being called by module A.

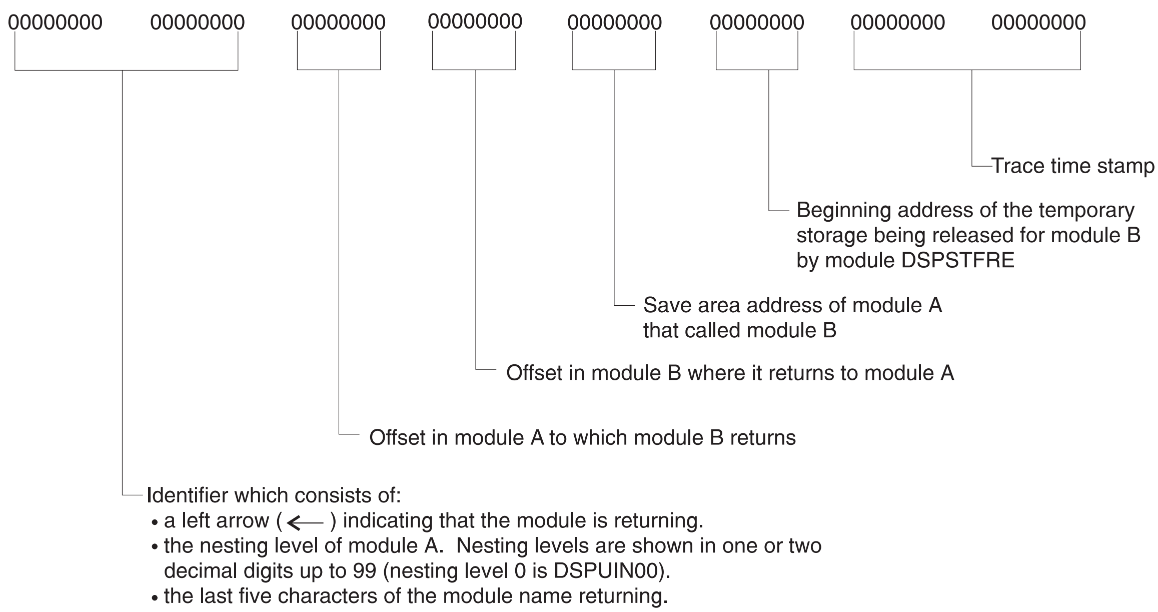


Figure 133. A one-line trace entry that is produced when module B returns to module A. A one-line trace entry that is produced when module B calls DSPSTFRE to release all of its temporary storage before returning to module A.

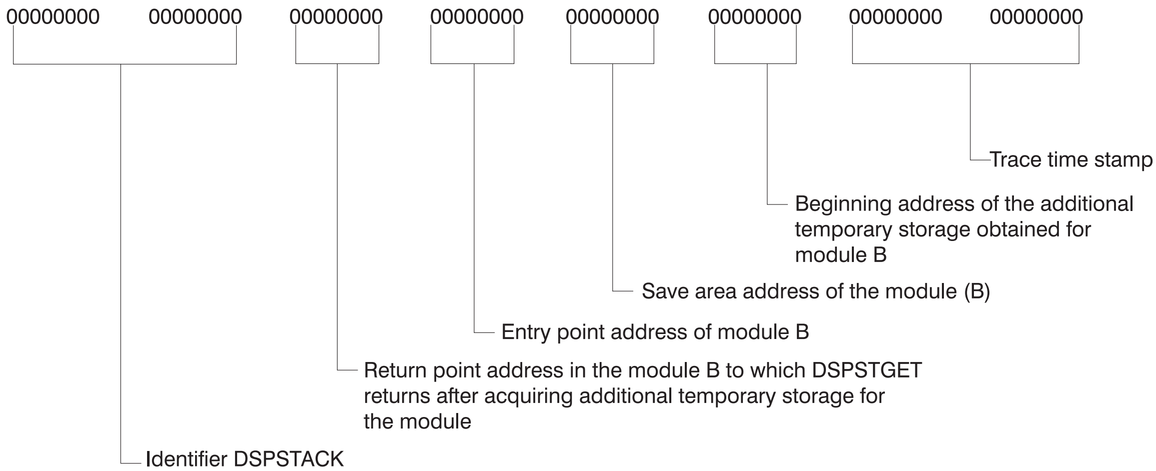


Figure 134. DSPSTACK Trace Entry. A one-line trace entry that is produced when module B issues macro DSPGFSTK, which calls DSPSTGET to obtain additional temporary storage.

BGNCABN0, DSPCABN0, BGNRETRY, DSPCRTR0, and CRTR0XIT Trace Entries

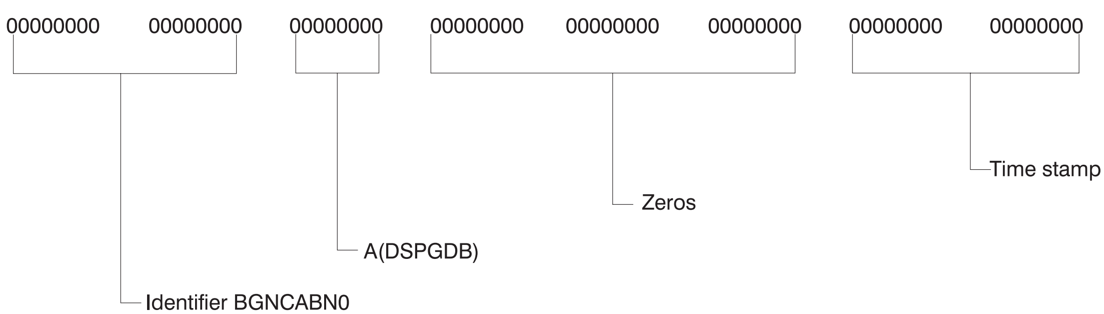
In DBRC, these modules have specific trace calls inserted in their processing flow:

DSPCABN0

DSPCRTR0

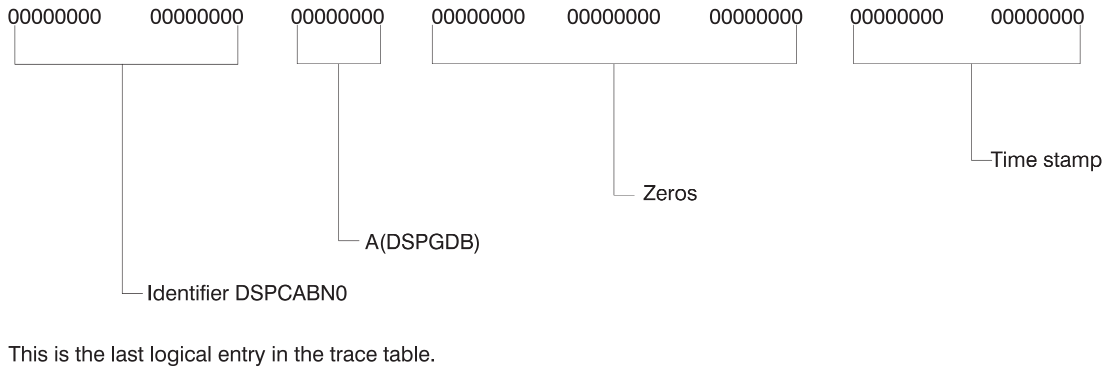
DSPURI00

Figure 136 on page 374, Figure 137 on page 374, Figure 138 on page 375, and Figure 139 on page 376 show the layout of the entries issued from BGNCABN0, DSPCABN0, and DSPCRTR0.



This is normally followed by either DSPCABN0 or a BGNRETRY entry.

Figure 135. BGNCABN0 Trace Entry



This is the last logical entry in the trace table.
Figure 136. DSPCABN0 Trace Entry. DBRC terminated because of an unrecoverable error.

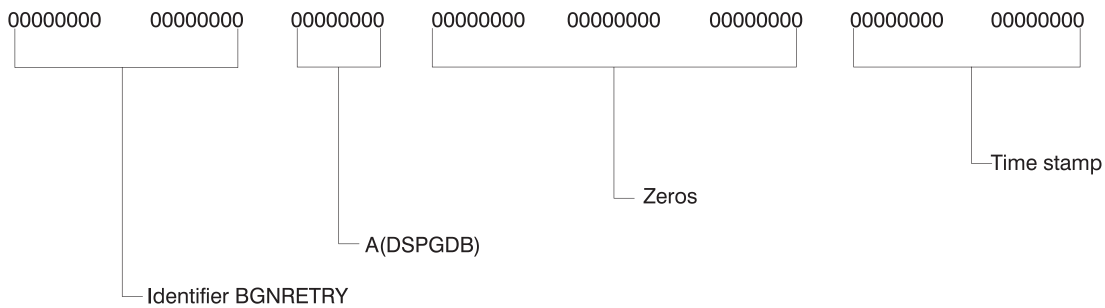


Figure 137. BGNRETRY Trace Entry. DBRC recovered from an abend condition and is beginning to execute a retry sequence of code.

Figure 138. DSPCRTR0 Trace Entry. The router made a trace call before passing control to the next DBRC routine scheduled to process the request identified by a DFSBRLSB.

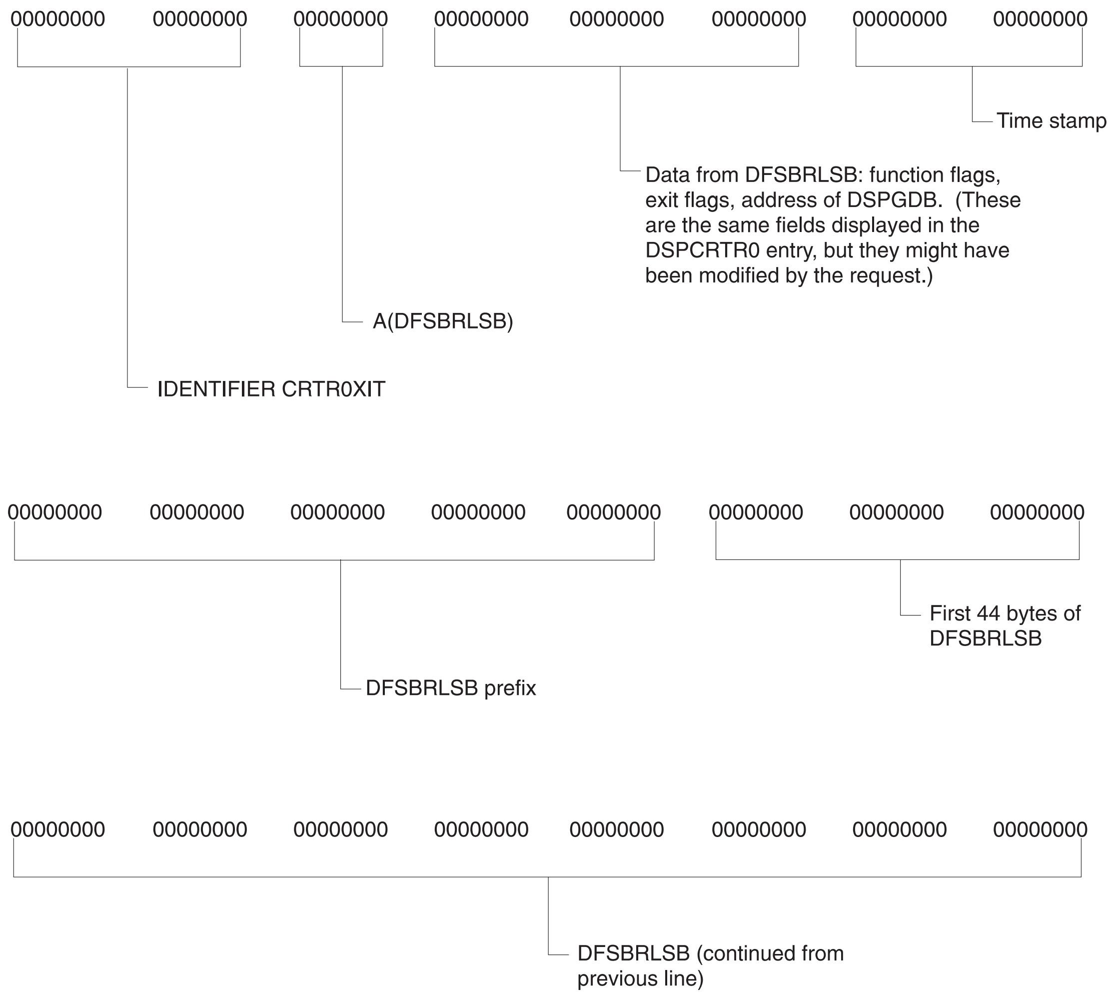


Figure 139. CRTR0XIT Trace Entry. The function requested in the DSPCRTR0 trace entry completed.

DSPURI00 Trace Entries

A trace entry with the identifier DSPURI00 indicates the beginning of a series of trace calls that show what occurs as DSPURI00 processes an I/O request. All trace calls from DSPURI00 result in 96-byte trace entries. There are nine separate calls to the trace routine in DSPURI00. The pointer to the DSPGDB follows the trace identifier. Table 71 shows the 8-character identifier and block-area pointer for each call.

Table 72. Calls to the Trace Routine in DSPURI00

8-Character Identifier	Block-Area Pointer	Explanation
DSPURI00	MODIRCAR	DSPURI00 receives control and the function-code value from DSPIRCAR indicates the type of call. (See Figure 140 on page 378.)
OPENER1	FILRESLT(I)	DSPURI00 starts a true open of the RECON data set.
OPENER2	FILRESLT(I)	DSPURI00 completes a true open of the RECON data set.
GETFEED	FILRESLT(I)	After DSPURI00 issues an I/O request, the GETFEED procedure is called to issue a SHOWCB. This trace entry shows the results that VSAM returns from the SHOWCB request and maps 64 bytes of the DSPVFILE data area starting with the FILRESLT field. (See Figure 141 on page 379.)
CLOSER1	FILRESLT(I)	DSPURI00 starts a true close of the RECON data set.

Table 72. Calls to the Trace Routine in DSPURI00 (continued)

8-Character Identifier	Block-Area Pointer	Explanation
CLOSER2	FILRESLT(I)	DSPURI00 completes a true close of the RECON data set.
VSAMERR	FILRESLT(I)	A VSAM error occurred and the routine to print a VSAM error message was entered.
DSPURI00	ENDIRCAR	DSPURI00 returns to its caller. Relevant exit condition information, if applicable, is traced. (See Figure 142 on page 380.)

Note: The sequence of trace entries identified by DSPURI00, OPENER1, OPENER2, and GETFEED shows DSPURI00 receiving control and doing a true open of one RECON data set. When DSPURI00 opens the second RECON data set, another sequence of OPENER1, OPENER2, and GETFEED entries follow the entries for the first RECON data set.

Figure 140 on page 378, Figure 141 on page 379, and Figure 142 on page 380 show the layout of three of the trace entries from DSPURI00.

The DSPIRCAR data area includes a 1-byte function code and a 3-byte flag field. The function codes are alphabetic characters that identify what operation DSPURI00 does. The flag bytes further identify the type of operation. Pertinent information is extracted from the DSPIRCAR data area and placed in a modified IRCAR area, along with other processing information, to produce both the entry and exit traces within DSPURI00.

The GETFEED trace entry maps 64 bytes of data from DBRC's DSPVFILE data area beginning with the FILRESLT field. (The last two lines of the entry contain this data.)

The exit trace entry is similar to the entry trace. It is written upon return from DSPURI00, but only if one or more of the following conditions is true:

- This was a request to locate a specific RECON record.
- The request did not complete successfully (RC greater than 0 was returned).
- The copy 1 or 2 RECON status changed on this entry to DSPURI00.

time stamp
Trace time stamp

c1c2
The DD statement number (1-3) of the copy 1 and copy 2 RECON, if any, on entry to DSPURI00

func
Function and option bits received from caller in DSPIRCAR

16-byte entry message
EBCDIC message readable at the right end of the trace entry, such as LOGICAL OPEN, END MULT, UPDATE, and others. Class and sequential locate requests and configuration requests have a “modifier” at the end of their message:

F Locate first

L Locate last

NX Locate next

P Locate previous

NG Locate not-greater-than

DSNS Supply dsnames of RECONs in DSPIRCAR

STAT Supply status of all RECONs in DSPIRCAR

DUAL Enter dual mode

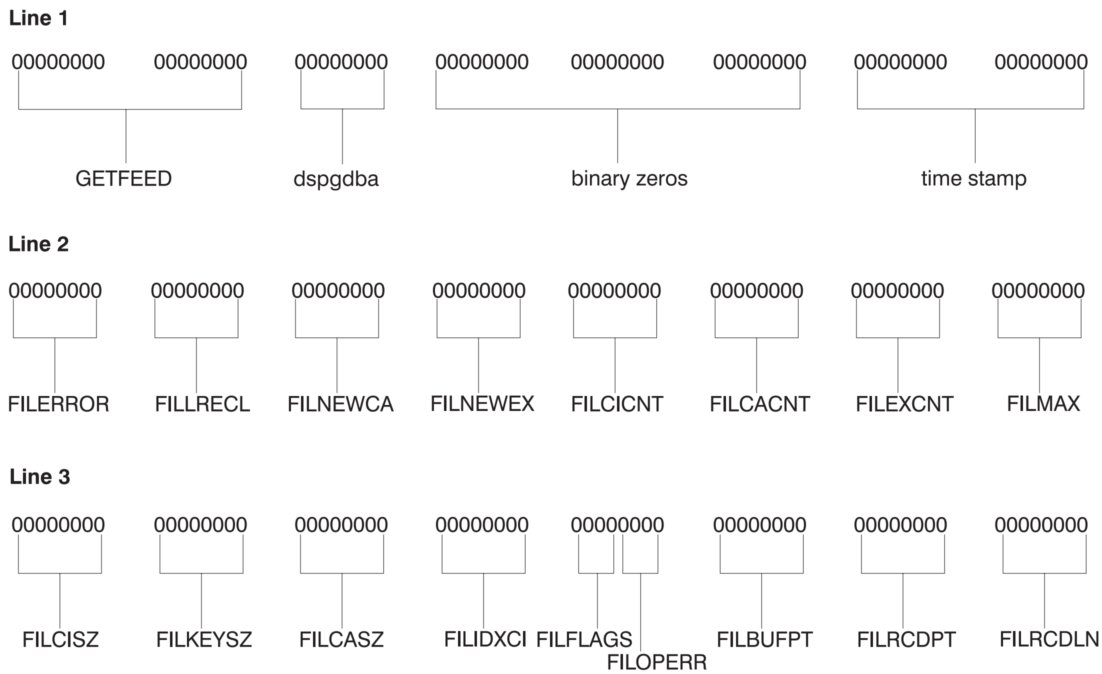
REPL Replace RECONx with spare (where x = 1-3, see key area)

key area
For all locate, change, insert, and delete requests, contains the 32-byte key of the record involved. For replace requests, contains the ddname of the RECON to be replaced

addr
Address of a record to be changed or inserted

leng
Length of a record to be changed or inserted

Figure 140. DSPURI00 Entry Trace Entry



dspgdba Address of the DSPGDB

time stamp Trace time stamp

FILERROR I/O feedback from the SHOWCB macro

FILLRECL Logical record length

FILNEWCA Starting high-used relative byte address (RBA)

FILNEWEX Starting high-allocated RBA

FILCICNT RECON changed counter value

FILCACNT Current high-used RBA

FILEXCNT Current high-allocated RBA

FILMAX VSAM maximum record size

FILCISZ Data control interval (CI) size

FILKEYSZ RECON key size

FILCASZ Number of data control intervals per CA

FILIDXCI Index control interval (CI) size

FILFLAGS RECON processing status flags (open, reserved, empty)

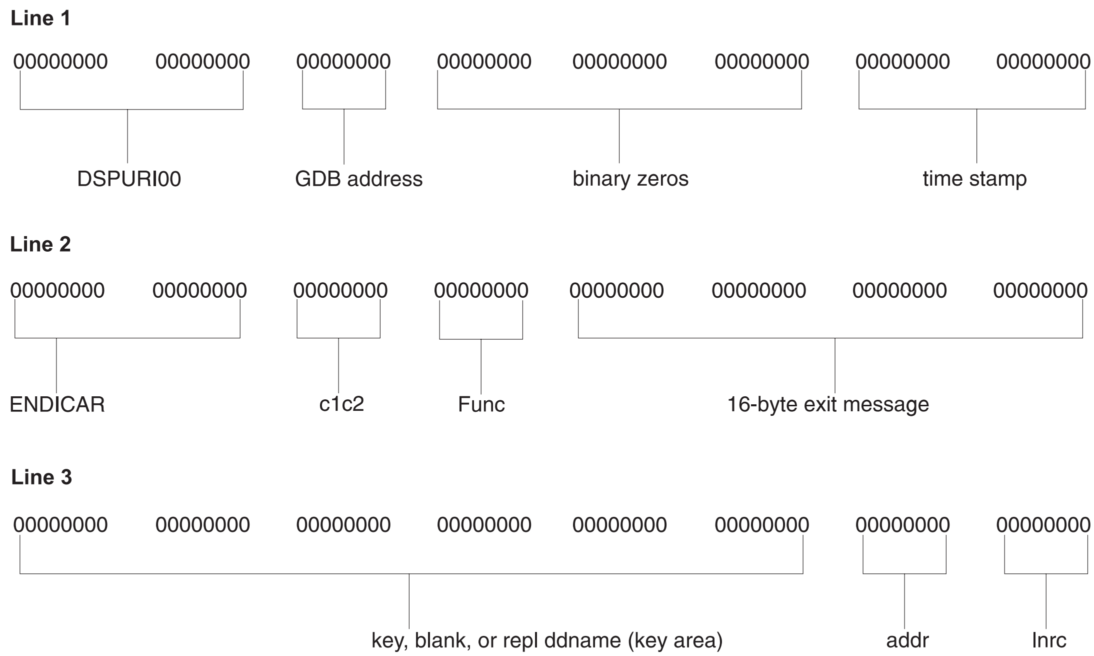
FILOPERR Open SVC reason code if RC is not 0

FILBUFPT Pointer to header record buffer

FILRCDPT Pointer to the record in the VSAM I/O buffer or user area

FILRCDLN Length of the record

Figure 141. GETFEED Trace Entry for One RECON



time stamp Trace time stamp

c1c2 The DD statement number (1-3) of the copy 1 and copy 2 RECON, if any, on exit from DSPURI00

func Function and option bits received from caller in DSPIRCAR

16-byte exit message
For locate requests, contains either the message RECORD WAS FOUND or RECORD NOT FOUND, depending on the outcome of the search. Otherwise, contains a repeat of MODIRCAR contents

key area For a successful locate request, contains the 32-byte key of the RECON record returned to caller. Otherwise, contains a repeat of MODIRCAR contents

addr Address of the record found for a successful locate. Otherwise, 0

lnrc Length of the record found for a successful locate, or the return code to be passed back to the module that called DSPURI00

Figure 142. DSPURI00 Exit Trace Entry

DBRC Unformatted Internal Trace Example

The following example shows module-call and module-return entries and DSPURI00 trace entries.

```
07B60300 E3D9C1C3 C5E3C2D3 00012D00 000003F7    07B60320 07B72F20 07B6E440 07B72FA0   *TRACETBL.......7..........U ....*
07B60320 606EF1E3 C9D4C5F0 00011674 07B4B298    00005548 00014010 98324F19 42499845   *->1TIME0.......q...... .q.³...q.*
07B60340 F04C60E3 C9D4C5F0 00011674 00000FFE    00005548 00014010 98324F19 42499846   *0<-TIME0.............. .q.³...q.*
07B60360 606EF1E4 D9C9F0F0 000117D4 07B00DD0    00005548 00014010 98324F19 42499848   *->1URI00...M...}...... .q.³...q.*
07B60380 C4E2D7E4 D9C9F0F0 00012E08 00000000    00000000 00000000 98324F19 42499848   *DSPURI00................q.³...q.*
07B603A0 D4D6C4C9 D9C3C1D9 40404040 D6000600    D7C8E8E2 C9C3C1D3 40D6D7C5 D5404040   *MODIRCAR    O...PHYSICAL OPEN   *
07B603C0 40404040 40404040 40404040 40404040    40404040 40404040 00000000 00000000   *                        ........*
07B603E0 606EF2E4 C3D7F4F0 000009D6 0000F400    00014010 00014890 98324F19 42499848   *->2UCP40...O..4... .....q.³...q.*
07B60400 F14C60E4 C3D7F4F0 000009D6 00000698    00014010 00014890 98324F19 42499849   *1<-UCP40...O...q.. .....q.³...q.*
07B60420 606EF2E4 D9C9F1F0 00000F22 07B05768    00014010 00014890 98324F19 42499850   *->2URI10.......... .....q.³...q&*
07B60440 606EF3E4 D9C9F2F0 00000100 07B07178    00014890 00014C28 98324F19 42499850   *->3URI20..............<.q.³...q&*
07B60460 F24C60E4 D9C9F2F0 00000100 00000134    00014890 00014C28 98324F19 42499852   *2<-URI20..............<.q.³...q.*
07B60480 606EF3E4 C5E7F0F0 00000220 07B4AA8C    00014890 00014C28 98324F19 42499852   *->3UEX00..............<.q.³...q.*
07B604A0 F24C60E4 C5E7F0F0 00000220 000004EE    00014890 00014C28 98324F19 42499855   *2<-UEX00..............<.q.³...q.*
07B604C0 606EF3E4 C5E7F0F0 000002C2 07B4AA92    00014890 00014C28 98324F19 42499855   *->3UEX00...B...k......<.q.³...q.*
07B604E0 F24C60E4 C5E7F0F0 000002C2 00000518    00014890 00014C28 98324F19 42499856   *2<-UEX00...B..........<.q.³...q.*
07B60500 606EF3E4 C5E7F0F0 00000220 07B4AA8C    00014890 00014C28 98324F19 42499856   *->3UEX00..............<.q.³...q.*
07B60520 F24C60E4 C5E7F0F0 00000220 000004EE    00014890 00014C28 98324F19 42499858   *2<-UEX00..............<.q.³...q.*
07B60540 606EF3E4 C5E7F0F0 000002C2 07B4AA92    00014890 00014C28 98324F19 42499858   *->3UEX00...B...k......<.q.³...q.*
07B60560 F24C60E4 C5E7F0F0 000002C2 00000518    00014890 00014C28 98324F19 42499859   *2<-UEX00...B..........<.q.³...q.*
07B60580 606EF3E4 C5E7F0F0 00000220 07B4AA8C    00014890 00014C28 98324F19 42499859   *->3UEX00..............<.q.³...q.*
07B605A0 F24C60E4 C5E7F0F0 00000220 000004EE    00014890 00014C28 98324F19 42499861   *2<-UEX00..............<.q.³...q/*
07B605C0 606EF3E4 C5E7F0F0 000002C2 07B4AA92    00014890 00014C28 98324F19 42499861   *->3UEX00...B...k......<.q.³...q/*
07B605E0 F24C60E4 C5E7F0F0 000002C2 00000518    00014890 00014C28 98324F19 42499861   *2<-UEX00...B..........<.q.³...q/*
07B60600 606EF3D9 E2E5F0F0 0000034A 0000E1E0    00014890 00014C28 98324F19 42499862   *->3RSV00......\......<.q.³...q.*
07B60620 F24C60D9 E2E5F0F0 0000034A 00000226    00014890 00014C28 98324F19 42502677   *2<-RSV00.............<.q.³..&..*
07B60640 606EF3E4 D9C9F1F0 00000382 07B0578C    00014890 00014C28 98324F19 42502678   *->3URI10...b..........<.q.³..&..*
07B60660 D6D7C5D5 C5D9F140 00012E08 00000000    00000000 00000000 98324F19 42502678   *OPENER1 ................q.³..&..*
07B60680 00000000 00000000 00000000 00000000    00000000 00000000 00000000 00000000   *................................*
      LINE 07B606A0  SAME AS ABOVE
07B606C0 606EF4E4 D9C9F1F0 00000A92 07B05792    00014C28 00014FC0 98324F19 42531836   *->4URI10...k...k..<...³{q.³.....*
07B606E0 F34C60E4 D9C9F1F0 00000A92 00000CA6    00014C28 00014FC0 98324F19 42531850   *3<-URI10...k...w..<...³{q.³....&*
07B60700 D6D7C5D5 C5D9F240 00012E08 00000000    00000000 00000000 98324F19 42531850   *OPENER2 ................q.³....&*
07B60720 00000000 00000000 00000000 00000000    00000000 00028800 00028800 0000314A   *......................h...h....*
JOB DBRLATAM         STEP DBRLATAM        TIME 114417   DATE 98324                                            PAGE 00000326

07B60740 00004800 00000020 00000000 00000000    00000000 00000000 00000000 00000009   *................................*
07B60760 F24C60E4 D9C9F1F0 00000382 00000B3C    00014890 00014C28 98324F19 42531850   *2<-URI10...b..........<.q.³....&*
07B60780 606EF3E4 D9C9F1F0 00000382 07B0578C    00014890 00014C28 98324F19 42531850   *->3URI10...b..........<.q.³....&*
07B607A0 D6D7C5D5 C5D9F140 00012E08 00000000    00000000 00000000 98324F19 42531850   *OPENER1 ................q.³....&*
07B607C0 00000000 00000000 00000000 00000000    00000000 00000000 00000000 00000000   *................................*
      LINE 07B607E0  SAME AS ABOVE
07B60800 606EF4E4 D9C9F1F0 00000A92 07B05792    00014C28 00014FC0 98324F19 42561630   *->4URI10...k...k..<...³{q.³.....*
07B60820 F34C60E4 D9C9F1F0 00000A92 00000CA6    00014C28 00014FC0 98324F19 42561645   *3<-URI10...k...w..<...³{q.³.....*
07B60840 D6D7C5D5 C5D9F240 00012E08 00000000    00000000 00000000 98324F19 42561645   *OPENER2 ................q.³.....*
07B60860 00000000 00000000 00000000 00000000    00000000 00028800 00028800 0000314A   *......................h...h....*
07B60880 00004800 00000020 00000000 00000000    00000000 00000000 00000000 00000009   *................................*
07B608A0 F24C60E4 D9C9F1F0 00000382 00000B3C    00014890 00014C28 98324F19 42561645   *2<-URI10...b..........<.q.³.....*
07B608C0 606EF3E4 D9C9F1F0 00000382 07B0578C    00014890 00014C28 98324F19 42561645   *->3URI10...b..........<.q.³.....*
07B608E0 D6D7C5D5 C5D9F140 00012E08 00000000    00000000 00000000 98324F19 42561645   *OPENER1 ................q.³.....*
07B60900 00000000 00000000 00000000 00000000    00000000 00000000 00000000 00000000   *................................*
      LINE 07B60920  SAME AS ABOVE
07B60940 D6D7C5D5 C5D9F240 00012E08 00000000    00000000 00000000 98324F19 42584695   *OPENER2 ................q.³....n*
07B60960 00000000 00000000 00000000 00000000    00000000 00000000 00028800 0000314A   *..........................h....*
07B60980 00004800 00000020 00000000 00000000    00000000 00000000 00000000 00000009   *................................*
07B609A0 F24C60E4 D9C9F1F0 00000382 00000B3C    00014890 00014C28 98324F19 42584695   *2<-URI10...b..........<.q.³....n*
07B609C0 606EF3E4 D9C9F2F0 0000053A 07B0719C    00014890 00014C28 98324F19 42585009   *->3URI20..............<.q.³...&.*
07B609E0 F24C60E4 D9C9F2F0 0000053A 0000045A    00014890 00014C28 98324F19 43018240   *2<-URI20.......!......<.q.³...b *
07B60A00 606EF3C4 C5D8F0F0 0000055A 07B0CC10    00014890 00014C28 98324F19 43018240   *->3DEQ00...!..........<.q.³...b *
07B60A20 F24C60C4 C5D8F0F0 0000055A 000006F6    00014890 00014C28 98324F19 43018810   *2<-DEQ00...!...6......<.q.³...h.*
07B60A40 606EF3E4 D9C9F1F0 00001652 07B0578C    00014890 00014C28 98324F19 43053109   *->3URI10..............<.q.³.....*
07B60A60 D6D7C5D5 C5D9F140 00012E08 00000000    00000000 00000000 98324F19 43053109   *OPENER1 ................q.³.....*
```

Figure 143. Example of internal trace table entries (Part 1 of 4)

```
07B60A80 00000000 00000000 00028800 00028800   00000010 00028800 00028800 0000314A   *..........h...h.......h...h....*
07B60AA0 00004800 00000020 00000000 00000000   00000000 00000000 00000000 00000009   *................................*
07B60AC0 606EF4E4 D9C9F1F0 00000A92 07B05792   00014C28 00014FC0 98324F19 43074833   *->4URI10...k...k..<...³{q.³.....*
07B60AE0 F34C60E4 D9C9F1F0 00000A92 00000CA6   00014C28 00014FC0 98324F19 43074849   *3<-URI10...k...w..<...³{q.³.....*
07B60B00 D6D7C5D5 C5D9F240 00012E08 00000000   00000000 00000000 98324F19 43074850   *OPENER2 ................q.³....&*
07B60B20 00000000 00000000 00028800 00028800   00000010 00028800 00028800 0000314A   *..........h...h.......h...h....*
07B60B40 00004800 00000020 00000000 00000000   00000000 00000000 00000000 00000009   *................................*
07B60B60 F24C60E4 D9C9F1F0 00001652 00000B3C   00014890 00014C28 98324F19 43074850   *2<-URI10.............<.q.³....&*
07B60B80 606EF3E4 D9C9F1F0 00001652 07B0578C   00014890 00014C28 98324F19 43104632   *->3URI10.............<.q.³.....*
07B60BA0 D6D7C5D5 C5D9F140 00012E08 00000000   00000000 00000000 98324F19 43104632   *OPENER1 ................q.³.....*
07B60BC0 00000000 00000000 00028800 00028800   00000010 00028800 00028800 0000314A   *..........h...h.......h...h....*
07B60BE0 00004800 00000020 00000000 00000000   00000000 00000000 00000000 00000009   *................................*
07B60C00 606EF4E4 D9C9F1F0 00000A92 07B05792   00014C28 00014FC0 98324F19 43131955   *->4URI10...k...k..<...³{q.³.....*
07B60C20 F34C60E4 D9C9F1F0 00000A92 00000CA6   00014C28 00014FC0 98324F19 43131970   *3<-URI10...k...w..<...³{q.³.....*
07B60C40 D6D7C5D5 C5D9F240 00012E08 00000000   00000000 00000000 98324F19 43131970   *OPENER2 ................q.³.....*
07B60C60 00000000 00000000 00028800 00028800   00000010 00028800 00028800 0000314A   *..........h...h.......h...h....*
07B60C80 00004800 00000020 00000000 00000000   00000000 00000000 00000000 00000009   *................................*
07B60CA0 F24C60E4 D9C9F1F0 00001652 00000B3C   00014890 00014C28 98324F19 43131970   *2<-URI10.............<.q.³.....*
07B60CC0 606EF3E4 D9C9F3F0 0000061A 07B0814C   00014890 00014C28 98324F19 43131971   *->3URI30......a<......<.q.³.....*
07B60CE0 F24C60E4 D9C9F3F0 0000061A 00000BB0   00014890 00014C28 98324F19 43132809   *2<-URI30.............<.q.³.....*
07B60D00 F14C60E4 D9C9F1F0 00000F22 00000792   00014010 00014890 98324F19 43132809   *1<-URI10.......k.. .....q.³.....*
07B60D20 C7C5E3C6 C5C5C440 00012E08 00000000   00000000 00000000 98324F19 43132810   *GETFEED ................q.³.....*
07B60D40 00000000 00000250 00028800 00028800   00000010 00028800 00028800 0000314A   *.......&..h...h.......h...h....*
07B60D60 00004800 00000020 00000000 00000000   00000000 00000000 00000000 00000009   *................................*
07B60D80 C7C5E3C6 C5C5C440 00012E08 00000000   00000000 00000000 98324F19 43132811   *GETFEED ................q.³.....*
07B60DA0 00000000 0000004E 00028800 00028800   00000010 00028800 00028800 0000314A   *.......+..h...h.......h...h....*
07B60DC0 00004800 00000020 00000000 00000000   00000000 00000000 00000000 00000009   *................................*
07B60DE0 606EF2C4 C5D8F0F0 000037C8 07B0CC10   00014010 00014890 98324F19 43132811   *->2DEQ00...H...... .....q.³.....*
07B60E00 F14C60C4 C5D8F0F0 000037C8 000006F6   00014010 00014890 98324F19 43133676   *1<-DEQ00...H...6.. .....q.³.....*
07B60E20 C4E2D7E4 D9C9F0F0 00012E08 00000000   00000000 00000000 98324F19 43133677   *DSPURI00................q.³.....*
JOB DBRLATAM         STEP DBRLATAM        TIME 114417   DATE 98324                                  PAGE 00000327

07B60E40 C5D5C4C9 D9C3C1D9 F1F24040 D6000600   D7C8E8E2 C9C3C1D3 40D6D7C5 D5404040   *ENDIRCAR12  O...PHYSICAL OPEN   *
07B60E60 40404040 40404040 40404040 40404040   40404040 40404040 00000000 00000000   *                        ........*
07B60E80 F04C60E4 D9C9F0F0 000117D4 0000079E   00005548 00014010 98324F19 43133677   *0<-URI00...M.......... .q.³.....*
07B60EA0 C4E2D7C3 D9E3D9F0 071CEC94 17172002   00000000 00012E08 98324F19 43185749   *DSPCRTR0...m............q.³.....*
07B60EC0 00000000 00000000 00CB7B38 00000001   00012E08 00000000 00000000 00000000   *..........#.....................*
07B60EE0 00000000 00000000 00000000 00000000   00000000 00000000 00000000 00000000   *................................*
07B60F00 606EF1E2 E2C9C7D5 00007AEC 07B10BD8   00005800 00014010 98324F19 43185750   *->1SSIGN..:....Q...... .q.³....&*
07B60F20 606EF2E4 D9C9F0F0 00000112 07B00DD0   00014010 000141F8 98324F19 43185750   *->2URI00.......}.. ....8q.³....&*
07B60F40 C4E2D7E4 D9C9F0F0 00012E08 00000000   00000000 00000000 98324F19 43185750   *DSPURI00................q.³....&*
07B60F60 D4D6C4C9 D9C3C1D9 F1F24040 D6000000   40D3D6C7 C9C3C1D3 40D6D7C5 D5404040   *MODIRCAR12  O... LOGICAL OPEN   *
07B60F80 40404040 40404040 40404040 40404040   40404040 40404040 00000000 00000000   *                        ........*
07B60FA0 606EF3D9 E2E5F0F0 00001242 0000E1E0   000141F8 00014A78 98324F19 43185750   *->3RSV00.......\...8...q.³....&*
07B60FC0 F24C60D9 E2E5F0F0 00001242 00000226   000141F8 00014A78 98324F19 43186766   *2<-RSV00...........8...q.³.....*
07B60FE0 606EF3E4 D9C9F3F0 00001316 07B08152   000141F8 00014A78 98324F19 43186766   *->3URI30......a....8...q.³.....*
07B61000 606EF4E4 D9C9F2F0 00000F90 07B071A8   00014A78 000150C0 98324F19 43186768   *->4URI20.......y.....&{q.³.....*
07B61020 F34C60E4 D9C9F2F0 00000F90 00000A12   00014A78 000150C0 98324F19 43186768   *3<-URI20.............&{q.³.....*
07B61040 F24C60E4 D9C9F3F0 00001316 00000C8C   000141F8 00014A78 98324F19 43186768   *2<-URI30...........8...q.³.....*
07B61060 C7C5E3C6 C5C5C440 00012E08 00000000   00000000 00000000 98324F19 43187113   *GETFEED ................q.³.....*
07B61080 00000000 00000250 00028800 00028800   00000010 00028800 00028800 0000314A   *.......&..h...h.......h...h....*
07B610A0 00004800 00000020 00000000 00000000   00000000 00000000 00000000 00000009   *................................*
07B610C0 C7C5E3C6 C5C5C440 00012E08 00000000   00000000 00000000 98324F19 43187114   *GETFEED ................q.³.....*
07B610E0 00000000 0000004E 00028800 00028800   00000010 00028800 00028800 0000314A   *.......+..h...h.......h...h....*
07B61100 00004800 00000020 00000000 00000000   00000000 00000000 00000000 00000009   *................................*
07B61120 F14C60E4 D9C9F0F0 00000112 0000079E   00014010 000141F8 98324F19 43187114   *1<-URI00.......... ....8q.³.....*
07B61140 606EF2E4 D9C9F0F0 000002A2 07B00DD0   00014010 000141F8 98324F19 43187115   *->2URI00...s...}.. ....8q.³.....*
07B61160 C4E2D7E4 D9C9F0F0 00012E08 00000000   00000000 00000000 98324F19 43187115   *DSPURI00................q.³.....*
07B61180 D4D6C4C9 D9C3C1D9 F1F24040 D3002000   C4C9D9C5 C3E340D3 D6C3C1E3 C5404040   *MODIRCAR12  L...DIRECT LOCATE   *
07B611A0 FFFFFFFF FFFFFFFF E2E8E2F1 40404040   3F000000 00000000 00000000 00000000   *........SYS1    ................*
07B611C0 C7C5E3C6 C5C5C440 00012E08 00000000   00000000 00000000 98324F19 43187116   *GETFEED ................q.³.....*
07B611E0 00000010 0000004E 00028800 00028800   00000010 00028800 00028800 0000314A   *.......+..h...h.......h...h....*
```

Figure 143. Example of internal trace table entries (Part 2 of 4)

```
07B61200 00004800 00000020 00000000 00000000    00000000 00000000 00000000 00000009   *................................*
07B61220 C4E2D7E4 D9C9F0F0 00012E08 00000000    00000000 00000000 98324F19 43187116   *DSPURI00................q.³.....*
07B61240 C5D5C4C9 D9C3C1D9 F1F24040 D3002000    D9C5C3D6 D9C440D5 D6E340C6 D6E4D5C4   *ENDIRCAR12  L...RECORD NOT FOUND*
07B61260 FFFFFFFF FFFFFFFF E2E8E2F1 40404040    3F000000 00000000 00000000 00000000   *........SYS1    ................*
07B61280 F14C60E4 D9C9F0F0 000002A2 0000079E    00014010 000141F8 98324F19 43187116   *1<-URI00...s...... ....8q.³.....*
07B612A0 606EF2E4 D9E3F0F0 00000384 07B1E300    00014010 000141F8 98324F19 43187116   *->2URT00...d..T... ....8q.³.....*
07B612C0 606EF3C3 C8D2E6C4 00000178 07B50DC8    000141F8 00014A18 98324F19 43187116   *->3CHKWD.......H...8...q.³.....*
07B612E0 F24C60C3 C8D2E6C4 00000178 00000092    000141F8 00014A18 98324F19 43187117   *2<-CHKWD.......k...8...q.³.....*
07B61300 606EF3E4 D9C9F0F0 0000026E 07B00DD0    000141F8 00014A18 98324F19 43187117   *->3URI00...>...}...8...q.³.....*
07B61320 C4E2D7E4 D9C9F0F0 00012E08 00000000    00000000 00000000 98324F19 43187117   *DSPURI00................q.³.....*
07B61340 D4D6C4C9 D9C3C1D9 F1F24040 D6002000    40D3D6C7 C9C3C1D3 40D6D7C5 D5404040   *MODIRCAR12  O... LOGICAL OPEN   *
07B61360 40404040 40404040 40404040 40404040    40404040 40404040 00000000 00000000   *                        ........*
07B61380 F24C60E4 D9C9F0F0 0000026E 0000079E    000141F8 00014A18 98324F19 43187117   *2<-URI00...>.......8...q.³.....*
07B613A0 606EF3E4 D9E3F7F0 0000210C 07B29398    000141F8 00014A18 98324F19 43187117   *->3URT70......lq...8...q.³.....*
07B613C0 606EF4E4 D9C9F0F0 000000B0 07B00DD0    00014A18 00014D68 98324F19 43187117   *->4URI00.......}.....(.q.³.....*
07B613E0 C4E2D7E4 D9C9F0F0 00012E08 00000000    00000000 00000000 98324F19 43187118   *DSPURI00................q.³.....*
07B61400 D4D6C4C9 D9C3C1D9 F1F24040 C2002000    C2C5C740 D4E4D3E3 40E4D7C4 C1E3C540   *MODIRCAR12  B...BEG MULT UPDATE *
07B61420 40404040 40404040 40404040 40404040    40404040 40404040 00000000 00000000   *                        ........*
07B61440 606EF5E4 D9C9F4F0 0000062C 07B0B45C    00014D68 000155E8 98324F19 43187118   *->5URI40.......*..(....Yq.³.....*
07B61460 606EF6E4 D9C9F3F0 00000102 07B0814C    000155E8 000158B8 98324F19 43187118   *->6URI30......a<...Y....q.³.....*
07B61480 F54C60E4 D9C9F3F0 00000102 00000BB0    000155E8 000158B8 98324F19 43187723   *5<-URI30...........Y....q.³.....*
07B614A0 F44C60E4 D9C9F4F0 0000062C 00000122    00014D68 000155E8 98324F19 43187723   *4<-URI40..........(....Yq.³.....*
07B614C0 F34C60E4 D9C9F0F0 000000B0 0000079E    00014A18 00014D68 98324F19 43187723   *3<-URI00.............(.q.³.....*
07B614E0 606EF4E4 D9C9F0F0 0000011C 07B00DD0    00014A18 00014D68 98324F19 43187724   *->4URI00.......}.....(.q.³.....*
07B61500 C4E2D7E4 D9C9F0F0 00012E08 00000000    00000000 00000000 98324F19 43187724   *DSPURI00................q.³.....*
07B61520 D4D6C4C9 D9C3C1D9 F1F24040 D3002000    C4C9D9C5 C3E340D3 D6C3C1E3 C5404040   *MODIRCAR12  L...DIRECT LOCATE   *
JOB DBRLATAM         STEP DBRLATAM        TIME 114417   DATE 98324                                         PAGE 00000328

07B61540 FFFFFFFF FFFFFFFF E2E8E2F1 40404040    3F000000 00000000 00000000 00000000   *........SYS1    ................*
07B61560 C7C5E3C6 C5C5C440 00012E08 00000000    00000000 00000000 98324F19 43187725   *GETFEED ................q.³.....*
07B61580 00000010 0000004E 00028800 00028800    00000010 00028800 00028800 0000314A   *.......+..h...h.......h...h....*
07B615A0 00004800 00000020 00000000 00000000    00000000 00000000 00000000 00000009   *................................*
07B615C0 C4E2D7E4 D9C9F0F0 00012E08 00000000    00000000 00000000 98324F19 43187725   *DSPURI00................q.³.....*
07B615E0 C5D5C4C9 D9C3C1D9 F1F24040 D3002000    D9C5C3D6 D9C440D5 D6E340C6 D6E4D5C4   *ENDIRCAR12  L...RECORD NOT FOUND*
07B61600 FFFFFFFF FFFFFFFF E2E8E2F1 40404040    3F000000 00000000 00000000 00000000   *........SYS1    ................*
07B61620 F34C60E4 D9C9F0F0 0000011C 0000079E    00014A18 00014D68 98324F19 43187725   *3<-URI00.............(.q.³.....*
07B61640 606EF4E4 D9C9F0F0 000001F8 07B00DD0    00014A18 00014D68 98324F19 43187725   *->4URI00...8...}.....(.q.³.....*
07B61660 C4E2D7E4 D9C9F0F0 00012E08 00000000    00000000 00000000 98324F19 43187726   *DSPURI00................q.³.....*
07B61680 D4D6C4C9 D9C3C1D9 F1F24040 E6082000    C9D5E2C5 D9E340D5 C5E640D9 C5C3D9C4   *MODIRCAR12  W...INSERT NEW RECRD*
07B616A0 FFFFFFFF FFFFFFFF E2E8E2F1 40404040    3F000000 00000000 00000000 00000000   *........SYS1    ................*
07B616C0 606EF5E3 C9D4C5F0 00000478 07B4B298    00014D68 000155E8 98324F19 43187726   *->5TIME0.......q..(....Yq.³.....*
07B616E0 F44C60E3 C9D4C5F0 00000478 00000FFE    00014D68 000155E8 98324F19 43187727   *4<-TIME0..........(....Yq.³.....*
07B61700 606EF5E4 D9C9F4F0 00002F44 07B0B468    00014D68 000155E8 98324F19 43187728   *->5URI40..........(....Yq.³.....*
07B61720 F44C60E4 D9C9F4F0 00002F44 000000A6    00014D68 000155E8 98324F19 43188448   *4<-URI40.......w..(....Yq.³...d.*
07B61740 C7C5E3C6 C5C5C440 00012E08 00000000    00000000 00000000 98324F19 43188856   *GETFEED ................q.³...h.*
07B61760 00000000 00000048 00028800 00028800    00000010 00028800 00028800 0000314A   *..........h...h.......h...h....*
07B61780 00004800 00000020 00000000 00000000    00000000 00000000 00000000 00000009   *................................*
07B617A0 C7C5E3C6 C5C5C440 00012E08 00000000    00000000 00000000 98324F19 43189271   *GETFEED ................q.³...k.*
07B617C0 00000000 00000048 00028800 00028800    00000010 00028800 00028800 0000314A   *..........h...h.......h...h....*
07B617E0 00004800 00000020 00000000 00000000    00000000 00000000 00000000 00000009   *................................*
07B61800 F34C60E4 D9C9F0F0 000001F8 0000079E    00014A18 00014D68 98324F19 43189271   *3<-URI00...8.........(.q.³...k.*
07B61820 606EF4E4 D9C9F0F0 00000268 07B00DD0    00014A18 00014D68 98324F19 43189272   *->4URI00.......}.....(.q.³...k.*
07B61840 C4E2D7E4 D9C9F0F0 00012E08 00000000    00000000 00000000 98324F19 43189272   *DSPURI00................q.³...k.*
07B61860 D4D6C4C9 D9C3C1D9 F1F24040 C5082000    C5D5C440 D4E4D3E3 40E4D7C4 C1E3C540   *MODIRCAR12  E...END MULT UPDATE *
07B61880 40404040 40404040 40404040 40404040    40404040 40404040 00000000 00000000   *                        ........*
07B618A0 606EF5E4 D9C9F4F0 00000646 07B0B462    00014D68 000155E8 98324F19 43189272   *->5URI40..........(....Yq.³...k.*
07B618C0 606EF6E4 D9C9F3F0 00000162 07B0814C    000155E8 000158B8 98324F19 43189272   *->6URI30......a<...Y....q.³...k.*
07B618E0 F54C60E4 D9C9F3F0 00000162 00000BB0    000155E8 000158B8 98324F19 43189998   *5<-URI30...........Y....q.³...rq*
07B61900 606EF6E4 D9C9F3F0 0000017C 07B0814C    000155E8 000158B8 98324F19 43190682   *->6URI30...@..a<...Y....q.³....b*
07B61920 F54C60E4 D9C9F3F0 0000017C 00000BB0    000155E8 000158B8 98324F19 43191334   *5<-URI30...@.......Y....q.³.....*
07B61940 F44C60E4 D9C9F4F0 00000646 000001A0    00014D68 000155E8 98324F19 43191334   *4<-URI40..........(....Yq.³.....*
07B61960 F34C60E4 D9C9F0F0 00000268 0000079E    00014A18 00014D68 98324F19 43191334   *3<-URI00.............(.q.³.....*
07B61980 F24C60E4 D9E3F7F0 0000210C 000002B8    000141F8 00014A18 98324F19 43191335   *2<-URT70...........8...q.³.....*
07B619A0 606EF3E4 D9C9F0F0 000002BC 07B00DD0    000141F8 00014A18 98324F19 43191335   *->3URI00.......}...8...q.³.....*
```

Figure 143. Example of internal trace table entries (Part 3 of 4)

```
07B619C0 C4E2D7E4 D9C9F0F0 00012E08 00000000    00000000 00000000 98324F19 43191335   *DSPURI00................q.³.....*
07B619E0 D4D6C4C9 D9C3C1D9 F1F24040 C3082000    40D3D6C7 C9C3C1D3 40C3D3D6 E2C54040   *MODIRCAR12  C... LOGICAL CLOSE  *
07B61A00 40404040 40404040 40404040 40404040    40404040 40404040 00000000 00000000   *                        ........*
07B61A20 F24C60E4 D9C9F0F0 000002BC 0000079E    000141F8 00014A18 98324F19 43191335   *2<-URI00...........8...q.³.....*
07B61A40 F14C60E4 D9E3F0F0 00000384 000002E6    00014010 000141F8 98324F19 43191335   *1<-URT00...d...W.. ....8q.³.....*
07B61A60 606EF2E4 D9C9F0F0 000001CA 07B00DD0    00014010 000141F8 98324F19 43191336   *->2URI00.......}.. ....8q.³.....*
07B61A80 C4E2D7E4 D9C9F0F0 00012E08 00000000    00000000 00000000 98324F19 43191336   *DSPURI00................q.³.....*
07B61AA0 D4D6C4C9 D9C3C1D9 F1F24040 C3082000    40D3D6C7 C9C3C1D3 40C3D3D6 E2C54040   *MODIRCAR12  C... LOGICAL CLOSE  *
07B61AC0 40404040 40404040 40404040 40404040    40404040 40404040 00000000 00000000   *                        ........*
07B61AE0 606EF3C4 C5D8F0F0 000037C8 07B0CC10    000141F8 00014A78 98324F19 43191336   *->3DEQ00...H.......8...q.³.....*
07B61B00 F24C60C4 C5D8F0F0 000037C8 000006F6    000141F8 00014A78 98324F19 43192645   *2<-DEQ00...H...6...8...q.³.....*
```

Figure 143. Example of internal trace table entries (Part 4 of 4)

DBRC External Trace

If you start the Generalized Trace Facility (GTF) and enter the CHANGE.RECON TRACEON command, the DBRC trace (DSPTRACE) creates an external trace record and issues the GTRACE macro to invoke GTF. The GTRACE macro passes the address and length of a DBRC external trace record to GTF. A DBRC external trace record is put in the user data area of a GTF trace record.

If more than two DBRC jobs run concurrently, the GTF data set or buffer can contain multiple trace records. Therefore, DBRC external trace records contain either the IMS subsystem ID or a job name. In a DB/DC or DBCTL environment, the SSID is added to the trace record. In other IMS environments, a job name is added to the trace record. Figure 144 shows the format of these records.

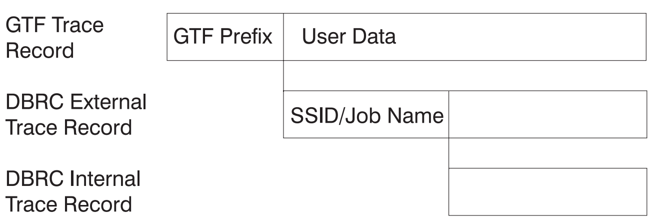


Figure 144. Format of Trace Records

The GTF cataloged procedure is supplied in SYS1.PROCLIB with member name GTF or GRFSNP. If you want the DBRC trace records to be put in the GTF data set, specify MODE=EXT on the EXEC parameter and USR on the GTF option in the cataloged procedure. For detailed information about invoking GTF and its cataloged procedure, see *MVS/ESA Diagnosis: Tools and Service Aids*.

You can format and print DBRC trace records in the GTF data set by using the GTFTRACE subcommand of IPCS. You must specify the exit HMDUSRF2 on this subcommand. For detailed information about using IPCS, see *OS/390 MVS IPCS User's Guide*.

Examples of Output

The following two examples show the unformatted and then formatted output for DBRC router processing and RECON I/O error processing.

In Figure 145 on page 385:

- DBRCJOB1 is the job name.
- TIME is the time stamp of the trace entry.
- DSPCRTR0 passed control to the next routine to process the request identified by the DFSBRLSB.
- GDB is the address of the Global Data Block.
- LSB is the address of the DFSBRLSB.

- FUNC indicates the function flags (from the BRLBFFLG field of the DFSBRLSB).
- EXIT indicates the exit flags (from the BRLBEFLG field of the DFSBRLSB).

```
GTF USR Record containing DBRC Unformatted Trace Record Data
HEXFORMAT AID FF FID F2 EID  EFAD
 +0010  00000000  C4E2D7C3  D9E3D9F0  05F79C94  ³ ....DSPCRTR0.7.m ³
 +0020  17172002  00000000  00012D78  99085F22  ³ ............r... ³
 +0030  48397685  00000000  00000000  00D4C080  ³ ...e.........M{. ³
 +0040  00000001  00012D78  00000000  00000000  ³ ................ ³
 +0050  00000000  00000000  00000000  00000000  ³ ................ ³
 +0060  00000000  00000000  00000000  00000000  ³ ................ ³
 +0070  00000000                                ³ ....             ³
Formatted Output
........  TIME=99085F2248397685  DSPCRTR0  GDB=00012D78 LSB=05F79C94 FUNC=17172002 EXIT=00000000
00000000 00000000 00D4C080 00000001   00012D78 00000000 00000000 00000000   *.........M......................*
00000000 00000000 00000000 00000000   00000000 00000000 00000000 00000000   *................................*
```

Figure 145. DBRC External Trace Output for DBRC Router Processing

In Figure 146 a SHOWCB macro instruction was executed after the I/O request was issued.

- IMS1 is the SYSID.
- TIME is the time stamp of the trace entry.
- DSPURI00 has control.
- GDB is the address of the Global Data Block.
- A locate was done. For a locate, a flag and record key are also shown in the trace record.
- RSCD is the VSAM reason code.

```
GTF USR Record containing DBRC Unformatted Trace Record Data
 HEXFORMAT AID FF FID F2 EID  EFAD
  +0000  00FA2980  C4C2D9D6  C3E3C1D4  C9D4E2F1  | ....DBROCTAMIMS1 |
  +0010  40404040  C4E2D7E4  D9C9F0F0  00012D78  |     DSPURI00.... |
  +0020  00000000  00000000  00000000  99085F22  | ............r.¬. |
  +0030  48398254  C4E2D7C9  D9C3C1D9  00000190  | ..b.DSPIRCAR.... |
  +0040  D3002000  00000000  00000000  FFFFFFFF  | L............... |
  +0050  FFFFFFFF  C9D4E2F1  40404040  3F000000  | ....IMS1    .... |
  +0060  00000000  00000000  00000000  00000000  | ................ |
  +0070  00000000                                | ....             |
Formatted Output
IMS1      TIME=99085F2248398254  DSPURI00  GDB=00012D78 FUNC=LOCATE  FLAG=0020
RECKEY=FFFFFFFFFFFFFFFFC9D4E2F1404040403F00000000000000000000000000000
```

Figure 146. DBRC External Trace Output for RECON I/O Error Processing

Samples of JCL to Create Trace Output

Here is a sample of a job that was used to create unformatted USR(FAD) trace output:

```
//PRTUSRF2 JOB IMSCVT8,MSGLEVEL=1,CLASS=K,MSGCLASS=A,REGION=4096K
//*******************************************************************
//* JOB NAME:        PRINTGTF JCL                                   *
//* JOB DEPENDENCIES: The GTF data set named below must exist.      *
//* JOB Source:       See the IPCS User's Guide, Appendix B.        *
//* JOB DESCRIPTION: This job prints the specified GTF data set using *
//* the Batch IPCS feature.                                         *
//*******************************************************************
/*ROUTE PRINT THISCPU/IMSM3405
//*OBLIB   DD DSN=IMSTESTL.TNUC0,DISP=SHR
//*        DD DISP=SHR,DSN=IMSBLD.I710TS25.CRESLIB
//*        DD DISP=SHR,DSN=IMSTESTG.IMS710.TSTRES
//*        DD DISP=SHR,DSN=IMSTESTG.IMSQA.ACPLIB
//*        DD DISP=SHR,DSN=IMSTESTG.IMSQA.PGMLIB
```

```
//JOBCAT   DD DISP=SHR,DSN=VCATQAV
//         DD DISP=SHR,DSN=VCATDCL
//*******************************************************************
//* Print the SYS1.TRACE data set.                                  *
//* Member BLSCDDIR resides in SYS1.SBLSCLI0, an IPCS system proclib. *
//* IT ISSUES THE DEFINE CLUSTER FOR 'DBRX06.IPCS.DDIR' ON USER01 AND *
//* catalogs it in SYS1.ECTEST.MASTER.CATALOG.                      *
//*******************************************************************
//IPCS     EXEC  PGM=IKJEFT01,DYNAMNBR=20,REGION=1500K
//TRACE    DD  DSN=SYS1.TRACE,DISP=SHR,
//         UNIT=SYSDA,VOL=SER=000000
//SYSPROC  DD  DSN=SYS1.SBLSCLI0,DISP=SHR
//SYSTSPRT DD  SYSOUT=A
//IPCSPRNT DD  SYSOUT=A
//IPCSTOC  DD  SYSOUT=A
//SYSUDUMP DD  SYSOUT=A
//SYSTSIN  DD *
 PROFILE MSGID
 %BLSCDDIR DSNAME(DBRX06.IPCS.DDIR) VOLUME(USER01)
 IPCS NOPARM
 SETDEF DDNAME(TRACE) NOCONFIRM
 GTFTRACE USR(FAD)
 END
/*
//*******************************************************************
//*   Delete the IPCS dump directory created by the previous step     *
//*   so that the re-IPL of the ec machine will not orphan the data   *
//*   set.                                                          *
//*******************************************************************
//AMS01    EXEC PGM=IDCAMS,COND=EVEN
//SYSPRINT DD   SYSOUT=A
//DD1      DD   UNIT=SYSDA,VOL=SER=USER01,DISP=SHR
//SYSIN    DD   *
  DELETE DBRX06.IPCS.DDIR FILE(DD1)
/*
```

Here is a sample of a job that was used to create the DBRC formatted output:

```
//PRINTHMD JOB IMSCVT8,MSGLEVEL=1,CLASS=K,MSGCLASS=A,REGION=4096K
//*******************************************************************
//* JOB NAME:        PRINTHMD JCL                                   *
//* JOB DEPENDENCIES: The GTF data set named below must exist.       *
//* JOB Source:       See the IPCS User's Guide, Appendix B.         *
//* JOB DESCRIPTION: This job prints the specified GTF data set using *
//* the Batch IPCS feature.                                         *
//*******************************************************************
/*ROUTE PRINT THISCPU/IMSM3405
//JOBLIB   DD DSN=IMSTESTL.TNUC0,DISP=SHR
//         DD DISP=SHR,DSN=IMSBLD.I710TS25.CRESLIB
//         DD DISP=SHR,DSN=IMSTESTG.IMS710.TSTRES
//         DD DISP=SHR,DSN=IMSTESTG.IMSQA.ACPLIB
//         DD DISP=SHR,DSN=IMSTESTG.IMSQA.PGMLIB
//JOBCAT   DD DISP=SHR,DSN=VCATQAV
//         DD DISP=SHR,DSN=VCATDCL
//*******************************************************************
//* Print the SYS1.TRACE data set.                                  *
//* Member BLSCDDIR resides in SYS1.SBLSCLI0, an IPCS system proclib. *
//* IT ISSUES THE DEFINE CLUSTER FOR 'DBRX06.IPCS.DDIR' ON USER01 AND *
//* catalogs it in SYS1.ECTEST.MASTER.CATALOG.                      *
//*******************************************************************
//IPCS     EXEC  PGM=IKJEFT01,DYNAMNBR=20,REGION=1500K
//TRACE    DD  DSN=SYS1.TRACE,DISP=SHR,
//         UNIT=SYSDA,VOL=SER=000000
//SYSPROC  DD  DSN=SYS1.SBLSCLI0,DISP=SHR
//SYSTSPRT DD  SYSOUT=A
//IPCSPRNT DD  SYSOUT=A
//IPCSTOC  DD  SYSOUT=A
```

```
//SYSUDUMP DD  SYSOUT=A
//SYSTSIN  DD *
 PROFILE MSGID
 %BLSCDDIR DSNAME(DBRX06.IPCS.DDIR) VOLUME(USER01)
 IPCS NOPARM
 SETDEF DDNAME(TRACE) NOCONFIRM
 GTFTRACE EXIT(HMDUSRF2)
 END
/*
//*********************************************************************
//*   Delete the IPCS dump directory created by the previous step     *
//*   so that the re-IPL of the ec machine will not orphan the data   *
//*   set.                                                            *
//*********************************************************************
//AMS01    EXEC PGM=IDCAMS,COND=EVEN
//SYSPRINT DD   SYSOUT=A
//DD1      DD   UNIT=SYSDA,VOL=SER=USER01,DISP=SHR
//SYSIN    DD   *
  DELETE DBRX06.IPCS.DDIR FILE(DD1)
/*
```

Chapter 14. DRA—Database Resource Adapter Service Aids

In a Database Control (DBCTL) environment, if you think the coordinator controller (CCTL) did not cause the problem, then start your analysis here.

This chapter provides service aids and tips that can help you analyze problems in a Database Control (DBCTL) environment. It discusses:
- DRA dumps
- Analyzing DRA problems

The DRA is the interface between DBCTL and the CCTL. The functions of the DRA are to:
- Request connection to and disconnection from DBCTL
- Tell the CCTL when DBCTL has failed or when the operator has requested a shutdown
- Manage threads

For a description of the DRA interface, see the *IMS Version 7 Customization Guide*.

DRA Dumps

The DRA creates a dump when a DRA request fails or when DRA processing fails. A DRA request is a request (such as INIT or TERMINATE) made by the CCTL that has passed through the DRA. A DRA request failure produces either a system abend or an IMS pseudoabend. A DRA processing failure produces a system abend. For either type of failure, the DRA first tries to take an MVS SDUMP. If that fails, the DRA takes a SNAP dump. In some situations the DRA takes a SNAP dump without attempting an SDUMP. For certain pseudoabends, the DRA produces neither an SDUMP nor a SNAP.

To determine what type of dump the DRA created, check field PAPLRETC in the DFSPAPL (the parameter list used to pass information between the CCTL and DBCTL). PAPLRETC has the format:

```
hhsssuuu
```

where hh indicates the type of dump.

The following table lists the values for *hh* and tells which dump the DRA creates for different types of failures.

Table 73. Determining the Type of Dump the DRA Created

hh	Type of Dump	Failures
X'80'	SDUMP or SNAP	An SDUMP is taken for all IMS abend codes not listed below, and for all MVS retryable abend codes. If the SDUMP fails, a SNAP is taken.
X'84'	SNAP	A SNAP is taken for IMS abend codes U0260, U0261, and U0263.
X'88'	No dump	No SDUMP or SNAP is taken for: • IMS abend codes U0775, U0777, U2478, U2479, U3303 • MVS nonretryable abend codes (for example, S222, S13E) • DRA return codes (See *IMS Version 7 Messages and Codes, Volume 1* for DRA return codes and their meanings.)

SDUMP

SDUMP output contains:
- IMS control region
- DLISAS address space

- Key 0 and key 7 CSA
- Selected parts of DRA private storage, including the ASCB, TCB, and RBs

A DRA SDUMP has its own SDUMP option list. To add to the DRA's SDUMP option list, you can use the CHNGDUMP parameter. However, you cannot use CHNGDUMP to delete areas from the list.

You can format the IMS control blocks by using the Offline Dump Formatter (ODF) described in "Formatting IMS Dumps Offline" on page 129. The ODF does not format DRA storage. You can use IPCS to format the MVS blocks in the CCTL's private storage.

SNAPs

The SNAP dump data sets are dynamically allocated whenever a SNAP is needed. A parameter in the DRA Startup Table defines the SYSOUT class.

SNAP output contains:

- Selected parts of DRA private storage, including the ASCB, TCB, and RBs
- DBCTL's thread blocks

Recovery Tokens

In a DBCTL environment, you need to correlate the information produced by the CCTL with information produced by DBCTL. The link between the CCTL and DBCTL is the recovery token, which uniquely identifies each unit of recovery (UOR).

The recovery token appears in the DRA dump (both SDUMPs and SNAPs) and in the dump title. It contains a mixture of EBCDIC and hexadecimal data and has the following format:

CCTL subsystem ID	Unique UOR ID (created by the CCTL)
8 bytes (EBCIDIC)	8 bytes (hexadecimal)

Analyzing DRA Problems

To analyze DRA problems, first investigate any external conditions that might have caused the problem. If you can eliminate external causes, then an unexpected DBCTL return code or another IMS function might have caused the problem. Follow these steps to analyze the problem.

Procedure

1. Did external conditions cause the problem?
   - For CCTL external problems, check the status of applications or transactions. DBCTL and the DRA do not control these resources.
   - For DBCTL external problems, check the status of databases, PSBs, and dependent regions (BMPs and CCTLs) by using the `/DISPLAY` commands.
   - For DRA external problems:
     - Make sure you are using the correct DRA startup table for this DBCTL/CCTL session. Values such as Fast Path buffer allocations and minimum/maximum thread specifications can cause scheduling and resource problems.
     - Become familiar with the CCTL control exit.

       The DRA calls the control exit to notify the CCTL of certain events, such as a DRA failure, an identify failure, a DBCTL failure, and so on. The DRA passes this information in a parameter list (DFSPAPL). The CCTL responds by passing back a return code in field PAPLRETC to tell the

DRA what action to perform. Understanding which actions the CCTL is allowed to request can help you distinguish between valid actions and failures.

For a detailed description of the control exit, see *IMS Version 7 Customization Guide*. For information about the codes passed between the DRA and the CCTL, see *IMS Version 7 Messages and Codes, Volume 1*.

– The DRA does not issue any messages that report the actions it performed.

- If an external condition caused the problem, stop here and fix the problem. Otherwise, continue with the next step.

2. You reach this point by eliminating external reasons as the cause of the problem.
   - Determine if DBCTL returned a nonzero return code, indicating that the request from the CCTL was not successfully completed. For a description of DBCTL return codes, see *IMS Version 7 Messages and Codes, Volume 1*.
     – If yes, take an MVS online dump of the CCTL and contact the IBM Support Center.
     – If no, then other functions might be involved in the problem. Use the appropriate chapter in this manual to analyze the problem. The keyword procedures in Chapter 4, “Selecting the Keywords,” on page 19 are useful in narrowing the problem to a specific cause.

Notes on Dumping

For suspected problems in a DBCTL environment, first take a dump of the CCTL address space. Dumps produced by SDUMP and by specifying the DUMP option on the CCTL /SHUTDOWN command are acceptable for problem diagnosis. If IMS service needs to analyze the CCTL dump, send the unformatted dump to enable them to obtain DBCTL DRA storage.

Chapter 15. RSR—Remote Site Recovery Service Aids

This chapter provides Fast Path Tracker Trace Entries (“Fast Path Tracker Trace Entries” on page 395) and Database Tracker Trace Entries (“Database Tracker Trace Entries” on page 411) that might help you analyze problems in a Remote Site Recovery (RSR) Environment.

The RSR tracking process creates a local log that mirrors the activity at the currently active system.

In some cases, however, the tracking system might not receive copies of all log records before takeover. This might happen if there is a tracking session failure before takeover occurs while the active system is still processing transactions normally. If there is a tracking session failure before takeover, subsequent attempts to start Finance, SLU P, and ISC sessions or MSC links might result in resynchronization errors.

The MTO is notified of both non-MSC errors and MSC errors. as follows:
- Message DFS2948 notifies the MTO of non-MSC errors.
- Either message DFS3211 or message DFS3212 notifies the MTO of MSC errors.

Use the remote takeover message information in conjunction with the received log data to determine the last terminal or MSC message recorded by the tracking process. Then input or output any messages that were lost.

Determining Last Non-MSC Message Recorded

Non-MSC, Non-Fast Path Messages

For a non-MSC, non-Fast Path message, use the following procedure to determine the last input or output message recorded via RSR tracking and its status within the new active IMS following takeover.

1. Print all these log records for information:

   X’01’
   X’03’
   X’31’
   X’35’
   X’36’
   X’37’
   X’63’
   X’66’
2. Determine the last input or output message. First look for the last X’66’ or X’63’ log record for the terminal.

   ISC parallel sessions qualify the node name in the log record with user ID.

   If an X’63’ log record is last, that indicates whether the session was started cold (without message numbers) or warm (with last input/output message numbers).

   If an X’66’ log record is last, that log record will indicate the message sequence number and whether the message was input or output. The X’66’ log record marks an attempt to commit the message for recovery and restart, if necessary. Additional log records will indicate the exact status of the message.
3. Determine the last committed input message by inspecting the last X’66’ marked as input for the specific terminal. It will be followed by X’01’ and X’35’ log records for the input message. The X’35’ log record considers the input message (log record X’66’) committed, or made recoverable, for input processing on nonresponse mode transactions.

   Restriction: Nonconversational response mode transactions are *not* restartable. That is, they must be resubmitted to IMS if any failure occurs prior to completion of transaction processing. Therefore, the input is not considered committed until the transaction processing is complete and output is available to send to the terminal (see output process that follows).

4. Before the terminal begins the output process, completion of the input transaction processing results in an X’03’, ending with an X’3730.’ The X’3730’ commits the transaction changes, including making the output message available for the terminal. The X’3730’ also commits the associated nonconversational response mode input transaction, as described above.

   To determine the last committed output message sent to the terminal. begin with the last X’66’ marked as output. This output message is committed, that is dequeued, with the following X’36’ log record that follows, reflecting successful receipt by the terminal.

Fast Path Messages

For Fast Path messages, use the following procedure to determine the last input or output message recorded via RSR tracking.

1. Print all these log records for information:

   X’5901’

   X’5903’

   X’5936’

   X’5937’

   X’63’

   X’66’
2. Determine the last input or output message. First look for the last X’66’ or X’63’ log record for the terminal.

   ISC parallel sessions qualify the node name in the log record with user ID.

   If an X’63’ log record is last, that indicates whether the session was started cold (without message numbers) or warm (with last input/output message numbers).

   If an X’66’ log record is last, that log record will indicate the message sequence number and whether the message was input or output. The X’66’ log record marks an attempt to commit the message for recovery and restart, if necessary. Additional log records will indicate the exact status of the message.
3. Fast Path input is always considered nonrestartable and must be resubmitted to IMS if any failure occurs before transaction input processing is complete and the output message is made available to the terminal output process.
4. To determine the last Fast Path input transaction received and committed, begin with the last X’66’ marked as input for the specific terminal. It will be followed by an X’5901’ with the input message and an X’5937’ indicating input transaction processing complete. The input and all changes have been committed.
5. To determine the last committed output message to the terminal, begin with the X’5903’ for the output message followed by the X’5937’, which makes it available for the terminal output process. This is the same X’5937’ that also commits the input above. This will be followed by an X’66’ log record indicating an attempt to deliver output to the terminal. This output is committed (dequeued) when also followed by the X’5936’ log record.

Determining Last MSC Message Recorded

MSC links keep track of the sending and receiving of data on a message by message basis. Each message block sent across an MSC link is appended with a sequence number. The IMS receiving system updates its receive count with each message block received, and records (logs) each message successfully received and enqueued to the message queue. Similarly, the sending system updates its sending count with each message block sent and logs the sequence number of the last message successfully sent and dequeued.

Across link restarts, RSR takeovers, or IMS failures, these sequence numbers are exchanged and used to resynchronize the message traffic, to continue sending and receiving messages at the same point. Therefore, messages are not lost or duplicated.

The key to the success of this concept is the logging of the messages that were sent and received across the link, and enqueued on the receiving side and dequeued from the sending side. There are primarily five log records used to resynchronize this message traffic. They are:

01 - Input message to IMS - input transaction or message switch

03 - Transaction Output, program-to-program switch or error message (DFSxxxx)

35 - Enqueue message

36 - Dequeue message

66 - Message sequence recovery

If log records are lost and not processed by the tracking system prior to a remote takeover, message resynchronization may result in the loss or duplication of messages. This may be evidenced by error messages that are issued by IMS when the links are restarted, such as DFS3211 and DFS3212, DFS2145, and DFS2948.

Should link resynchronization fail after an RSR takeover, it may be possible to analyze which messages were lost or duplicated, from the information in the DFS error message issued by IMS at the time of error, and from the 01, 03, 35, 36, and 66 log records.

Fast Path Tracker Trace Entries

Trace Entry: Fast Path Tracker Log Router Interface (9E)

9E01

Table 74. Trace Record 9E01 - DBFDT210 Redo Record Processor Module Entry

Module: DBFDT210 Redo Record Processor Module Entry

Explanation: Record cut at entry to DBFDT210 (Level - High)

Trace Subcode DT210 Entry

Offset	Type	Length	Description
0	Fixed	4	Log Id
4	Character	20	LPD Volatile

Example:

```
                          LSN       streamID  OFRID
                          |         |         |
DT210 Entry     9E018A65  000023AB  00000001  00000000
                00000090  0094122F  1141138F  8613CD64
                |         |
                milestone prilog time
                index
```

9E02

Table 75. Trace Record 9E02 - DBFDT220 Commit/Abort Record Processor Module Entry

Module: DBFDT220 Commit/Abort Record Processor Module Entry

Explanation: Record cut at entry to DBFDT220 (Level - High)

Trace Subcode DT220 Entry

Table 75. Trace Record 9E02 - DBFDT220 Commit/Abort Record Processor Module Entry (continued)

Offset	Type	Length	Description
0	Fixed	4	Log Id
4	Character	20	LPD Volatile

Example:

```
                          LSN       streamID  OFRID
                          |         |         |
DT220 Entry     9E028A69  000023AD  00000001  00000000
                00000090  0094122F  1141138F  8613CFC0
                |         |
                milestone prilog time
                index
```

9E03

Table 76. Trace Record 9E03 - DBFDT255 Commit Redo Record Processor Module Entry

Module: DBFDT255 Commit Redo Record Processor Module Entry

Explanation: Record cut at entry to DBFDT255 (Level - High)

Trace Subcode DT255 Entry

Offset	Type	Length	Description
0	Fixed	4	Log Id
4	Character	20	LPD Volatile

Example:

```
                          LSN       streamID  OFRID
                          |         |         |
DT255 Entry     9E03B863  000018E2  00000001  00000000
                00000050  0092314F  1432062F  A0AF4B90
                |         |
                milestone prilog time
                index
```

9E04

Table 77. Trace Record 9E04 - DBFDT260 End Update/End Active Stream Module Entry

Module: DBFDT260 End Update/End Active Stream Module Entry

Explanation: Record cut at entry to DBFDT260 (Level - High)

Trace Subcode DT260 Entry

Offset	Length	Length	Description
0	Fixed	4	Log Id
4	Character	20	LPD Volatile

Example:

```
                          LSN       streamed  AFRAID
                          |         |         |
DT260 Entry     9E04CD09  0000250A  00000001  00000000
```

```
00000003  0094122F  1430091F  9D9846A0
|         |
milestone prilog time
index
```

9E05

Table 78. Trace Record 9E05 - DBFDT256 Init/Term Record Processor Module Entry

Module: DBFDT256 Init/Term Record Processor Module Entry

Explanation: Record cut at entry to DBFDT256 (Level - High)

Trace Subcode DT256 Entry

Offset	Type	Length	Description
0	Fixed	4	Log Id
4	Character	20	LPD Volatile

Example:

```
                LSN       streamID  OFRID
                |         |         |
DT256 Entry     9E05CFE0  000025E2  00000001  00000000
                00000003  0094122F  1430091F  A1F51BDC
                |         |
                milestone prilog time
                index
```

9E06

Table 79. Trace Record 9E06 - DBFDT210 NoTUR

Module: DBFDT210 NoTUR

Explanation: Record cut in DBFDT210 when no storage for TUR in data space (Level - High)

Trace Subcode DT210 NoTUR

Offset	Type	Length	Description
0	Fixed	4	Log Id
4	Character	20	LPD Volatile

9E07

Table 80. Trace Record 9E07 - DBFDT210 NoERQE

Module: DBFDT210 NoERQE

Explanation: Record cut in DBFDT210 when no storage for ERQE in data space (Level - High)

Trace Subcode DT210 NoERQE

Offset	Type	Length	Description
0	Fixed	4	Log Id
4	Character	20	LPD Volatile

9E08

Table 81. Trace Record 9E08 - DBFDT255 NoERQE

Module: DBFDT255 NoERQE

Table 81. Trace Record 9E08 - DBFDT255 NoERQE (continued)

Explanation: Record cut in DBFDT255 when no storage for ERQE in data space (Level - High)

Trace Subcode DT255 NoERQE

Offset	Type	Length	Description
0	Fixed	4	Log Id
4	Character	20	LPD Volatile

9E09

Table 82. Trace Record 9E09 - DBFDT260 NoERQE

Module: DBFDT260 NoERQE

Explanation: Record cut in DBFDT260 when no storage for ERQE in data space (Level - High)

Trace Subcode DT260 NoERQE

Offset	Type	Length	Description
0	Fixed	4	Log Id
4	Character	20	LPD Volatile

9E0A

Table 83. Trace Record 9E0A - DBFDT210 Redo Record Processor Module Entry2

Module: DBFDT210 Redo Record Processor Module Entry2

Explanation: Record cut at entry to DBFDT210 (Level - High)

Trace Subcode DT210 Entry2

Offset	Type	Length	Description
0	Fixed	4	Stream type
4	Character	4	RBA
8	Character	2	Data offset
10	Character	2	Data length

Example:

```
                          stream            offset
                          type      RBA     |   length
                          |         |       |   |
DT210 Entry2    9E0A8A66  00000000  00004800  000000B0
                00000000  00000000  00000000  8613CD97
```

9E0B

Table 84. Trace Record 9E0B - DBFDT220 Commit/Abort Record Processor Module Entry2

Module: DBFDT220 Commit/Abort Record Processor Module Entry2

Explanation: Record cut at entry to DBFDT220 (Level - High)

Trace Subcode DT220 Entry2

Offset	Type	Length	Description
0	Fixed	4	Stream type

Table 84. Trace Record 9E0B - DBFDT220 Commit/Abort Record Processor Module Entry2 (continued)

4	Fixed	1	Log Record Code
5	Fixed	1	Log Record Subcode

Example:

```
                         stream    log record code
                         type      & subcode
                         |         |
DT220 Entry2    9E0B8A6A  00000000  59370000  00000000
                00000000  00000000  00000000  8613CFF3
```

9E11

Table 85. Trace Record 9E11 - DBFDT262 End Active Stream Notify Processor Module Entry

Module: DBFDT262 End Active Stream Notify Processor Module Entry

Explanation: Record cut at entry to DBFDT262 (Level - High)

Trace Subcode DT262 Entry

Offset	Type	Length	Description
0	Character	8	Prilog Time
8	Fixed	4	Milestone Index
12	Fixed	1	Stream Id

Example:

```
                         prilog time         milestone index
                         |                   |
DT262 Entry     9E11D32B  0094122F  1430091F  00000003
                01000000  00000000  00000000  A60A48AD
                |
                streamID
```

9E12

Table 86. Trace Record 9E12 - DBFDT270 Begin/End OFR Stream Processor Module Entry

Module: DBFDT270 Begin/End OFR Stream Processor Module Entry

Explanation: Record cut at entry to DBFDT270 (Level - High)

Trace Subcode DT270 Entry

Offset	Type	Length	Description
0	Fixed	4	OFR Id
4	Fixed	4	OFRL Number Entries
8	Address	4	OFRL Fast Path Area (1)

9E13

Table 87. Trace Record 9E13 - DBFDT271 Restart OFR Stream Processor Module Entry

Module: DBFDT271 Restart OFR Stream Processor Module Entry

Explanation: Record cut at entry to DBFDT271 (Level - High)

Trace Subcode DT271 Entry

Table 87. Trace Record 9E13 - DBFDT271 Restart OFR Stream Processor Module Entry (continued)

Offset	Type	Length	Description
0	Fixed	4	OFR Id
4	Fixed	4	OFRL Number Entries
8	Address	4	OFRL Fast Path Area (1)

Example:

```
                              OFRL # of entities
                         OFRID      |          FOFR
                         |          |          |
DT271 Entry     9E13705E  00000004  00000001  C6D6C6D9
                00000000  00000000  00000000  5780F307
```

9E14

Table 88. Trace Record 9E14 - DBFDT272 End OFR Stream Notification Processor Module Entry

Module: DBFDT272 End OFR Stream Notification Processor Module Entry

Explanation: Record cut at entry to DBFDT272 (Level - High)

Trace Subcode DT272 Entry

Offset	Type	Length	Description
0	Fixed	4	OFR Id
4	Fixed	4	Milestone Index
8	Character	8	Area Name
16	Bit	4	OFR flags

9E15

Table 89. Trace Record 9E15 - DBFDT272 NoTUR Module Entry

Module: DBFDT272 NoTUR Module Entry

Explanation: Record cut in DBFDT272 when no storage for TUR in data space (Level - High)

Trace Subcode DT272 NoTUR

Offset	Type	Length	Description
0	Fixed	4	OFR Id
4	Fixed	4	Milestone Index
8	Character	8	Area Name
16	Address	4	Address of Top ERQE

9E16

Table 90. Trace Record 9E16 - DBFDT272 NoERQE Module Entry

Module: DBFDT272 NoERQE Module Entry

Explanation: Record cut in DBFDT272 when no storage for ERQE in data space (Level - High)

Trace Subcode DT272 NoERQE

Offset	Type	Length	Description

Table 90. Trace Record 9E16 - DBFDT272 NoERQE Module Entry (continued)

0	Fixed	4	OFR Id
4	Fixed	4	Milestone Index
8	Character	8	Area Name
16	Address	4	Address of EMAC

9E21

Table 91. Trace Record 9E21 - DBFDT250 Fast Path/Log Router TCB AWE Queue Server Module Entry

Module: DBFDT250 Fast Path/Log Router TCB AWE Queue Server Module Entry

Explanation: Record cut at entry to DBFDT250 (Level - High)

Trace Subcode DT250 Entry

Offset	Type	Length	Description
0	Address	4	AWE Enqueuer
4	Fixed	4	AWE Function Code
8	Character	16	AWE Contents

Example:

```
                          enqueuer  function  EMAC
                          |         |         |
DT250 Entry     9E21E5EC  84E01A5A  00000016  041843C0
                00000000  04DB98B0  84CD28F0  B4970B08
```

9E31

Table 92. Trace Record 9E31 - DBFDT263 End Active Stream ERQE Processor Module Entry

Module: DBFDT263 End Active Stream ERQE Processor Module Entry

Explanation: Record cut at entry to DBFDT263 (Level - High)

Trace Subcode DT263 Entry

Offset	Type	Length	Description
0	Address	4	Address EMAC
4	Fixed	4	Milestone Index
8	Fixed	1	Stream Id

9E33

Table 93. Trace Record 9E33 - DBFDT273 End Active Stream ERQE Processor Module Entry

Module: DBFDT273 End Active Stream ERQE Processor Module Entry

Explanation: Record cut at entry to DBFDT273 (Level - High)

Trace Subcode DT273 Entry

Offset	Type	Length	Description
0	Address	4	Address EMAC

9E34

Table 94. Trace Record 9E34 - DBFDT261 Tracking of End Update ERQE

Module: DBFDT261 Tracking of End Update ERQE

Explanation: Record cut at entry to DBFDT261 (Level - High)

Trace Subcode DT261 Entry

Offset	Type	Length	Description
0	Address	4	Address EMAC
4	Fixed	4	Milestone Index
8	Fixed	4	Stream Id

9E41

Table 95. Trace Record 9E41 - DBFDT180 Area Status Change Module Entry

Module: DBFDT180 Area Status Change Module Entry

Explanation: Record cut at entry to DBFDT180 (Level - High)

Trace Subcode DT180 Entry

Offset	Type	Length	Description
0	Fixed	4	Function Code
4	Fixed	4	Reason Code
8	Address	4	Address EMAC
12	Address	4	Address PST

9E42

Table 96. Trace Record 9E42 - DBFDT251 Request Area Auth/Open Module Entry

Module: DBFDT251 Request Area Auth/Open Module Entry

Explanation: Record cut at entry to DBFDT251 (Level - High)

Trace Subcode DT251 Entry

Offset	Type	Length	Description
0	Address	4	Address EMAC

9E43

Table 97. Trace Record 9E43 - DBFDT252 TUR Cleanup During Shutdown Module Entry

Module: DBFDT252 TUR Cleanup During Shutdown Module Entry

Explanation: Record cut at entry to DBFDT252 (Level - High)

Trace Subcode DT252 Entry

Offset	Type	Length	Description
0	Address	4	Address EMAC

9E60

Table 98. Trace Record 9E60 - DBFDT291 Prepare Milestone Module Entry

Module: DBFDT291 Prepare Milestone Module Entry

Explanation: Record cut at entry to DBFDT291 (Level - High)

Trace Subcode DT291 Entry

Offset	Type	Length	Description
0	Fixed	4	ECB

Example:

```
                          ECB
                          |
DT291 Entry     9E60A7EB  40C9D8F1  00000000  00000000
                00000000  00000000  00000000  74939BC5
```

9E61

Table 99. Trace Record 9E61 - DBFDT291 Prepare Milestone Module Exit

Module: DBFDT291 Prepare Milestone Module Exit

Explanation: Record cut at exit from DBFDT291 (Level - High)

Trace Subcode DT291 Exit

Offset	Type	Length	Description
0	Fixed	4	ECB

9E62

Table 100. Trace Record 9E62 - DBFDT291 Prepare Milestone IWAIT Issued

Module: DBFDT291 Prepare Milestone IWAIT Issued

Explanation: Record cut prior to issuing IWAIT in DBFDT291 (Level - High)

Trace Subcode DT291 IWAIT Issued

Offset	Type	Length	Description
0	Address	4	EDBTF

9E63

Table 101. Trace Record 9E63 - DBFDT291 Prepare Milestone IPOST Received

Module: DBFDT291 Prepare Milestone IPOST Received

Explanation: Record after IPOST Received in DBFDT291 (Level - High)

Trace Subcode DT291 IPOST Received

Offset	Type	Length	Description
0	Address	4	EDBTF

9E64

Table 102. Trace Record 9E64 - DBFDT292 Begin Milestone Module Entry

Module: DBFDT292 Begin Milestone Module Entry

Table 102. Trace Record 9E64 - DBFDT292 Begin Milestone Module Entry (continued)

Explanation: Record cut at entry to DBFDT292 (Level - High)

Trace Subcode DT292 Entry

Offset	Type	Length	Description
0	Fixed	4	Parameter type

9E65

Table 103. Trace Record 9E65 - DBFDT292 Begin Milestone Module Exit

Module: DBFDT292 Begin Milestone Module Exit

Explanation: Record cut at exit from DBFDT292 (Level - High)

Trace Subcode DT292 Exit

Offset	Type	Length	Description
0	Fixed	4	Parameter type
4	Fixed	4	Milestone Index Feedback

9E66

Table 104. Trace Record 9E66 - DBFDT292 Begin Milestone IPOST Received

Module: DBFDT292 Begin Milestone IPOST Received

Explanation: Record after IPOST Received in DBFDT292 (Level - High)

Trace Subcode DT292 IPOST Received

Offset	Type	Length	Description
0	Fixed	4	EDB IOTI Count

9E67

Table 105. Trace Record 9E67 - DBFDT290 Milestone Routine Entry

Module: DBFDT290 Milestone Routine Entry

Explanation: Record after Entry to DBFDT290 (Level - High)

Trace Subcode DT290 Entry

Offset	Type	Length	Description
0	Fixed	4	Function Code
4	Fixed	4	Parameter type
8	Fixed	4	Latest Milestone Index

9E68

Table 106. Trace Record 9E68 - DBFDT290 Milestone Routine Exit

Module: DBFDT290 Milestone Routine Exit

Explanation: Record after Exit from DBFDT290 (Level - High)

Trace Subcode DT290 Exit

Offset	Type	Length	Description

Table 106. Trace Record 9E68 - DBFDT290 Milestone Routine Exit (continued)

0	Fixed	4	Function Code
4	Fixed	4	Parameter type
8	Fixed	4	Feedback Milestone Index

9E69

Table 107. Trace Record 9E69 - DBFDT290 Milestone Enqueue 1

Module: DBFDT290 Milestone Enqueue 1

Explanation: Record at AWE Enqueue 1 in DBFDT290 (Level - High)

Trace Subcode DT290 Enqueue 1

Offset	Type	Length	Description
0	Address	4	Enqueuer's Address
4	Fixed	4	Function Code
8	Character	16	AWE Contents

9E6A

Table 108. Trace Record 9E6A - DBFDT290 Milestone Enqueue 2

Module: DBFDT290 Milestone Enqueue 2

Explanation: Record at AWE Enqueue 2 in DBFDT290 (Level - High)

Trace Subcode DT290 Enqueue 2

Offset	Type	Length	Description
0	Address	4	Enqueuer's Address
4	Fixed	4	Function Code
8	Fixed	16	AWE Contents

Trace Entry: Fast Path Tracker Log Router Interface (9F)

9F22

Table 109. Trace Record 9F22 - DBFDT300 Fast Path/Fast Path TCB AWE Queue Server Module Entry

Module: DBFDT300 Fast Path/Fast Path TCB AWE Queue Server Module Entry

Explanation: Record cut at entry to DBFDT300 (Level - High)

Trace Subcode DT300 Entry

Offset	Type	Length	Description
0	Address	4	AWE Enqueuer
4	Character	4	AWE Function Code
8	Character	16	AWE Contents

Example:

```
                        enqueuer  open area EMAC
                        |         |         |
DT300 Entry     9F22B879  04F9E5E2  00000003  0476A3C0
                00000001  00000002  00000000  A0AFD862
                |         |
                streamID  USID
```

9F41

Table 110. Trace Record 9F41 - DBFDT180 Area Status Change Module Entry

Module: DBFDT180 Area Status Change Module Entry

Explanation: Record cut at entry to DBFDT180 (Level - High)

Trace Subcode DT180 Entry

Offset	Type	Length	Description
0	Fixed	4	Function Code
4	Fixed	4	Reason Code
8	Address	4	Address EMAC
12	Address	4	Address PST

Example:

```
                                  ndtrk_fail
                        stop_req  |         EMAC
                        |         |         |
DT180 Entry     9F41D6C0  00000001  00000007  041843C0
                00B3C060  00000000  00000000  AC97BB2C
                |
                PST
```

9F44

Table 111. Trace Record 9F44 - DBFROFR0 OFR Module Entry

Module: DBFROFR0 OFR Module Entry

Explanation: Record cut at entry to DBFROFR0 (Level - High)

Trace Subcode ROFR0 Entry

Offset	Type	Length	Description
0	Fixed	4	Function Code
4	Fixed	4	Area Count

Table 112. Trace Record 9F44 - DBFROFR0 OFR Module Entry

Module: DBFROFR0 OFR Module Entry

Explanation: Record cut at entry to DBFROFR0 (Level - High)

Trace Subcode ROFR0 Entry

Offset	Type	Length	Description
0	Fixed	4	Function Code
4	Address	4	Address of DMAC

Table 113. Trace Record 9F44 - DBFROFR0 OFR Module Entry

Module: DBFROFR0 OFR Module Entry

Explanation: Record cut at entry to DBFROFR0 (Level - High)

Trace Subcode ROFR0 Entry

Offset	Type	Length	Description
0	Fixed	4	Function Code

9F50

Table 114. Trace Record 9F50 - DBFDT350 IPOST

Module: DBFDT350 IPOST

Explanation: Record cut at IPOST in DBFDT350 (Level - High)

Trace Subcode DT350 IPOST

Offset	Type	Length	Description
0	Character	4	Post Code
4	Address	4	EDBTWAQ

Example:

```
                                     EDBTWAQ
                          post code |
                          |          |
DT350 IPOSTed   9F508A97  40C6F2F2  02E85E40  00000000
                00000000  00000000  00000000  8613ED2D
```

9F51

Table 115. Trace Record 9F51 - DBFDT350 IWAIT

Module: DBFDT350 IWAIT

Explanation: Record cut at IWAIT in DBFDT350 (Level - High)

Trace Subcode DT350 IWAIT

Offset	Type	Length	Description
0	Character	4	Post Code
4	Address	4	EDBTWAQ Contents

Example:

```
                                     EDBTWAQ
                                    |
DT350 IWAIT     9F512DA3  00000000  846761EC  00000000
                00000000  00000000  00000000  32DC4B1C
```

9F52

Table 116. Trace Record 9F52 - DBFDT350 GETEMAC

Module: DBFDT350 GETEMAC

Explanation: Record cut at EMAC in DBFDT350 (Level - High)

Trace Subcode DT350 EMAC

Table 116. Trace Record 9F52 - DBFDT350 GETEMAC (continued)

Offset	Type	Length	Description
0	Address	4	Address EMAC
4	Address	4	EMACEMAC WAQ
8	Address	4	EMACERQE WAQ
12	Address	4	EMACERQE WIOQ
16	Fixed	4	EMACERQE WIOQ Count

Example:

```
                                    EMACEMAC_WAQ
                          EMAC      |         EMACERQE_WAQ
                          |         |         |
DT350 GETEMAC   9F528A98  02E85E40  0329B1E4  0000A740
                00006000  0000000E  00000000  8613ED6C
                |         |
                |         EMACERQE_WIOQ count
                EMACERQE_WIOQ
```

9F53

Table 117. Trace Record 9F53 - DBFDT350 GETERQE

Offset	Type	Length	Description
Module: DBFDT350 GETERQE			
Explanation: Record cut at ERQE in DBFDT350 (Level - High)			
Trace Subcode DT350 ERQE			
Offset	Type	Length	Description
0	Address	4	Address ERQE
4	Address	4	ERQEEMAC
8	Character	1	ERQEtype
9	Character	1	ERQEEF
20	Fixed	4	ERQEMILE Index

Example:

```
                                              type
                          ERQE      ERQEEMAC  | flags
                          |         |         | |
DT350 GETERQE   9F538A9A  0000B180  02E85E40  01080000
                00000000  00000000  00000090  8613EDE9
                                    |
                                    milestone index
```

9F54

Table 118. Trace Record 9F54 - DBFDT350 EMAC2

Offset	Type	Length	Description
Module: DBFDT350 EMAC2			
Explanation: Record cut at EMAC in DBFDT350 (Level - High)			
Trace Subcode DT350 EMAC2			
Offset	Type	Length	Description

Table 118. Trace Record 9F54 - DBFDT350 EMAC2 (continued)

0	Character	8	Area name

Example:

```
                          Area name
                          |
DT350 EMAC2     9F548A99  C4C4F0F1  C1D9F040  00000000
                00000000  00000000  00000000  8613ED9D
```

9F55

Table 119. Trace Record 9F55 - DBFDT350 ERQE2

Module: DBFDT350 ERQE2

Explanation: Record cut at ERQE in DBFDT350 (Level - High)

Trace Subcode DT350 ERQE2

Offset	Type	Length	Description
0	Character	8	Log Record ID

Example:

```
                          Log Record ID
                          |
DT350 ERQE2     9F558AA5  00000000  000023B3  00000000
                00000000  00000000  00000000  8613F10D
```

9F70

Table 120. Trace Record 9F70 - DBFDT400 IPOST

Module: DBFDT400 IPOST

Explanation: Record cut at IPOST in DBFDT400 (Level - High)

Trace Subcode DT400 IPOST

Offset	Type	Length	Description
0	Address	4	Address IOTI
4	Character	4	Post Code

9F71

Table 121. Trace Record 9F71 - DBFDT400 IWAIT

Module: DBFDT400 IWAIT

Explanation: Record cut at IWAIT in DBFDT400 (Level - High)

Trace Subcode DT400 IWAIT

Offset	Type	Length	Description
0	Address	4	Address IOTI

9F72

Table 122. Trace Record 9F72 - DBFDT400 EMAC

Module: DBFDT400 EMAC

Table 122. Trace Record 9F72 - DBFDT400 EMAC (continued)

Explanation: Record cut for EMAC in DBFDT400 (Level - High)

Trace Subcode DT400 EMAC

Offset	Type	Length	Description
0	Address	4	Address IOTI
4	Address	4	Address EMAC
8	Fixed	4	EDBT Milestone IOTI Done

9F73

Table 123. Trace Record 9F73 - DBFDT400 Read

Module: DBFDT400 Read

Explanation: Record cut at Read in DBFDT400 (Level - High)

Trace Subcode DT400 Read

Offset	Type	Length	Description
0	Address	4	Address IOTI
4	Address	4	Address DMHR
8	Fixed	4	DMHRSRBA
12	Address	4	DMHRDMAC
16	Fixed	4	IOTIERQE Count
20	Address	4	IOTIEMAC

Example:

```
                        IOTI      DMHR      DMHRSRBA
                        |         |         |
DT400 Read    9F7374E5  02F553A0  0316A860  00014000
              028F7E28  00000002  02867040  8FA4571B
              |         |         |
              DMAC      |         EMAC
                        IOTIERQE count
```

9F74

Table 124. Trace Record 9F74 - DBFDT400 Write

Module: DBFDT400 Write

Explanation: Record cut at Write in DBFDT400 (Level - High)

Trace Subcode DT400 Write

Offset	Type	Length	Description
0	Address	4	Address IOTI
4	Address	4	Address DMHR
8	Fixed	4	DMHRSRBA
12	Address	4	DMHRDMAC
16	Address	4	IOTIEMAC

Example:

```
                IOTI      DMHR      DMHRSRBA
                |         |         |
DT400 Write     9F7474DC  02F553A0  0316AD38  00013000
                028F7E28  02867040  00000000  8FA06CE5
                |         |
                DMAC      EMAC
```

Database Tracker Trace Entries

The DL/I database tracker trace entries are included in the DL/I Trace Table (dldnote—insert cross-reference to Dbtrace). The second byte (xx) of each trace entry is the PST number.

Table 125. DB TRACKING TRACE ENTRIES

W2(Code)	W2(Scode)	W3	W4	W5	W6	W7	W8
Note: W1 always has: Function (1 byte), Tracking PST number (1 byte, where appropriate), and Trace sequence number (2 bytes).							
1 DRQE q-drwq	stream id	TDBC	DRQE	DRWQ			
2 DRQE q-tdbc	stream id	TDBC	DRQE	DRWQ			
3 DRQE freed w/o track	stream id	TDBC	DRQE				
4 DBTI	1 dispatched for work	PST	DTT	DTTPCTL content			
5 DT240 AWE	AWE function	TDBC	AWE				
Note: Function (hex) 15=open ok. 16=DB stop request. 17=stream does not apply. 18=process tdbc queue. 1A =online change add ddir.							
6 DT300 AWE	AWE function	TDBC	AWE	return code			
Note: Function 1=initialization. 2=termination. 3=open | nusid DB. 4=close DB. 5=dspndtrk hardenend. 6=dspndtrk. 7=stream complete. 8=stream complete hardened. 9=do load balancing. 10=ofr needed. 11=ofr complete ndofr. 16=ndofr hardened.							
7 shut down	x'30' DT300						
	x'40' DT400	PST	DTT				
	x'50' DT500	PST					
8 DB stop	function	reason	TDBC	ecb			
Note: Function 0=initiate. 1=stopped. 2=may need ofr. 3=log tdbc state. 4=dfslretr. 5=started.							
9 milestone	0 ok begin?	type code	new index				
	1 begin request	type code	new index				
	2 end request	type code					
	3 xfer done	PST	PSTfnctn	index xferd			
	4 purg done	PST	PSTfnctn	index purged			
A: end stream	stream type	stream id	milestone index				
B: load balance	0 DTT stats	DTT	busy%	DTTwork	DTTwait	DTTpctl	
	1 summary	avg busy%	active DBTIs	backlog			
	2 DRWQ assign	new DTT	DRWQ	Q busy%	old DTT		
	3 DRWQ assigns complete	old DTT					

Table 125. DB TRACKING TRACE ENTRIES (continued)

W2(Code)	W2(Scode)	W3	W4	W5	W6	W7	W8
C: OFR	0 lrofr issued	OFR id	OFRL	DB count			
	1 restart ofr	OFR id	TDBC	TDBCT	flags		
	2 begin ofr	OFR id	TDBC	TDBCT	flags		
	3 end ofr	OFR id	TDBC	TDBCT	flags		
	4 begin ofr ignored	OFR id	TDBC	TDBCT	flags		
	5 restart ofr ignored	OFR id	TDBC	TDBCT	flags		
Note: Follows is the format of the DL/I buffer handler trace entry after being modified for DB tracking. * = changed							
PSTdmbnm	dcbnm rtcde	trmid msc bhflg subcd	rba of data in ci/blk *	PSTdata	PSTbuffa	log seq num *	PSTbytnm

Log Router Trace Data

The log router (LRTT) trace entries are documented below. Field lengths are in bytes.

Trace Entry: Log Router Data Set Services (370x)

3701

Table 126. Trace Record 3701 - Data Set Services Control Routine Entry

Module: DFSLRDSS Data Set Services Control ITASK Routine

Explanation: Record cut at entry to DFSLRDSS (Level - Low)

Trace Subcode LRDSS Entry

Offset	Type	Length	Description
4	Fixed	1	AWLGFUNC (AWE Function)
5	Fixed	1	AWLGDSFL (DSS Request Code)
6	Fixed	1	AWLGDSTP (Data Set Type)
	1... .... .1.. .... .1. ....1 1111	1	Tracking_SLDS (AWLGDTRK) Archive SLDS (AWLGDARC) Archive RLDS (AWLGDRLD)
7	Fixed	1	Request Priority (AWLGDPRI)
8	Address	4	LTDCB address (AWLGDLTD)
12	Address	4	LDSD address (AWLGDLDS)
16	Bit	4	DSS Flags (LGBDSSFLAGS)
	1... .... .1.. .... ..1. ....1 .... 1... .... .1.. .... ..1. ....1 1... .... .1.. .... ..11 1111	1	LGB_CBTE_ALTERED LGBDSS_DUAL_TRACKING_SLDS LGBDSS_DUAL_ARCHIVE_SLDS LGBDSS_DUAL_ARCHIVE_RLDS LGB_ARCHIVE_SLDS LGB_ARCHIVE_RLDS LGB_INITIALIZEDSS LGB_TERMINATINGDSS LGB_DSS_DATASETS_RETURNED LGB_DSS_RESTART_INIT *
20	Fixed	4	LGB_DATASET_NUMBER
24	Bit	2	Data set Action Flags (AWLGDSAC)

Table 126. Trace Record 3701 - Data Set Services Control Routine Entry (continued)

	1...		Delete data set (AWLGDSDE)
	.1..		Input/Output (AWLGDSIO)
	..1.		Last active data set (AWLGDLST)
	...1		Allocate for restart (AWLGDARS)
	.... 1...		4906 delete record (AWLGD4906)
	.... .1..		Delete for restart (AWLGDRST)
	.... ..1.		End stream notification (AWLGDEST)
	1		Create prealloc data set (AWLGDLGB)
25	Bit	2	LTDCB_FLAGS
	1...		LTDCB_DBRC_OPEN
	.1..		LTDCB_DBRC_CLOSED
	..1.		LTDCB_LAST_BUFFER_WRITTEN
	...1		LTDCB_EODAD
	.... 1...		LTDCB_DELETE_DATASET
	.... .1..		LTDCB_OPEN_ERROR_1
	.... ..1.		LTDCB_OPEN_ERROR_2
	1		LTDCB_MOUNTABLE

3702

Table 127. Trace Record 3702 - Create Data Set Routine Invoke DYA

Module: DFSLRDCR Data Set Create Routine

Explanation: Invoke DYA from DFSLRDCR (Level - Medium)

Trace Subcode LRDCR Create

Offset	Type	Length	Description
4	Char	8	DD Name (LTDCB_DDNAME)
12	Char	8	DS Type (from DS Name)
20	Char	8	DS Name (LTDCB_DSN)
28	Address	4	LDSD address (AWLGDLDS)

3703

Table 128. Trace Record 3703 - Create Data Set Routine Exit

Module: DFSLRDCR Data Set Create Routine

Explanation: Record cut at exit from DFSLRDCR (Level - Medium)

Trace Subcode LRDCR Exit

Offset	Type	Length	Description
4	Fixed	1	AWLGFUNC (AWE Function)
5	Fixed	1	AWLGDSFL (DSS Request Code)
6	Fixed	1	AWLGDSTP (Data Set Type)
	1...		Tracking_SLDS (AWLGDTRK)
	.1..		Archive SLDS (AWLGDARC)
	..1.		Archive RLDS (AWLGDRLD)
	...1 1111		
7	Fixed	1	Request Priority (AWLGDPRI)
8	Fixed	4	Return Code
12	Fixed	2	Return Code from Data Set One
14	Fixed	2	Reason Code from Data Set One
16	Fixed	2	Return Code from Data Set Two
18	Fixed	2	Reason Code from Data Set Two
20	Address	4	LTDCB address (AWLGDLTD)
24	Address	4	LDSD address (AWLGDLDS)

3704

Table 129. Trace Record 3704 - Allocate Data Set Routine Exit

Module: DFSLRDAL Data Set Allocate Routine

Explanation: Record cut at exit from DFSLRDAL (Level - Medium)

Trace Subcode LRDAL Exit

Offset	Type	Length	Description
4	Fixed	4	Return Code
8	Fixed	2	Return Code from Data Set One

Table 129. Trace Record 3704 - Allocate Data Set Routine Exit (continued)

10	Fixed	2	Reason Code from Data Set One
12	Fixed	2	Return Code from Data Set Two
14	Fixed	2	Reason Code from Data Set Two
16	Address	4	LTDCB Address (AWLGDLTD)
20	Address	4	LDSD address (AWLGDLDS)
24	Address	4	R13

3705

Table 130. Trace Record 3705 - Open Data Set Routine Exit

Module: DFSLRDOP Data Set Open Routine

Explanation: Record cut at exit from DFSLRDOP (Level - Medium)

Trace Subcode LRDOP Exit

Offset	Type	Length	Description
4	Fixed	1	AWLGDSFL (DSS Request Code)
5	Bit	1	Data set Action Flags (AWLGDSAC)
	1...		Delete data set (AWLGDSDE)
	.1..		Input/Output (AWLGDSIO)
	..1.		Last active data set (AWLGDLST)
	...1		Allocate for restart (AWLGDARS)
	.... 1...		4906 delete record (AWLGD4906)
	.... .1..		Delete for restart (AWLGDRST)
	.... ..1.		End stream notification (AWLGDEST)
	1		Create prealloc data set (AWLGDLGB)
6	Fixed	2	Reason Code from Open Routine
8	Fixed	2	Return Code from Open Macro for Data Set One
10	Fixed	2	Return Code from Open Macro for Data Set Two
12	Address	3	LTDCB address (AWLGDLTD)
16	Address	4	LDSD address (AWLGDLDS)
20	Address	4	AWE address
24	Address	4	R13

3707

Table 131. Trace Record 3707 - Deallocate/Delete Data Set Routine Exit

Module: DFSLRDDE Data Set Deallocate/Delete Routine

Explanation: Record cut at exit from DFSLRDDE (Level - Medium)

Trace Subcode LRDDE Exit

Offset	Type	Length	Description
4	Fixed	1	AWLGFUNC (AWE Function
5	Fixed	1	AWLGDSFL (DSS Request Code)
6	Fixed	1	AWLGDSTP (Data Set Type)
	1...		Tracking_SLDS (AWLGDTRK)
	.1..		Archive SLDS (AWLGDARC)
	..1.		Archive RLDS (AWLGDRLD)
	...1 1111		
7	Fixed	1	Request Priority (AWLGDPRI)
8	Address	4	LTDCB address (AWLGDLTD)
12	Address	4	LDSD address (AWLGDLDS)
16	Fixed	2	Return Code from Data Set One
18	Fixed	2	Reason Code from Data Set One
20	Fixed	2	Return Code from Data Set Two
22	Fixed	2	Reason Code from Data Set Two
24	Bit	1	Data set Action Flags (AWLGDSAC)
	1...		Delete data set (AWLGDSDE)
	.1..		Input/Output (AWLGDSIO)
	..1.		Last active data set (AWLGDLST)
	...1		Allocate for restart (AWLGDARS)
	.... 1...		4906 delete record (AWLGD4906)
	.... .1..		Delete for restart (AWLGDRST)
	.... ..1.		End stream notification (AWLGDEST)
	1		Create prealloc data set (AWLGDLGB)
25	Bit	2	LTDCB_flags

Table 131. Trace Record 3707 - Deallocate/Delete Data Set Routine Exit (continued)

	1...	LTDCB_DBRC_OPEN
	.1..	LTDCB_DBRC_CLOSED
	..1.	LTDCB_LAST_BUFFER_WRITTEN
	...1	LTDCB_EODAD
	.... 1...	LTDCB_DELETE_DATASET
	.... .1..	LTDCB_OPEN_ERROR_1
	.... ..1.	LTDCB_OPEN_ERROR_2
	1	LTDCB_MOUNTABLE

Trace Entry: Log Router Record Router (3709/370E/370F/371x)

3709

Table 132. Trace Record 3709 - End of Merge

Module: DFSLRMRG Log Router Log Merge

Explanation: Record is cut when a stream is removed from a merge (Level - Low)

Trace Subcode LRMRG End Mrg

Offset	Type	Length	Description
4	Char	8	Stream subsystem ID
12	Char	1	mrb_status
13	Char	1	Spare
14	Fixed	2	Number of remaining merge blocks
16	Fixed	4	Stream ID
20	Char	4	stb_last_routed_LSN(5-8)

Trace Entry: Log Router Record Router (370E/370F/371x)

370E

Table 133. Trace Record 370E - Received last buffer of the active stream

Module: DFSLRRR0 Log Record Router

Explanation: Record cut at End Buffer (Level - Low)

Trace Subcode LRRR0 End Strm

Offset	Type	Length	Description
4	Fixed	4	stb_routing_prilog_token
8	Char	8	stb_last_routed_LSN
2	Bit	16	stb_flags
	1...		STB_DATASHARING
	.1..		STB_TERMINATED
	..1.		STB_BATCH
	...1		STB_OFR_CACHING
	.... 1...		STB_TERMINATING
	.... .1..		STB_CONV_WITH_LOGGER
	.... ..1.		STB_ACTIVE_ABENDED
	1		STB_SHUTDOWN_IN_PROGRESS
	1...		STB_RESTARTING
	.1..		STB_READ_IN_PROGRESS
	..1.		STB_READ_ERROR
	...1		STB_ROUTING_SUSPENDED
	.... 1...		STB_END_OF_STREAM
	.... .1..		STB_UNABLE_TO_ROUTE
	.... ..1.		STB_SHUTDOWN_REQUESTED
	1		STB_SHUTDOWN_COMPLETE
18	Bit	2	LRB_BUFFER_flags
	1...		LRB_BUFFER_DS_FULL
	1...		LRB_BUFFER_EODAD
	.1..		STB_BUFFER_IO_ABEND
	..11 1111		*

Table 133. Trace Record 370E - Received last buffer of the active stream (continued)

	1...		LRB_READ_COMPLETE
	.1..		LRB_BUFFER_LAST
	..1.		LRB_BUFFER_ENDDS
	...1		LRB_BUFFER_RESTART
	.... 11..		LRB_BUFFER_ORIGIN
	00		LRB_FROM_LOGGER
	01		LRB_FROM_ILS
	10		LRB_FROM_READER
	11		LRB_FROM_ARCH
	.... ..1.		LRB_ACTIVE_ABEND
	1		LRB_BEGIN_OFR_CACHING
20	Fixed	4	stb_streamID
24	Char	4	stb_routing_prilog_token
28	Fixed	4	stb_last_routed_lsn(5-8)

370F

Table 134. Trace Record 370F - Routed Log Records from Buffer to Trackers

Module: DFSLRRBF Route Buffer Routine

Explanation: Record cut at exit from DFSLRRBF (Level - High)

Trace Subcode LRRBF Route

Offset	Type	Length	Description
4	Char	4	lrb_record_id(5-8)
8	Char	4	First routed LSN
12	Char	4	Last routed LSN
16	Fixed	4	offset to first LSN routed
20	Fixed	4	lpd_stream_type
24	Fixed	4	lpd_stream_id
28	Address	4	R13 value

3710

Table 135. Trace Record 3710 - Active Stream Tracker RSR04_PTKO

Module: DFSLRAST Active Stream Tracker Routine

Explanation: Record cut at received 0401 log (Level - Low)

Trace Subcode LRAST PTKO Req

Offset	Type	Length	Description
4	Char	1	rsr04code
5	Char	1	rsr04sub
7	Char	1	lpd_flags
	1...		stream is being merged
8	Char	4	lpd_feedback
12	Char	4	lrb_record_ID(5-8)
16	Char	8	r04_stck
24	Fixed	4	lpd_stream_id

3712

Table 136. Trace Record 3712 - Active Stream Tracker RSR04SUB

Module: DFSLRAST Active Stream Tracker Routine

Explanation: Record cut at received 0402 through 0407 log (Level - Low)

Trace Subcode LRAST DataShr

Offset	Type	Length	Description
4	Char	1	rsr04code

Table 136. Trace Record 3712 - Active Stream Tracker RSR04SUB (continued)

5	Char	1	rsr04sub
8	Char	4	r04_hipritoken
12	Char	4	lrb_record_ID(5-8)
16	Char	8	r04_prilgts(1-8)
24	Fixed	31	lpd_stream_id

Trace Entry: Log Router I/O (373x)

3731

Table 137. Trace Record 3731 - Stream Archiver Controller Entry

Module: DFSLRSAR Stream Archiver Controller ITASK Routine

Explanation: Record cut on entry to DFSLRSAR for all requests except for write (awlgfwrt) and return buffer from reader during truncation (awlgfrtb) (Level - High)

Trace Subcode LRSAR Entry

Offset	Type	Length	Description
4	Address	4	SAA® Address
8	Bit	4	SAA_flags
	1...		SAA_NEW_STREAM
	.1..		SAA_LAST_BUFFER_WRITTEN
	..1.		SAA_ARCHIVER_WAITING
	...1		SAA_DUAL_LOGGING
	.... 1...		SAA_SETUPFORARCHIVE
	.... .1..		SAA_CLOSE_FAILED (to DBRC)
	.... ..1.		SAA_SHUTDOWN
	1		SAA_IS_ACTIVE
	1...		SAA_WAIT_FOR_ALL_ITASKS
	.1..		SAA_BEGIN_OFR_CACHING
	..1.		SAA_WRITE_IN_PROGRESS
	...1		
	.... 1...		SAA_CREATEDITASKS
	.... .1..		SAA_NO_WRITE_DONE
	.... ..1.		*
	1		SAA_TERM_MSG_SENT
			*
	1...		SAA_BAD_BUFFER_DETECTED
	.1..		SAA_TERMINATING
	..1.		SAA_ERROR_DETECTED
	...1		SAA_EXIT_NO_BUFFER
	.... 1...		SAA_DO_NOT_ROUTE
	.... .1..		SAA_TRACKS_MATCH
	.... ..1.		SAA_HANDLE_IO_ERROR
	1		SAA_GAP_FILLED
	1...		SAA_COLDSTART
	.1..		SAA_NOBMP
	..1.		SAA_XRF_TAKEOVER
	...1		SAA_1ST_BFR_CK_INPROG
	.... 1...		SAA_1ST_BUFR_CK_OK
	.... .111		
12	Bit	2	AWLGFUNC
14	Bit	1	SAA_ITASK_CONTROL_flags
	.1..		SAA_DS_FULL
	.1..		*
	..1.		SAA_IO_ERROR_1
	...1		SAA_IO_ERROR_2
	 1111		*
15	Bit	1	SAA_DS_type
	.1..		SAA_TRACKING_SLDS
	.1..		SAA_ARCHIVE_SLDS
	..1.		SAA_ARCHIVE_RLDS
	...1 1111		*
16	Bit	2	SAA_NUM_ITASKS
18	Bit	2	SAA_LOG_COPIES
20	Bit	2	SAA_AVAIL_ITASK
22	Bit	2	SAA_OLDEST_BUSY_ITASK

Table 137. Trace Record 3731 - Stream Archiver Controller Entry (continued)

24	Character	8	SAA_PRILOG_TIME

3732

Table 138. Trace Record 3732 - Stream Archiver Controller Exit

Module: DFSLRSAR Stream Archiver Controller ITASK Routine

Explanation: Record cut on exit from DFSLRSAR (Level - Medium)

Trace Subcode LRSAR Exit

Offset	Type	Length	Description
4	Address	4	SAA Address
8	Bit	4	SAA_flags
	1...		SAA_NEW_STREAM
	.1..		SAA_LAST_BUFFER_WRITTEN
	..1.		SAA_ARCHIVER_WAITING
	...1		SAA_DUAL_LOGGING
	.... 1...		SAA_SETUPFORARCHIVE
	.... .1..		SAA_CLOSE_FAILED (to DBRC)
	.... ..1.		SAA_SHUTDOWN
	1		SAA_IS_ACTIVE
	1...		SAA_WAIT_FOR_ALL_ITASKS
	.1..		SAA_BEGIN_OFR_CACHING
	..1.		SAA_WRITE_IN_PROGRESS
	...1		SAA_CREATEDITASKS
	.... 1...		SAA_NO_WRITE_DONE
	.... .1..		*
	.... ..1.		SAA_TERM_MSG_SENT
	1		*
	1...		SAA_BAD_BUFFER_DETECTED
	.1..		SAA_TERMINATING
	..1.		SAA_ERROR_DETECTED
	...1		SAA_EXIT_NO_BUFFER
	.... 1...		SAA_DO_NOT_ROUTE
	.... .1..		SAA_TRACKS_MATCH
	.... ..1.		SAA_HANDLE_IO_ERROR
	1		SAA_GAP_FILLED
	1...		SAA_COLDSTART
	.1..		SAA_NOBMP
	..1.		SAA_XRF_TAKEOVER
	...1		SAA_1ST_BFR_CK_INPROG
	.... 1...		SAA_1ST_BUFR_CK_OK
	.... .111		
12	Bit	2	AWLGFUNC
14	Bit	2	SAA_ITASK_CONTROL_flags
	.1..		SAA_DS_FULL
	.1..		*
	..1.		SAA_IO_ERROR_1
	...1		SAA_IO_ERROR_2
	 1111		*
15	Bit	1	SAA_DS_type
	.1..		SAA_TRACKING_SLDS
	.1..		SAA_ARCHIVE_SLDS
	..1.		SAA_ARCHIVE_RLDS
	...1 1111		*
16	Fixed	4	Feedback Code
18	Bit	2	SAA_AVAIL_ITASK
20	Bit	2	SAA_OLDEST_BUSY_ITASK
24	Character	8	SAA_PRILOG_TIME

3733

Table 139. Trace Record 3733 - Stream Archiver WRITE Invocation

Module: DFSLRWRT Stream Archiver WRITE Routine

Explanation: Record cut just prior to invocation of the WRITE macro in DFSLRWRT (Level - High)

Trace Subcode LRWRT Write

Offset a	Type	Length	Description
4	Address	4	SAA Address

Table 139. Trace Record 3733 - Stream Archiver WRITE Invocation (continued)

8	Address	4	SAA_CURRENT_DATA_WRITTEN
12	Address	4	LTDCB_DCB_PTR(*)
16	Address	4	SAA_ITASK_BUFFER(*)
20	Fixed	4	LRB_BUFFER_HARD last 4 bytes of the last committed log sequence number
24	Fixed	4	LRB_RECORD_ID
28	Fixed	4	LRB_BUFFER_LLSN number in buffer being written (lower half word)

3734

Table 140. Trace Record 3734 - Stream Archiver Switch Data Set

Module: DFSLRSDS Stream Archiver Switch Data Set Routine

Explanation: Record cut just prior to switching data sets when a data set full or other error condition is recognized (Level - High)

Trace Subcode LRSDS Switch

Offset	Type	Length	Description
4	Address	4	SAA address
8	Bit	4	SAA_flags
	1...		SAA_NEW_STREAM
	.1..		SAA_LAST_BUFFER_WRITTEN
	..1.		SAA_ARCHIVER_WAITING
	...1		SAA_DUAL_LOGGING
	.... 1...		SAA_SETUPFORARCHIVE
	.... .1..		SAA_CLOSE_FAILED (to DBRC)
	.... ..1.		SAA_SHUTDOWN
	1		SAA_IS_ACTIVE
	1...		SAA_WAIT_FOR_ALL_ITASKS
	.1..		SAA_BEGIN_OFR_CACHING
	..1.		SAA_WRITE_IN_PROGRESS
	...1		SAA_CREATEDITASKS
	.... 1...		SAA_NO_WRITE_DONE
	.... .1..		*
	.... ..1.		SAA_TERM_MSG_SENT
	1		*
	1...		SAA_BAD_BUFFER_DETECTED
	.1..		SAA_TERMINATING
	..1.		SAA_ERROR_DETECTED
	...1		SAA_EXIT_NO_BUFFER
	.... 1...		SAA_DO_NOT_ROUTE
	.... .1..		SAA_TRACKS_MATCH
	.... ..1.		SAA_HANDLE_IO_ERROR
	1		SAA_GAP_FILLED
	1...		SAA_COLDSTART
	.1..		SAA_NOBMP
	..1.		SAA_XRF_TAKEOVER
	...1		SAA_1ST_BFR_CK_INPROG
	.... 1...		SAA_1ST_BUFR_CK_OK
	.... .111		
12	Address	4	SAA_LDSD
16	Address	4	SAA_LTDCB
20	Character	4	AWLG_CSW_LSN
24	Character	4	LRB_RECORD_ID
28	Fixed	4	Switch feedback

3736

Table 141. Trace Record 3736 - Stream Archiver Log Truncation Start Exit

Module: DFSLRLTS Log Truncation Start Routine

Explanation: Record cut at exit from DFSLRLTS (Level - Low)

Trace Subcode LRLTS Exit

Offset	Type	Length	Description
4	Char	8	SAA_TRUNC_LSN_POINT
12	Address	4	SAA Address
16	Address	4	SAA_LDSD
20	Character	8	SAA_PRILOG_TIME

Table 141. Trace Record 3736 - Stream Archiver Log Truncation Start Exit (continued)

28	Bit	2	SAA_TRUNC_flags
	1...		SAA_TRUNCATION
	.1..		SAA_TRUNC_READ_COMPLETE
	..1.		SAA_TRUNC_WRITE_COMPLETE
	...1		SAA_TRUNC_NO_DATASET
	.... 1...		SAA_TRUNC_RESTART_WRITE
	.... .1..		SAA_RETRY
	.... ..1.		SAA_PRIOR_RDR_ERR
	1		SAA_RETRY_SENT
	1...		SAA_TRUNC_NONE_DONE
	.111 1111		*
30	Fixed	16	SAA_TRUNC_STAGE

3737

Table 142. Trace Record 3737 - Log Router Log Truncation exit

Module: DFSLRLTR Log Truncation Routine

Explanation: Record cut at exit from DFSLRLTR (Level - Low)

Trace Subcode LRLTR Exit

Offset	Type	Length	Description
4	Address	4	SAA address
8	Bit	4	SAA_flags
	1...		SAA_NEW_STREAM
	.1..		SAA_LAST_BUFFER_WRITTEN
	..1.		SAA_ARCHIVER_WAITING
	...1		SAA_DUAL_LOGGING
	.... 1...		SAA_SETUPFORARCHIVE
	.... .1..		SAA_CLOSE_FAILED (to DBRC)
	.... ..1.		SAA_SHUTDOWN
	1		SAA_IS_ACTIVE
	1...		SAA_WAIT_FOR_ALL_ITASKS
	.1..		SAA_BEGIN_OFR_CACHING
	..1.		SAA_WRITE_IN_PROGRESS
	...1		SAA_CREATEDITASKS
	.... 1...		SAA_NO_WRITE_DONE
	.... .1..		*
	.... ..1.		SAA_TERM_MSG_SENT
	1		*
	1...		SAA_BAD_BUFFER_DETECTED
	.1..		SAA_TERMINATING
	..1.		SAA_ERROR_DETECTED
	...1		SAA_EXIT_NO_BUFFER
	.... 1...		SAA_DO_NOT_ROUTE
	.... .1..		SAA_TRACKS_MATCH
	.... ..1.		SAA_HANDLE_IO_ERROR
	1		SAA_GAP_FILLED
	1...		SAA_COLDSTART
	.1..		SAA_NOBMP
	..1.		SAA_XRF_TAKEOVER
	...1		SAA_1ST_BFR_CK_INPROG
	.... 1...		SAA_1ST_BUFR_CK_OK
	.... .111		
12	Bit	2	SAA_TRUNC_flags
	1...		SAA_TRUNCATION
	.1..		SAA_TRUNC_READ_COMPLETE
	..1.		SAA_TRUNC_WRITE_COMPLETE
	...1		SAA_TRUNC_NO_DATASET
	.... 1...		SAA_TRUNC_RESTART_WRITE
	.... .1..		SAA_RETRY
	.... ..1.		SAA_PRIOR_RDR_ERR
	1		SAA_RETRY_SENT
	1... .... .111 1111	1	SAA_TRUNC_NONE_DONE *
14	Fixed	2	SAA_TRUNC_ID
16	Bit	1	SAA_DS_flags

Table 142. Trace Record 3737 - Log Router Log Truncation exit (continued)

	.1..		SAA_TRACKING_SLDS
	.1..		SAA_ARCHIVE_SLDS
	..1.		SAA_ARCHIVE_RLDS
	...1 1111		*
18	Bit	2	SAA_ITASK_CONTROL_flags
	.1..		SAA_DS_FULL
	.1..		*
	..1.		SAA_IO_ERROR_1
	...1		SAA_IO_ERROR_2
	 1111		*
20	Address	4	SAA_LTDCB
24	Character	4	SAA_TRUNC_LSN_POINT

3738

Table 143. Trace Record 3738 - Log Router Log Read Controller exit

Module: DFSLRRDC Log Read Controller ITASK Routine

Explanation: Record cut on exit from DFSLRRDC (Level - Low)

Trace Subcode LRRDC Entry

Offset	Type	Length	Description
4	Fixed	1	AWLGFUNC
8	Address	4	LDSD (if func=CRD), GFR (if func=RCU), LRA (if func=TRD)
12	Address	4	LRB Buffer Chain Address or AWLG_TRD_RDR_TOKEN (if func=TRD)
16	Address	4	Requester Routine Address
20	Character	4	First LSN of read interval
24	Character	4	Last LSN of read interval
28	Address	4	AWEENQER

373A

Table 144. Trace Record 373A - Log Router Log Reader First Read Request

Module: DFSLRRDR Log Reader

Explanation: Record cut upon the initial entry to a log reader (Level - Low)

Trace Subcode LRRDR 1st Read

Offset	Type	Length	Description
4	Address	4	LRA Address
8	Bit	4	LRA_flags
	1...		LRA_LOGREADER_WAITING
	.1..		LRA_WAIT_FOR_ALL_ITASKS
	..1.		LRA_CURRENT_DATASET_ALLOCATED
	...1		LRA_READ_COMPLETE
	.... 1...		LRA_THROTTLE_ENABLED
	.... .1..		LRA_DEALLOCATE_ENABLED
	.... ..1.		LRA_HIT_EODAD
	1		LRA_ALLOC_DS_ERROR
	1...		LRA_RESTART
	.1..		LRA_CATCHUP_RDR
	..1.		LRA_SENT_DONE
	...1		LRA_READ_STARTED
	.... 1...		LRA_ONE_DATASET
	.... .1..		LRA_CURRENT_DUAL
	.... ..1.		LRA_ALLOCATED_SECOND
	1		LRA_EODADHANDLER_IN_PROGRESS
	1...		LRA_ALLOCATE_IN_PROGRESS
	.1..		LRA_TERM_CALLER
	..1.		LRA_CHECK_IPOST
	...1		LRA_IPOSTED_READER
	.... 1...		LRA_CLOSE_ONLY
	.... .1..		LRA_CLOSE_LAST
	.... ..1.		LRA_BIR_PROCESSING
	1		LRA_BUFFER_LAST

Table 144. Trace Record 373A - Log Router Log Reader First Read Request (continued)

	1...		LRA_CLOSE_PRIOR_DS
	.1..		LRA_AUTOARCH
	..1.		LRA_DO_NOT_IPOST
	...1 1111		
12	Address	4	LRA_LDSD_LIST
16	Address	4	LRA_LRB_PTR
20	Address	4	LRA_FIRST_LSN interval
24	Address	4	LRA_LAST_LSN
28	Address	4	Feedback Code

373B

Table 145. Trace Record 373B - Log Router Log Reader Buffer Return

Module: DFSLRBIR Log Reader BSAM Buffer ITASK

Explanation: Record cut when returning a buffer to requester (Level - Medium)

Trace Subcode LRBIR Ret Buf

Offset	Type	Length	Description
4	Address	4	LRA Address
8	Bit	4	LRA_flags
	1...		LRA_LOGREADER_WAITING
	.1..		LRA_WAIT_FOR_ALL_ITASKS
	..1.		LRA_CURRENT_DATASET_ALLOCATED
	...1		LRA_READ_COMPLETE
	.... 1...		LRA_THROTTLE_ENABLED
	.... .1..		LRA_DEALLOCATE_ENABLED
	.... ..1.		LRA_HIT_EODAD
	1		LRA_ALLOC_DS_ERROR
	1...		LRA_RESTART
	.1..		LRA_CATCHUP_RDR
	..1.		LRA_SENT_DONE
	...1		LRA_READ_STARTED
	.... 1...		LRA_ONE_DATASET
	.... .1..		LRA_CURRENT_DUAL
	.... ..1.		LRA_ALLOCATED_SECOND
	1		LRA_EODADHANDLER_IN_PROGRESS
	1...		LRA_ALLOCATE_IN_PROGRESS
	.1..		LRA_TERM_CALLER
	..1.		LRA_CHECK_IPOST
	...1		LRA_IPOSTED_READER
	.... 1...		LRA_CLOSE_ONLY
	.... .1..		LRA_CLOSE_LAST
	.... ..1.		LRA_BIR_PROCESSING
	1		LRA_BUFFER_LAST
	1...		LRA_CLOSE_PRIOR_DS
	.1..		LRA_AUTOARCH
	..1.		LRA_DO_NOT_IPOST
	...1 1111		
12	Fixed	4	LRA_USER_token
16	Address	4	LRB address
20	Fixed	2	ITASK index
22	Fixed	2	LRA_OLDEST_BUSY_ITASK
24	Character	4	LRB_RECORD_ID
28	Character	4	LRB_BUFFER_LLSN

373C

Table 146. Trace Record 373C - Log Router Log Reader Reread Data Set Request

Module: DFSLRRDR Log Read Controller ITASK Routine

Explanation: Record cut when an error occurred on first copy of a data set and an attempt is being made to read the dual copy (Level - Low)

Trace Subcode LRRDR ReRead

Offset	Type	Length	Description
4	Address	4	LRA Address
8	Bit	4	LRA_flags

Table 146. Trace Record 373C - Log Router Log Reader Reread Data Set Request (continued)

	1...		LRA_LOGREADER_WAITING
	.1..		LRA_WAIT_FOR_ALL_ITASKS
	..1.		LRA_CURRENT_DATASET_ALLOCATED
	...1		LRA_READ_COMPLETE
	.... 1...		LRA_THROTTLE_ENABLED
	.... .1..		LRA_DEALLOCATE_ENABLED
	.... ..1.		LRA_HIT_EODAD
	1		LRA_ALLOC_DS_ERROR
	1...		LRA_RESTART
	.1..		LRA_CATCHUP_RDR
	..1.		LRA_SENT_DONE
	...1		LRA_READ_STARTED
	.... 1...		LRA_ONE_DATASET
	.... .1..		LRA_CURRENT_DUAL
	.... ..1.		LRA_ALLOCATED_SECOND
	1		LRA_EODADHANDLER_IN_PROGRESS
	1...		LRA_ALLOCATE_IN_PROGRESS
	.1..		LRA_TERM_CALLER
	..1.		LRA_CHECK_IPOST
	...1		LRA_IPOSTED_READER
	.... 1...		LRA_CLOSE_ONLY
	.... .1..		LRA_CLOSE_LAST
	.... ..1.		LRA_BIR_PROCESSING
	1		LRA_BUFFER_LAST
	1...		LRA_CLOSE_PRIOR_DS
	.1..		LRA_AUTOARCH
	..1.		LRA_DO_NOT_IPOST
	...1 1111		
12	Fixed	4	LRA_REREAD_ITASK
16	Character	8	LRA_DS_LSN
20	Character	8	LRA_FIRST_LSN
24	Character	8	LRA_LAST_LSN
28	Address	4	Feedback Code

373D

Table 147. Trace Record 373D - Log Router Log Reader Exit

Module: DFSLRRDR Log Reader

Explanation: Record cut on exit from DFSLRRDR (Level - Low)

Trace Subcode LRRDR Exit

Offset	Type	Length	Description
4	Address	4	LRA Address
8	Bit	4	LRA flags
	1...		LRA_LOGREADER_WAITING
	.1..		LRA_WAIT_FOR_ALL_ITASKS
	..1.		LRA_CURRENT_DATASET_ALLOCATED
	...1		LRA_READ_COMPLETE
	.... 1...		LRA_THROTTLE_ENABLED
	.... .1..		LRA_DEALLOCATE_ENABLED
	.... ..1.		LRA_HIT_EODAD
	1		LRA_ALLOC_DS_ERROR
	1...		LRA_RESTART
	.1..		LRA_CATCHUP_RDR
	..1.		LRA_SENT_DONE
	...1		LRA_READ_STARTED
	.... 1...		LRA_ONE_DATASET
	.... .1..		LRA_CURRENT_DUAL
	.... ..1.		LRA_ALLOCATED_SECOND
	1		LRA_EODADHANDLER_IN_PROGRESS
	1...		LRA_ALLOCATE_IN_PROGRESS
	.1..		LRA_TERM_CALLER
	..1.		LRA_CHECK_IPOST
	...1		LRA_IPOSTED_READER
	.... 1...		LRA_CLOSE_ONLY
	.... .1..		LRA_CLOSE_LAST
	.... ..1.		LRA_BIR_PROCESSING
	1		LRA_BUFFER_LAST

Table 147. Trace Record 373D - Log Router Log Reader Exit (continued)

	1...		LRA_CLOSE_PRIOR_DS
	.1..		LRA_AUTOARCH
	..1.		LRA_DO_NOT_IPOST
	...1 1111		
12	Fixed	2	LRA_AVAIL_ITASK
14	Fixed	2	LRA_OLDEST_BUSY_ITASK
16	Address	4	LRA_GOOD_LSN
20	Address	4	LRA_FIRST_LSN interval
24	Address	4	LRA_LAST_LSN
28	Address	4	Feedback Code

373E

Table 148. Trace Record 373E - Log Router Start Log Reader Entry

Module: DFSLRRDS Start Log Reader

Explanation: Record cut on entry to DFSLRRDS (Level - Low)

Trace Subcode LRRDS Entry

Offset	Type	Length	Description
4	Address	1	AWE function Code
5	Fixed	3	Number of GDS
8	Address	4	LDSD or GDS address
12	Address	2	LRB chain address
16	Address	4	User's routine Address
20	Fixed	4	User's token interval
24	Char	4	First LSN (bytes 5:8)
28	Char	4	Last LSN (bytes 5:8)

Trace Entry: Log Router Create Active Stream Support (374x)

3740

Table 149. Trace Record 3740 - DFSLRCAS Create Active Stream New Stream

Module: DFSLRCAS Create Active Stream Routine

Explanation: Record cut on create new Stream to DFSLRCAS (Level - Low)

Trace Subcode LRCAS New Strm

Offset	Type	Length	Description
4	Fixed	4	Addr of STB block
8	Character	8	Instance name
16	Fixed	4	Conversation token
20	Fixed	4	Initial Routing Position

3741

Table 150. Trace Record 3741 - DFSLRCAS Create Active Stream Allocate Conversation

Module: DFSLRCAS Create Active Stream Allocate Conversation

Explanation: Record cut on allocate conversation to exist stream (Level - Low)

Trace Subcode LRCAS All Conv

Offset	Type	Length	Description
4	Fixed	4	Addr of STB block
8	Character	8	STB active Instance name
16	Fixed	4	Conversation token
20	Fixed	4	Routing Position

3742

Table 151. Trace Record 3742 - DFSLRCAS Create Active Stream Set Position

Module: DFSLRCAS Create Active Stream Set Position

Explanation: Record cut on set the current position (Level - Low)

Trace Subcode LRCAS Set Pos

Offset	Type	Length	Description
4	Fixed	4	Addr of STB block
8	Fixed	4	STB routing prilog token
12	Character	8	STB last routed LSN

Trace Entry: Log Router Active Conversation Support (374x)

374F

Table 152. Trace Record 374F - DFSLRASC Active Stream Control Entry

Module: DFSLRASC Active Stream Control Routine

Explanation: Record cut on entry to DFSLRASC (Level - Medium)

Trace Subcode LRASC Entry

Offset	Type	Length	Description
4	Fixed	1	Entry Function
5	Char	3	Spares
8	Address	4	STB Address
12	Address	4	SAA Address
16	Address	4	SRA Address
20	Char	8	Active Instance Name

Trace Entry: Log Router Online Forward Recovery (375x)

3750

Table 153. Trace Record 3750 - Initiate online forward recovery (OFR)

Module: DFSLRORH Online Forward Recovery Request Handler

Explanation: Record cut on entry to and exit from DFSLRORH (Level - Low)

Trace Subcode LRORH Request

Offset	Type	Length	Description
4	Address	4	OFB address
8	Address	4	OFRL address
12	Fixed	4	OFR identifier
16	Fixed	4	return Code
20	Fixed	4	DBRC return code

3751

Table 154. Trace Record 3751 - Create the OFR ITASK

Module: DFSLROIC Online Forward Recovery Controller

Explanation: Record cut after OFR ITASK created (Level - Low)

Trace Subcode LROIC Start

Offset	Type	Length	Description
4	Address	4	OFB address
8	Address	4	OFRL address

Table 154. Trace Record 3751 - Create the OFR ITASK (continued)

12	Fixed	4	OFR identifier
16	Address	4	ECB address
20	Fixed	4	current OFR count

3752

Table 155. Trace Record 3752 - OFR processor request

Module: DFSLROPR Online Forward Recovery Processor
Explanation: Record cut at entry to DFSLROPR (Level - Low)
Trace Subcode LROPR Request

Offset	Type	Length	Description
4	Address	4	OFB address
8	Address	4	OFRL address
12	Address	4	buffer address if AWLGFUNC=002E, AWE address otherwise
16	Fixed	2	AWLGFUNC
18	Bit	1	OFB_FLAGS ------------------------
	1...	1	ofb_started
	.1..		ofb_in_merge
	..1.		ofb_terminated
	...1		ofb_restarted
	.... 1...		ofb_pending
	.... .1..		ofb_terminating
	11		* -------------------------------
20	Fixed	2	index to POS_SS entry if AWLGFUNC=002E, 0 otherwise

3753

Table 156. Trace Record 3753 - OFR processor exit

Module: DFSLROPR Online Forward Recovery Processor
Explanation: Record cut at exit from DFSLROPR (Level - Low)
Trace Subcode LROPR Exit

Offset	Type	Length	Description
4	Address	4	OFB address
8	Address	4	OFRL address
12	Address	4	buffer address if AWLGFUNC=002E, AWE address otherwise
16	Fixed	2	AWLGFUNC
18	Bit	2	OFB_flags
20	Fixed	4	OFR identifier

3754

Table 157. Trace Record 3754 - Log descriptors obtained from DBRC

Module: DFSLRORH Online Forward Recovery Request Handler
Explanation: Record cut for each log descriptor (LDSD) (Level - Low)
Trace Subcode LRORH Log Desc

Offset	Type	Length	Description
4	Address	4	OFR identifier
8	Character	8	LDSD_ssid
16	Character	4	LDSD_first_LSN(5:8)
20	Character	4	LDSD_last_LSN(5:8)
24	Bit	1	LDSD_flags
26	Fixed	2	LDSD_mergeID

Table 157. Trace Record 3754 - Log descriptors obtained from DBRC (continued)

28	Character	4	LDSD_prilog_time(5:8)

3756

Table 158. Trace Record 3756 - Log descriptors obtained from DBRC

Module: DFSLRORM Online Forward Recovery Read Next Data set

Explanation: Record cut for each log descriptor (LDSD) (Level - Low)

Trace Subcode LRORM Log Desc

Offset	Type	Length	Description
4	Address	4	OFR identifier
8	Character	8	LDSD_ssid
16	Character	4	LDSD_first_LSN(5:8)
20	Character	4	LDSD_last_LSN(5:8)
24	Bit	1	LDSD_flags
26	Fixed	2	LDSD_mergeID
28	Character	4	LDSD_prilog_time(5:8)

3757

Table 159. Trace Record 3757 - Log descriptors obtained from DBRC

Module: DFSLRORM - Online Forward Recovery Read Next Data Set

Explanation: During OFR, DBRC returned a start point for a stream that was earlier than the stream's current routed position. (Level - Low)

Trace Subcode LRORM Startpoint Error

Offset	Type	Length	Description
4	Fixed	4	pos_old_ptoken
8	Character	4	pos_old_LSN(5:8)
12	Fixed	4	pos_new_ptoken
16	Character	4	pos_new_LSN(5:8)
20	Fixed	2	ofb_flags(0-15)
22	Fixed	2	index to OFRL_entity
28	Character	8	DB/Area name

3758

Table 160. Trace Record 3758 - Start Points List Error detected

Module: DFSLROPR - Log Router Online Forward Recovery Processor

Explanation: During OFR, the record ID (first LSN in buffer) of the next buffer to process is after the start LSN in the startpoints list (ofrsp_start_lsn) and the process has not yet reached this start LSN. (Level - Low)

Trace Subcode LRORM Startpoint Missed

Offset	Type	Length	Description
4	Fixed	4	pos_ptoken
8	Character	8	pos_LSN
16	Fixed	4	index to OFRL_entity
20	Character	4	ofrsp_start_lsn(5:8)
24	Character	4	lgb_record_ID(5:8)

Trace Entry: Log Router Automatic Archive (376x)

3760

Table 161. Trace Record 3760 - DFSLRARC Auto Archive Controller entry

Module: DFSLRARC Auto Archive Controller

Explanation: Record cut on entry to DFSLRARC for archive request (Level - Medium)

Trace Subcode LRARC Request

Offset	Type	Length	Description	
4	Fixed	2	AWLGFUNC='3E'x	x
6	Character	1	*	
7	Character	1	AWLGAtype	
8	Character	8	AWLGASSID	
16	Character	8	AWLGATIME	
24	Character	8	AWLGRTIME	

Table 162. Trace Record 3760 - DFSLRARC Auto Archive Controller entry

Module: DFSLRARC Auto Archive Controller

Explanation: Record cut on entry to DFSLRARC for available request (Level - Medium)

Trace Subcode LRARC Request

Offset	Type	Length	Description
4	Fixed	2	AWLGFUNC='3F'x
6	Character	2	*
8	Fixed	4	AAB address
12	Bit	16	AAB_flags
	1...	1	AAB_START
	.1..		AAB_INIT_ERROR
	..1.		AAB_TERMINATE
	...1		AAB_BATCH
	.... 1...		AAB_READER_EXIST
	.... .1..		AAB_SAR_EXIST
	.... ..1.		AAB_LDSD_LAST
	1		*
	1...	1	AAB_READ_COMPLETED
	.1..		AAB_XBUF_ENQD
	..1.		AAB_ALL_RB_RETURNED
	...1		AAB_RDR_INALLOC
	.... 1...		AAB_READ_ERROR
	.... .1..		*
	.... ..1.		AAB_READ_DCB
	1		AAB_TS_DUAL
	1...	1	AAB_TAP
	.1..		AAB_EOV
	..1.		AAB_WRITE_ERROR
	...1		*
	.... 1...		AAB_ARC_SLDS_DONE
	.... .1..		AAB_AS_LAST_WRITE
	.... ..1.		AAB_AS_DCB
	1		AAB_AS_DUAL
	1...	1	AAB_RLDS_REQD
	.1..		AAB_ARC_RLDS_DELETE
	..11		*
	.... 1...		AAB_ARC_RLDS_DONE
	.... .1..		AAB_AR_LAST_WRITE
	.... ..1.		AAB_AR_DCB
	....		AAB_AR_DUAL

3761

Table 163. Trace Record 3761 - DFSLRARC Auto Archive Controller exit

Module: DFSLRARC Auto Archive Controller

Explanation: Record cut on exit from DFSLRARC (Level - Medium)

Table 163. Trace Record 3761 - DFSLRARC Auto Archive Controller exit (continued)

Trace Subcode LRARC Exit

Offset	Type	Length	Description
4	Fixed	2	Feedback Code

3762

Table 164. Trace Record 3762 - DFSLRARP Auto Archive Processor entry

Module: DFSLRARP Auto Archive Processor

Explanation: Record cut on entry to DFSLRARP for archive request (Level - Medium)

Trace Subcode LRARP Request

Offset	Type	Length	Description
4	Fixed	2	AWLGFUNC='3E'x
6	Character	2	*
8	Fixed	4	AAB address
12	Character	4	LDSD_FLRID
16	Fixed	4	AAB_LDSD_LIST
20	Character	4	LDSD_LLRID
24	Character	4	*
28	Bit	16	AAB_flags
	1...	1	AAB_START
	.1..		AAB_INIT_ERROR
	..1.		AAB_TERMINATE
	...1		AAB_BATCH
	.... 1...		AAB_READER_EXIST
	.... .1..		AAB_SAR_EXIST
	.... ..1.		AAB_LDSD_LAST
	1		*
	1...	1	AAB_READ_COMPLETED
	.1..		AAB_XBUF_ENQD
	..1.		AAB_ALL_RB_RETURNED
	...1		AAB_RDR_INALLOC
	.... 1...		AAB_READ_ERROR
	.... .1..		*
	.... ..1.		AAB_READ_DCB
	1		AAB_TS_DUAL
	1...	1	AAB_TAP
	.1..		AAB_EOV
	..1.		AAB_WRITE_ERROR
	...1		*
	.... 1...		AAB_ARC_SLDS_DONE
	.... .1..		AAB_AS_LAST_WRITE
	.... ..1.		AAB_AS_DCB
	1		AAB_AS_DUAL
	1...	1	AAB_RLDS_REQD
	.1..		AAB_ARC_RLDS_DELETE
	..11		*
	.... 1...		AAB_ARC_RLDS_DONE
	.... .1..		AAB_AR_LAST_WRITE
	.... ..1.		AAB_AR_DCB
	1		AAB_AR_DUAL

Table 165. Trace Record 3762 - DFSLRARP Auto Archive Processor entry

Module: DFSLRARP Auto Archive Processor

Explanation: Record cut on entry to DFSLRARP for return read buffer (Level - Medium)

Trace Subcode LRARP Entry

Offset	Type	Length	Description
4	Fixed	2	AWLGFUNC='2E'x
6	Character	2	*
8	Fixed	4	AAB address
12	Character	4	LRB_RECORD_ID

Table 165. Trace Record 3762 - DFSLRARP Auto Archive Processor entry (continued)

16	Fixed	4	AWLG_RBF_LRB
20	Character	4	LRB_LLSN
24	Bit	4	AWE's flags
	1...		AWLG_RBF_READ_COMPLET
	.1..		AWLG_RBF_IO_ERROR
	..1.		AWLG_RBF_DATASET_OPEN
	...1		LRB_BUFFER_DS_FULL
	.... 1...		LRB_BUFFER_IO_ABEND
	.... .1..		LRB_READ_COMPLETE
	.... ..1.		LRB_BUFFER_ENDDS
	1		LRB_AA_LAST_RETURN
25	1...		AWLG_RBF_NODATA
26	Character	2	*
28	Bit	16	AAB_flags
	1...	1	AAB_START
	.1..		AAB_INIT_ERROR
	..1.		AAB_TERMINATE
	...1		AAB_BATCH
	.... 1...		AAB_READER_EXIST
	.... .1..		AAB_SAR_EXIST
	.... ..1.		AAB_LDSD_LAST
	1		*
	1...	1	AAB_READ_COMPLETED
	.1..		AAB_XBUF_ENQD
	..1.		AAB_ALL_RB_RETURNED
	...1		AAB_RDR_INALLOC
	.... 1...		AAB_READ_ERROR
	.... .1..		*
	.... ..1.		AAB_READ_DCB
	1		AAB_TS_DUAL
	1...	1	AAB_TAP
	.1..		AAB_EOV
	..1.		AAB_WRITE_ERROR
	...1		*
	.... 1...		AAB_ARC_SLDS_DONE
	.... .1..		AAB_AS_LAST_WRITE
	.... ..1.		AAB_AS_DCB
	1		AAB_AS_DUAL
	1...	1	AAB_RLDS_REQD
	.1..		AAB_ARC_RLDS_DELETE
	..11		*
	.... 1...		AAB_ARC_RLDS_DONE
	.... .1..		AAB_AR_LAST_WRITE
	.... ..1.		AAB_AR_DCB
	1		AAB_AR_DUAL

Table 166. Trace Record 3762 - DFSLRARP Auto Archive Processor entry

Module: DFSLRARP Auto Archive Processor

Explanation: Record cut on entry to DFSLRARP for return write Buffer (Level - Medium)

Trace Subcode LRARP Entry

Offset	Type	Length	Description
4	Fixed	2	AWLGFUNC='08'x
6	Character	2	*
8	Fixed	4	AAB address
12	Fixed	4	*
16	Address	4	AWLG_RTBBUFP
20	Character	4	*
24	Bit	4	AWE's flags
	1...		AWLG_RTB_TRK

Table 166. Trace Record 3762 - DFSLRARP Auto Archive Processor entry (continued)

	.1..		AWLG_RTB_ARC
	..1.		AWLG_RTB_RLD
	...1		AWLG_RTB_WRITE_COMPLE
	.... 1...		AWLG_RTB_IO_ERROR
	.... .1..		AWLG_RTB_EOV
	.... .111		*
26	Character	2	*
28	Bit	4	AAB flags
	1...	1	AAB_START
	.1..		AAB_INIT_ERROR
	..1.		AAB_TERMINATE
	...1		AAB_BATCH
	.... 1...		AAB_READER_EXIST
	.... .1..		AAB_SAR_EXIST
	.... ..1.		AAB_LDSD_LAST
	1		*
	1...	1	AAB_READ_COMPLETED
	.1..		AAB_XBUF_ENQD
	..1.		AAB_ALL_RB_RETURNED
	...1		AAB_RDR_INALLOC
	.... 1...		AAB_READ_ERROR
	.... .1..		*
	.... ..1.		AAB_READ_DCB
	1		AAB_TS_DUAL
	1...	1	AAB_TAP
	.1..		AAB_EOV
	..1.		AAB_WRITE_ERROR
	...1		*
	.... 1...		AAB_ARC_SLDS_DONE
	.... .1..		AAB_AS_LAST_WRITE
	.... ..1.		AAB_AS_DCB
	1		AAB_AS_DUAL
	1...	1	AAB_RLDS_REQD
	.1..		AAB_ARC_RLDS_DELETE
	..11		*
	.... 1...		AAB_ARC_RLDS_DONE
	.... .1..		AAB_AR_LAST_WRITE
	.... ..1.		AAB_AR_DCB
	1		AAB_AR_DUAL

Table 167. Trace Record 3762 - DFSLRARP Auto Archive Processor entry

Module: DFSLRARP Auto Archive Processor

Explanation: Record cut on entry to DFSLRARP for Auto Archive Data set (Level - Medium)

Trace Subcode LRARP Entry

Offset	Type	Length	Description
4	Fixed	2	AWLGFUNC='47'x
6	Character	2	AAB_TRK_LDSD_NUM
8	Fixed	4	AAB address
12	Fixed	2	*
14	Fixed	2	AAB_TRK_ADS_IN
16	Address	4	AWLG_ADS_LTDCB
18	Address	4	AWLG_ADS_NUM_DATASETS
20	Character	4	*
24	Bit	2	AWLG_ADS_DSTYPE_flags
	1...		AWLG_ADS_TRACKING_SLDS
	.1..		AWLG_ADS_ARCHIVE_SLDS
	..1.		AWLG_ADS_ARCHIVE_RLDS
	...1 1111		*
26	Character	2	*

Table 167. Trace Record 3762 - DFSLRARP Auto Archive Processor entry (continued)

28	Bit	4	AAB flags
	1...	1	AAB_START
	.1..		AAB_INIT_ERROR
	..1.		AAB_TERMINATE
	...1		AAB_BATCH
	.... 1...		AAB_READER_EXIST
	.... .1..		AAB_SAR_EXIST
	.... ..1.		AAB_LDSD_LAST
	1		*
	1...	1	AAB_READ_COMPLETED
	.1..		AAB_XBUF_ENQD
	..1.		AAB_ALL_RB_RETURNED
	...1		AAB_RDR_INALLOC
	.... 1...		AAB_READ_ERROR
	.... .1..		*
	.... ..1.		AAB_READ_DCB
	1		AAB_TS_DUAL
	1...	1	AAB_TAP
	.1..		AAB_EOV
	..1.		AAB_WRITE_ERROR
	...1		*
	.... 1...		AAB_ARC_SLDS_DONE
	.... .1..		AAB_AS_LAST_WRITE
	.... ..1.		AAB_AS_DCB
	1		AAB_AS_DUAL
	1...	1	AAB_RLDS_REQD
	.1..		AAB_ARC_RLDS_DELETE
	..11		*
	.... 1...		AAB_ARC_RLDS_DONE
	.... .1..		AAB_AR_LAST_WRITE
	.... ..1.		AAB_AR_DCB
	1		AAB_AR_DUAL

3763

Table 168. Trace Record 3763 - DFSLRARC get LDSD list from DBRC

Module: DFSLRARC Auto Archive Controller

Explanation: Record cut after back from DBRC (Level - Medium)

Trace Subcode LRARP To DBRC

Offset	Type	Length	Description
4	Character	4	*
8	Fixed	4	AAB address
12	Fixed	4	AAB_PRILOG_STIME
20	Fixed	4	LDSD_FLRID
24	Fixed	4	LDSD_LLRID
28	Character	8	AAB_LDSD_LIST

3764

Table 169. Trace Record 3764 - DFSLRARP After Create Log Reader

Module: DFSLRARP Auto Archive Processor

Explanation: Record cut after back from create Log Reader (Level - Medium)

Trace Subcode LRARP To Rdr

Offset	Type	Length	Description
4	Fixed	4	Return Code
8	Fixed	4	AAB Address
12	Fixed	4	AAB_LDSD_LIST
16	Fixed	4	AAB_READ_RETQ
20	Character	4	LDSD_FLRID
24	Character	4	LDSD_LLRID

Table 169. Trace Record 3764 - DFSLRARP After Create Log Reader (continued)

28	Fixed	4	AAB_READ_Routine

3765

Table 170. Trace Record 3765 - DFSLRARP Enqueue Buffer to write

Module: DFSLRARP Auto Archive Processor

Explanation: Record cut at enqueue buffer to write (Level - Medium)

Trace Subcode LRARP To SAR

Offset	Type	Length	Description
4	Fixed	4	SAA address
8	Fixed	4	AAB address
12	Character	4	First LSN
16	Fixed	4	LRB address
20	Character	4	Last LSN
28	Bit	16	AAB_flags
	1...	1	AAB_START
	.1..		AAB_INIT_ERROR
	..1.		AAB_TERMINATE
	...1		AAB_BATCH
	.... 1...		AAB_READER_EXIST
	.... .1..		AAB_SAR_EXIST
	.... ..1.		AAB_LDSD_LAST
	1		*
	1...	1	AAB_READ_COMPLETED
	.1..		AAB_XBUF_ENQD
	..1.		AAB_ALL_RB_RETURNED
	...1		AAB_RDR_INALLOC
	.... 1...		AAB_READ_ERROR
	.... .1..		*
	.... ..1.		AAB_READ_DCB
	1		AAB_TS_DUAL
	1...	1	AAB_TAP
	.1..		AAB_EOV
	..1.		AAB_WRITE_ERROR
	...1		*
	.... 1...		AAB_ARC_SLDS_DONE
	.... .1..		AAB_AS_LAST_WRITE
	.... ..1.		AAB_AS_DCB
	1		AAB_AS_DUAL
	1...	1	AAB_RLDS_REQD
	.1..		AAB_ARC_RLDS_DELETE
	..11		*
	.... 1...		AAB_ARC_RLDS_DONE
	.... .1..		AAB_AR_LAST_WRITE
	.... ..1.		AAB_AR_DCB
	1		AAB_AR_DUAL

Trace Entry: Log Router Isolated Log Transport (377x)

3770

Table 171. Trace Record 3770 - Isolated Log Transport Control Routine Entry

Module: DFSLRILT Isolated Log Control Routine

Explanation: Record cut at entry to DFSLRILT (Level - Low)

Trace Subcode LRILT Request

Offset	Type	Length	Description
4	Fixed	4	Log Router AWE Address
8	Fixed	2	Reserved
10	Fixed	2	Isolated Log Request
12	Char	16	AWE parameters

3771

Table 172. Trace Record 3771 - Isolated Log Transport Control Routine Exit

Module: DFSLRILT Isolated Log Control Routine

Explanation: Record cut at exit from DFSLRILT (Level - Low)

Trace Subcode LRILT Exit

Offset	Type	Length	Description
4	Fixed	4	Log Router AWE Address
8	Fixed	2	Isolated Log Request
10	Fixed	2	Feedback Code
12	Fixed	4	Return Code

3772

Table 173. Trace Record 3772 - Isolated Log Transport Send Routine Entry

Module: DFSLRSCM Isolated Log Send Routine

Explanation: Record cut at entry to DFSLRSCM (Level - Low)

Trace Subcode LRSCM Send

Offset	Type	Length	Description
4	Fixed	2	ILTR length
6	Fixed	2	ILTR type
8	Fixed	4	ILTR Sequence #
12	Fixed	16	Trace Data

3773

Table 174. Trace Record 3773 - Isolated Log Transport Schedule Control Message

Module: DFSLRICM Isolated Log Schedule Control Message Routine

Explanation: Record cut at entry to DFSLRICM (Level - Low)

Trace Subcode LRICM Receive

Offset	Type	Length	Description
4	Char	24	Trace Data

3774

Table 175. Trace Record 3774 - Isolated Log Transport Gap Fill

Module: DFSLRICM Isolated Log Schedule Control Message Routine

Explanation: Record cut at entry to DFSLRICM Gap Fill Response (Level - Low)

Trace Subcode LRICM Gap Fill

Offset	Type	Length	Description
4	Fixed	2	Request ID
6	Fixed	2	Request Status
8	Fixed	4	Num Data sets
10	Fixed	4	PRILOG token
16	Char	8	PRILOG Time

3775

Table 176. Trace Record 3775 - Isolated Log Transport Query Response

Module: DFSLRICM Isolated Log Schedule Control Message Routine

Explanation: Record cut at entry to DFSLRICM Query Response (Level - Low)

Trace Subcode LRICM Query

Offset	Type	Length	Description
4	Fixed	4	PRILOG token

Table 176. Trace Record 3775 - Isolated Log Transport Query Response (continued)

8	Fixed	4	High PRILOG token
12	Fixed	4	DBRC rc

3776

Table 177. Trace Record 3776 - Isolated Log Transport DS Abort

Module: DFSLRICM Isolated Log Schedule Control Message Routine

Explanation: Record cut at entry to DFSLRICM DS Abort (Level - Low)

Trace Subcode LRICM DS Abort

Offset	Type	Length	Description
4	Fixed	2	Request ID
6	Fixed	1	Reserved
7	Fixed	1	flags
8	Char	8	First LSN
15	Char	4	Last LSN
18	Char	4	End Data Set

3777

Table 178. Trace Record 3777 - Isolated Log Transport Receive DS

Module: DFSLRIDS Isolated Log DS Processor Routine

Explanation: Record cut at entry to DFSLRIDS Receive DS (Level - Low)

Trace Subcode LRIDS Receive

Offset	Type	Length	Description
4	Fixed	2	Request ID
5	Fixed	2	Reserved
8	Char	4	First LSN
12	Char	4	Last LSN
16	Fixed	4	gds Address
20	Fixed	4	sra Address
24	Fixed	4	stb Address

3778

Table 179. Trace Record 3778 - Isolated Log Transport Send OK

Module: DFSLRIDS Isolated Log DS Processor Routine

Explanation: Record cut at entry to DFSLRIDS Send OK (Level - Low)

Trace Subcode LRIDS Send OK

Offset	Type	Length	Description
4	Fixed	2	ILTR type
6	Fixed	2	Reserved
8	Fixed	4	ILTR Sequence #

3779

Table 180. Trace Record 3779 - Isolated Log Transport DS Received

Module: DFSLRIDS Isolated Log DS Processor Routine

Explanation: Record cut at entry to DFSLRIDS DS Received (Level - Low)

Trace Subcode LRIDS Received

Offset	Type	Length	Description
4	Fixed	2	Request ID

Table 180. Trace Record 3779 - Isolated Log Transport DS Received (continued)

6	Fixed	2	Reserved
8	Fixed	4	SRA Address
12	Fixed	4	STB Address

377A

Table 181. Trace Record 377A - Isolated Log Transport DS Abort

Module: DFSLRIDS Isolated Log DS Processor Routine

Explanation: Record cut at entry to DFSLRIDS DS Abort (Level - Low)

Trace Subcode LRIDS DS Abort

Offset	Type	Length	Description
4	Fixed	2	Request ID
6	Fixed	1	Reserved
7	Fixed	1	flags
	1.......		Data set temporarily unavailable, immediate retry ok
	.1......		Data set temporarily unavailable, defer retry
	..1.....		Begin data set not sent
8	Char	8	First LSN
16	Char	4	Last LSN
20	Char	4	End DS LSN

Trace Entry: Log Router Miscellaneous Trace Codes (378x)

3780

Table 182. Trace Record 3780 - Milestone Request Entry

Module: DFSLRMIL Milestone Processor Routine

Explanation: Record cut at entry to DFSLRMIL (Level - Low)

Trace Subcode LRMIL entry

Offset	Type	Length	Description
4	Fixed	4	Milestone index
8	Fixed	4	LGB current milestone index
12	Char	1	flags
	1.......		Shutdown milestone
	.1......		Takeover milestone
	..1.....		Timer pop
24	Char	8	TimeStamp

3781

Table 183. Trace Record 3781 - Milestone Complete

Module: DFSLRMIL Milestone Processor Routine

Explanation: Record cut at exit to DFSLRMIL (Level - Medium)

Trace Subcode LRMIL entry

Offset	Type	Length	Description
4	Fixed	4	Milestone index
8	Fixed	4	LGB current milestone index
12	Char	1	flags
	1.......		Shutdown milestone
	.1......		Takeover milestone

Table 183. Trace Record 3781 - Milestone Complete (continued)

	..1.....		Timer pop
13	Char	3	Spares
16	Fixed	4	LGB restart milestone index
24	Char	8	TimeStamp

3782

Table 184. Trace Record 3782 - Unplan Takeover Process Phase 1 entry

Module: DFSLRTK0 Unplan Takeover Process Routine

Explanation: Record cut at entry to unplan takeover phase 1 (Level - Low)

Trace Subcode LRTK0 entry

Offset	Type	Length	Description
4	Fixed	4	LGB current milestone index
24	Char	8	TimeStamp

3783

Table 185. Trace Record 3783 - Unplan Takeover Process Phase 2 entry

Module: DFSLRTK0 Unplan Takeover Process Routine

Explanation: Record cut at entry to unplan takeover phase 2 (Level - Low)

Trace Subcode LRTK0 entry

Offset	Type	Length	Description
4	Fixed	4	LGB current milestone index
24	Char	8	TimeStamp

3784

Table 186. Trace Record 3784 - Log Router Master ITASK request

Module: DFSLRMST Master ITASK process Routine

Explanation: Record cut at entry to DFSLRMST (Level - Low)

Trace Subcode LRTK0 entry

Offset	Type	Length	Description
4	Fixed	4	Function Code
8	Fixed	4	Request AWE's AWLGCECB
12	Fixed	4	Data pointed by Request AWE's AWLGCECB

3785

Table 187. Trace Record 3785 - Log Router Master ITASK request done

Module: DFSLRMST Master ITASK process Routine

Explanation: Record cut after done the request to DFSLRMST (Level - Low)

Trace Subcode LRTK0 entry

Offset	Type	Length	Description
4	Bit	8	Takeover flags
12	1.......		Planned takeover requested
	.1......		Planned takeover in progress
	..1.....		Unplanned takeover requested
	...1....		Unplanned takeover in progress
16	Fixed	4	Current milestone index
24	Char	8	TimeStamp

3786

Table 188. Trace Record 3786 - Log Router Master ITASK exit

Module: DFSLRMST Master ITASK process Routine

Explanation: Record cut at exit to DFSLRMST (Level - Low)

Trace Subcode LRTK0 entry

Offset	Type	Length	Description
24	Char	8	TimeStamp

3787

Table 189. Trace Record 3787 - Log Router End DataBase Tracking

Module: DFSLREDT End Database/Area Tracking Routine

Explanation: Record cut at entry to DFSLREDT (Level - Low)

Trace Subcode LRTK0 entry

Offset	Type	Length	Description
4	Char	8	Database name
12	Char	8	Area name
20	Fixed	4	Milestone index

3788

Table 190. Trace Record 3788 - Create Active Stream begin takeover

Module: DFSLRCAS Create Active Stream Routine

Explanation: Record cut at begin planned takeover (Level - low)

Trace Subcode LRTK0 entry

Offset	Type	Length	Description
4	Fixed	4	LGB current mile index
8	Fixed	4	LGB plan tko token

Chapter 16. CQS Diagnosis

This chapter describes diagnostic information that helps you analyze problems in CQS.

In this Chapter:

CQS Log Records

CQS writes records to the MVS log stream that contains all CQS log records from all CQSs that are connected to a structure pair. You can use the log records to:

- Diagnose problems related to the CQS address space.

  For CQS internal errors, the IBM support representative will direct you to print the appropriate log records.

  You can sometimes use information in the log records to set up a keyword string to search APAR descriptions and compare them to your own problem.
- Generate various reports related to the CQS address space, such as statistics about the number of requests.

  By knowing the content and format of the log records, you can set up a DFSERA10 job to format and print the specific log records you want.

Each CQS log record contains a log record prefix, followed by data that is unique to the record. Macro CQSLGRFX maps the log record prefix.

You can view the CQS log record formats by assembling mapping macro CQSLGREC with TYPE=ALL.

For each CQS log record, Table 191 lists:
- The log record type and subtype.
- The macro that maps the record.
- The events that cause the record to be written.

Table 191. CQS Log Records

Type	Sub type	Mapping Macro	Conditions for Writing the Log Record
X'03'	X'01'	CQSLGCON	CQSCONN request: The client connect to a structure completed.
X'04'	X'01'	CQSLGDSC	CQSDISC request: The client disconnect from a structure completed.
X'07'	X'01'	CQSLGPUT	CQSPUT OBJECT request completed.
	X'02'		CQSPUT COMMIT request completed.
	X'03'		CQSPUT START request completed.
	X'04'		CQSPUT FORGET request completed.
	X'05'		CQSPUT ABORT request completed.
	X'06'		CQSPUT request failed.
	X'07'		CQSPUT system checkpoint record was written.
	X'08'		CQSPUT FORGET request completed. This is a batched log record.
X'08'	X'01'	CQSLGRD	CQSREAD request completed.
	X'02'		CQSREAD request failed.
	X'03'		CQSREAD system checkpoint record was written.
		CQSLGCHD	This system checkpoint header record is not a complete log record, but it is used in CQSLGPUT and CQSLGRD system checkpoint log records.

Table 191. CQS Log Records (continued)

Type	Sub type	Mapping Macro	Conditions for Writing the Log Record
X'0B'	X'01'	CQSLGMOV	CQSMOVE or CQSUNLCK request completed.
	X'02'		CQSMOVE or CQSUNLCK request failed.
	X'03'		CQSMOVE or CQSUNLCK request moved an object between the primary and overflow structure.
X'0D'	X'01'	CQSLGDEL	CQSDEL request: Delete-type 1 (delete by token) completed.
	X'02'		CQSDEL request: Delete-type 2 (delete by queue name) completed.
	X'03'		CQSDEL request: Delete-type 3 (delete by queue name and UOW) completed.
	X'04'		CQSDEL request: Delete-type 1 (delete by token) completed. This is a batched log record.
		CQSLGBHD	This batched log record header record is not a complete log record, but is used in CQSLGPUT and CQSLGDEL batched log records.
X'10'	X'01'	CQSLGSHT	CQSSHUT request completed.
X'32'	X'01'	CQSLGYCH	System checkpoint started.
	X'02'		System checkpoint ended.
	X'03'		System checkpoint failed.
X'40'	X'01'	CQSLGIST	Beginning of log stream.
X'42'	X'01'	CQSLGTCH	Structure checkpoint started.
	X'02'		Structure checkpoint ended.
	X'03'		Structure checkpoint failed.
X'43'	X'01'	CQSLGRBL	Structure rebuild started. Statistics about the old structure, the rebuild structure, and rebuild failure are mapped by CQSSSTT6.
	X'02'		Structure rebuild ended. Statistics about the old structure, the rebuild structure, and rebuild failure are mapped by CQSSSTT6.
	X'03'		Structure rebuild failed. Statistics about the old structure, the rebuild structure, and rebuild failure are mapped by CQSSSTT6.
	X'04'		Structure rebuild resulted in a lost UOW list. This record lists the lost UOWs.
X'44'	X'01'	CQSLGOFL	Overflow threshold began.
	X'02'		Overflow threshold ended.
	X'03'		Overflow threshold failed.
	X'04'		Overflow mode ended.
	X'05'		Overflow status change.
	X'06'		Qnames were moved to overflow.
	X'07'		Qnames were removed from overflow.
	X'08'		CQSOVERFLOWQNMR, a control list entry containing the list of queue names deleted from overflow, was deleted.
	X'09'		Overflow Scan Begin.
	X'0A'		Overflow Scan End.
	X'0B'		Private Queue Scan Begin.
	X'0C'		Structure to be deleted.

Table 191. CQS Log Records (continued)

Type	Sub type	Mapping Macro	Conditions for Writing the Log Record
X'60'	X'01'	CQSLGSTT	Structure statistics were written at the end of system checkpoint. Individual statistics areas are mapped by CQSSSTT1, CQSSSTT2, CQSSSTT3, CQSSSTT4, and CQSSSTT5.
	X'C0'		Internal BPE service statistics were written at the end of system checkpoint.

Printing CQS Log Records

To print the CQS log records from the MVS system log, use the IMS File Select and Formatting Print utility (DFSERA10) with exit routine CQSERA30. The following example shows the required JCL to print the log records from an MVS system log. This JCL causes the MVS logger to invoke the default log stream subsystem exit routine, IXGSEXIT, to copy the log records. The exit routine returns a maximum of 32760 bytes of data for each log record even though CQS supports larger log records. You can specify the name of a different exit routine, if necessary.

Example: Use the following JCL to print the CQS log records:

```
//CQSERA10 JOB   MSGLEVEL=1,MSGCLASS=A,CLASS=K
//STEP1    EXEC  PGM=DFSERA10
//STEPLIB  DD    DISP=SHR,DSN=IMS.RESLIB
//SYSPRINT DD    SYSOUT=A
//TRPUNCH  DD    SYSOUT=A,DCB=BLKSIZE=80
//SYSUT1   DD    DSN=SYSLOG.MSGQ01.LOG,
//               SUBSYS=(LOGR,IXGSEXIT),
//               DCB=(BLKSIZE=32760)
//SYSIN    DD  *
CONTROL   CNTL H=EOF
OPTION    PRINT EXITR=CQSERA30
END
//
```

DD statements

STEPLIB
DSN= points to IMS.RESLIB, which contains the IMS File Select and Formatting Print utility, DFSERA10.

SYSUT1
DSN= points to the CQS log stream name that was specified in the LOGNAME= parameter in the CQSSGxxx PROCLIB member.

Control Statements

H=
Specifies the number of log records to print. H=EOF prints all log records.

EXITR=CQSERA30
The CQS log record routine that is called to format each log record. This routine prints the record type and time-stamp information for each record, and dumps the contents of the record (up to a maximum of 32760 bytes (X'7FF8')).

Related Reading: For a complete description of the IMS File Select and Formatting Print utility, see *IMS Version 7 Utilities Reference: System*.

Part 4. Appendixes

Appendix A. IMS Keyword Dictionary

If you use a database search tool that requires keywords in a structured database (SDB) format, use this IMS keyword dictionary to translate free-form keywords into the SDB format.

Free-form searches allow you to retrieve only the RETAIN records that contain all the search keywords you specified. You can use the same keywords as a base from which to conduct a structured database search. An SDB prefix, which ends with a slash, identifies the type of symptom. These prefixes are used by all IBM products and are not exclusive to IMS. Examples of keyword strings that use both free form and SDB formats are provided throughout the procedures in Chapter 4, “Selecting the Keywords,” on page 19.

Related Information: For more information about SDB formats, see *Software Service General Information Manual*.

Category/Keyword Examples	RETAIN Formats Keyword	SDB
Abends: System 0C4 User 0845	 ABEND0C4 ABENDU0845	 AB/S00C4 AB/U0845
Access Methods: OSAM VSAM	 OSAM VSAM	 RIDS/OSAM RIDS/VSAM
Automatic Operator Interface	AOI	RIDS/AOI
APARS	PL12345	PTFS/PL12345
Checkpoint Processing: Checkpoint Extended Checkpoint	 CHKPT XCHKPT	 PCSS/CHKPT PCSS/XCHKPT
CICS Interface	CICSDLI	PCSS/CICSDLI
IMS Commands:[1] /ASSIGN /CHECKPOINT /ERESTART /TRACE /STOP	 CMDASS CMDCHE CMDERE CMDTRA CMDSTO	 PCSS/ASS PCSS/CHE PCSS/ERE PCSS/TRA PCSS/STO
DBRC Commands:[2] INIT.RECON CHANGE.PRILOG	 INITRECON CHANGEPRILOG	 PCSS/INITRECON PCSS/CHANGEPRIL
Condition Code	CC08 (HEX)	PRCS/00000008
Control Blocks: Data Control Block Database Descriptor	 DCB DBD	 FLDS/DCB FLDS/DBD
Database Organization	HDAM	PCSS/HDAM
Database Pre-Open	PRE-OPEN	RIDS/PREOPEN
Data Sharing Environment	DATA SHARING	RIDS/DATASHARE
Devices: 3270 LU TYPE1	 D/T3270 SLU1	 DEVS/3270 DEVS/SLU1

5. This is a sample of IMS keywords and is not intended to be a complete list.

Category/Keyword Examples	RETAIN Formats Keyword	SDB
DL/I Address Space	DLISAS	PCSS/DLISAS
DSECTS	IDSPWRK	FLDS/IDSPWRK
Emergency Restart Processing	ERE	RIDS/ERE
Error Codes (DBRC)	EC0182062	PRCS/00182062
Extended Restart	XRST	PCSS/XRST
Fast Path: Fast Path Area Second CI Main Storage Database Sequential Dependent	FASTPATH FPAREA DMAC MSDB SDEP	RIDS/FASTPATH PCSS/FPAREA FLDS/DMAC PCSS/MSDB PCSS/SDEP
Feedback Code	FDBK0C (HEX)	PRCS/0000000C
Fields: PSTUSID	PSTUSID	FLDS/PSTUSID
Function Sub-Function	SYS CHKRT	RIDS/SYS RIDS/CHKRT
Function Codes	FC0291	OPCS/0291
System Definition: ACB NUCLEUS	ACBGEN NUC	PCSS/ACB PCSS/NUC
IRLM	IRLM	RIDS/IRLM
Labels: LOOPNEXT FREEMAIN	LOOPNEXT FREEMAIN	RIDS/LOOPNEXT RIDS/FREEMAIN
Log Records: TYPE 18 TYPE 67FF	TYPE18 TYPE67FF	PCSS/TYPE18 PCSS/TYPE67FF
Macros: RWOS TERMINAL	RWOS TERMINAL	PCSS/RWOS PCSS/TERMINAL
Master Terminal Operator	MTO	PCSS/MTO
Messages: DFS045I IEC030I	MSGDFS045I MSGIEC030I	MS/DFS045I MS/IEC030I
Modules: DFSPCC20	DFSPCC20	RIDS/DFSPCC20
Online Change	OLCHG	PCSS/OLCHG
Online Data Set	OLDS	PCSS/OLDS
Online Image Copy	OLIC	RIDS/OLIC
Parameters: ERROPT=ACCEPT	ERROPT=ACCEPT	PCSS/ERROPT PCSS/ACCEPT
Processing Options: PROCOPT=GO	PROCOPT=GO	PCSS/PROCOPT PCSS/GO
Publication Numbers: SY26-3991-2	SY26399102	PUBS/SY26399102

Category/Keyword Examples	RETAIN Formats Keyword	SDB
Reason Codes	RSN08 (HEX)	PRCS/00000008
Registers: General purpose registers Control registers Floating point registers	 REG13 (DECIMAL) CREG10 FPREG01	 REGS/GR13 REGS/CR10 REGS/FP01
Restart Processing	RSTRT	RIDS/RSTRT
Release Levels: Version 6 Database Manager Version 6 Transaction Manager	 AR601 AR602	 LVLS/601 LVLS/602
Return Codes: Return code 12 (X'0C')	 RC0C	 PRCS/0000000C
RSR Environment: RSR	 IMSRSR	 RIDS/IMSRSR
Sense Codes: Sense 080B	 SNS080B	 PRCS/0000080B
Status Codes: Status code GE Status blank BLANK	 STATUSGE STATUS4040	 PRCS/000000GE PRCS/00004040
Subcode	SUBCODE101	PRCS/00000101
SVC Numbers	SVC255 (DECIMAL)	OPCS/SVCFE
Trace Entry Function Code	TRACEE6 (DL/I) TRACE03 (DISP)	PCSS/TRACEE6 PCSS/TRACE03
XRF Environments: XRF Takeover Alternate	 IMSXRF TAKEOVER ALTERNATE	RIDS/IMSXRF PCSS/TAKEOVER PCSS/ALTERNATE

Notes:

1. IMS commands begin with the special character “/”, which is not searchable in RETAIN. Therefore, the convention will be the letters “CMD” followed by the first three letters of the command. Please note these keywords are to be used for command processing only.
2. DBRC commands should omit the period (.) because of RETAIN search constraints.

Appendix B. AIBREASN Codes for Message Requeuer Errors

This appendix explains the AIBRETRN code and the AIBREASN codes set by the IMS message requeuer module DFSQMRQ0. These are recorded in both the SCRAPLOG and 6701-MRQE records when an error is detected requeuing messages to the IMS message queue. You use the AIBREASN codes when diagnosing problems with the Message Requeuer. The list beginning on page 451 provides detailed descriptions of the meanings of the AIBREASN codes summarized in Table 192.

For more information about diagnosing problems with the Message Requeuer, see “Diagnosing Problems in the Message Requeuer” on page 266. That section also describes how the Message Requeuer (MRQ) program product communicates with certain functions in the IMS/ESA Transaction Manager and System Services.

Table 192. AIBREASN Codes Set by DFSQMRQ0

Code	Routine	Error Message
X'0004'	ERROR	DEFAULT REASON CODE IF NONE SET
X'0008'	ENTRY	INVALID FUNC PASSED TO QMRQ0 ENTRY
X'000C'	GETLNB	SID PASSED IS ZERO
X'0010'	GETLNB	SID PASSED IS TOO HI VALUE
X'0014'	GETLNB	SID PASSED IS UNDEFINED TO SYSTEM
X'0018'	ENTRY	MSGQ DATA SET INVALID IMS RELEASE
X'1000'	INSERT	INSERT PCB NOT MODIFIABLE
X'1004'	INSERT	1ST ISRT NOT 1ST QUEUE BUFFER
X'1008'	INSERT	CAN’T FIND RACF PREFIX SEGMENT
X'100C'	INSERT	MSC NOT GEN BUT MSC SEG PRESENT
X'1010'	INSERT	MSC NOT GEN BUT ISC SEG PRESENT
X'1014'	INSERT	FINDEST ERR FOR SOURCE=MSGIDSTN
X'1018'	INSERT	MSGIDSTN BLOCK NOT CNT/LNB/QAB
X'101C'	INSERT	CAN’T FIND MSC SEGMENT MSGSIPEX
X'1020'	INSERT	FINDEST ERR FOR SOURCE=MSGMSINM
X'1024'	INSERT	FINDEST ERR FOR DEST = MSGODSTN
X'1028'	INSERT	MSGODSTN BLOCK NOT EXPECTED CNT
X'102C'	INSERT	MSG DEST FLAG NOT EXPECTED LTERM
X'1030'	INSERT	MSG DEST FLAG NOT EXPECTED TRAN
X'1034'	INSERT	DEST BLOCK NOT EXPECTED SMB
X'1038'	INSERT	ETO NEEDED BUT NOT SUPPORTED
X'103C'	INSERT	DEST LNB SID/DEST MSG SID NOMTCH
X'1040'	INSERT	FINDEST ERROR FOR DEST = MSGMSONM
X'1044'	INSERT	MSC DEST BLOCK NOT EXPECTED CNT
X'1048'	INSERT	MSG DEST NOT EXPECTED TRANSACT
X'104C'	INSERT	DEST SMB SID/DEST MSG SID NOMTCH
X'1050'	INSERT	DEST CONV BUT NO SPA SEG IN MSG
X'1054'	INSERT	DEST NOT CONV BUT MSG HAS SPASEG
X'1058'	INSERT	DEST = BLANKS AT CALL QMGR TIME

Table 192. AIBREASN Codes Set by DFSQMRQ0 (continued)

Code	Routine	Error Message
X'105C'	INSERT	DEST NAME INVALID AT CALLQMGR TIME
X'1060'	INSERT	NON ZERO RC ON ISRT CALL TO QMGR
X'1064'	INSERT	MSG CONTAINS INVALID QUEUE NUM
X'1068'	INSERT	MSGMSINM BLOCK NOT CNT TYPE
X'106C'	INSERT	DFSSLC CALL ERR FOR DEST MSGMSONM
X'1070'	INSERT	DFSSLC CALL ERR FOR DEST MSGIDSTM
X'1074'	INSERT	DFSSLC CALL ERR FOR DEST MSGMSINM
X'1078'	INSERT	DFSSLC CALL ERR FOR DST MSGODSTN
X'107C'	INSERT	APPC SEG NEEDED BUT NOT SUPPORTED
X'1080'	INSERT	MSG DEST = APPC SYNC = NON RECOV
X'1084'	INSERT	MSG DEST = NON RECOV
X'1088'	INSERT	MSG WAS CANCELED BY IMS
X'108C'	INSERT	ERROR LOCATING APPC ASYNC DEST
X'1090'	INSERT	MSGMRQF1 FLAG INVALID
X'1094'	INSERT	MSC DEST BLOCK NOT EXPECTED LNB
X'1098'	INSERT	SOURCE/DEST = DFSAPPC INVALID
X'109C'	INSERT	LU6.2 SCD EXTEN INVALID/NOTAVAIL
X'10A0'	INSERT	MSG NOT VALID 01/03 TYPE
X'10A4'	INSERT	INTERNAL IMS MESSAGE
X'10A8'	INSERT	SOURCE/DEST NAME CHANGED
X'10AC'	INSERT	DFSLUMIF BLDPRE ERROR
X'10B0'	INIT	ERROR GETTING DFSPOOL STORAGE
X'10B4'	INIT	ERROR GETTING AN AWE
X'10B8'	INSERT	NO EXTENDED PREFIX PRESENT
X'10BC'	INIT	ERROR INIT/ADDRESSING QMRQWORK
X'10C0'	INIT	CAN'T FIND RACF SEGMENT MSGSORAC
X'10C4'	INIT	CAN'T FIND LU6.1 SEGMENT MSGSILU6
X'10C8'	INIT	CAN'T FIND APPC SEGMENT MSGSOAP0
X'10CC'	INIT	CAN'T FIND EPH SEGMENT MSGSIEPH
X'10D0'	INIT	CAN'T FIND APPC SEGMENT MSGSIAP0
X'10D4'	INIT	CAN'T FIND SEC SEGMENT MSGSISEC
X'10D8'	INIT	CAN'T FIND WLM SEGMENT MSGSIWLM
X'10DC'	INIT	CAN'T FIND SYS EXT SEGMENT MSGSISEX
X'10E0'	INIT	CAN'T FIND MSC EXT SEGMENT MSGSIMEX
X'10E4'	ISRT	OTMA MESSAGES NOT SUPPORTED
X'10E8'	ISRT	MSC/APPC MESSAGE NOT SUPPORTED
X'10EC'	ISRT	MESSAGE REROUT NOT SUPPORTED
X'10F0'	ISRT	MSC SEG ITEM NOT PRESENT
X'2000'	PURGE	PURGE PCB NOT MODIFIABLE
X'2004'	PURGE	PURGE PCB DEST INVALID

Table 192. AIBREASN Codes Set by DFSQMRQ0 (continued)

Code	Routine	Error Message
X'2008'	PURGE	PURGE PCB DEST SET TO BLANKS
X'200C'	PURGE	PURGE DEST CTL BLK ADDR ZERO
X'2010'	PURGE	PURGE DEST NAME = DFS INVALID
X'2014'	PURGE	PURGE INQUIRY DEST NOT SIGNED ON
X'2018'	PURGE	PURGE NON 0 RC ON QMGR ENQ CALL
X'201C'	PURGE	PURGE I/O AREA INVALID
X'2020'	PURGE	PURGE MSGMRQF1 FLAG INVALID
X'2024'	PURGE	DEST BLK=DFSAPPC BUT MSG NOT APPC
X'3000'	SETPRFX	MESSAGE PREFIX SIZE INVALID
X'4000'	CPYPRFX	PREFIX SIZE NOT EXPECTED
X'4004'	CPYPRFX	CAN'T FIND MSC SEGMENT MSGSIPEX
X'5000'	CANCEL	NON ZERO RC ON CANCEL CALL TO QMGR
X'6004'	FMQINSRT	LOGREC TYPE NOT 4002, 01, OR 03
X'6008'	FMQINSRT	NO SECONDARY LOGREC WHEN EXPECTED
X'600C'	FMQINSRT	SECONDARY LOGREC DEST INVALID
X'7004'	XLATPFX	CAN'T FIND SYS EXT SEGMENT MSGSISEX

AIB Return Codes Set by DFSQMRQ0

X'000000F0' is a unique AIB return code assigned to the message queue manager message requeuer processor (DFSQMRQ0). It is set in the AIBRETRN field of the AIB by DFSQMRQ0 when an error is detected while requeuing a message to the message queue. DFSQMRQ0 also sets the AIBREASN field in the AIB to a code indicating the type of error detected. These codes are passed back to the MRQ FMQINSRT BMP program. FMQINSRT stores the codes in the MRQ prefix segment that is appended in front of the message record that caused the error. FMQINSRT writes this record to the SCRAPLOG data set. IMS logs a corresponding 6701-MRQE record to the online log data set (OLDS).

AIB return codes other than X'000000F0' indicate IMS errors that are not specific to message requeuing. To analyze these return codes and their associated reason codes, see *IMS Version 7 Messages and Codes, Volume 1*.

Each AIBREASN code associated with AIB return code X'000000F0' is described in the following list. Locate the unique AIBREASN code and analyze the error as described. Each AIBREASN code falls into one of three categories:

1. Error is a normal condition and AIBREASN is set for informational purposes. The message is discarded according to protocol. There are five AIBREASN codes in this category:

   1080 Message is an APPC synchronous conversation type.

   1084 Message is a nonrecoverable type.

   1088 Message was flagged to be canceled.

   10A4 Message is an internal IMS message that is not recoverable.

   2014 Destination is an inquiry LTERM not signed on.

2. Error is most likely due to unsupported or changed IMS features or destination or source resource names. An example is a transaction that was deleted from the SYSGEN and the MRQ tried to requeue a message destined for the deleted transaction. The MRQ processor would detect that the destination

no longer exists and set an AIBREASN code of 1024 or 1040. The IMS system programmer should analyze these errors (by following the explanations and programmer response guidelines found in the following AIBREASN code list) and verify if the resource has been deleted or altered.

3. Error is an IMS or MRQ internal error and should be reported to your IBM support personnel for resolution.

The following list describes all of the AIB reason codes associated with the AIB return code X'000000F0'.

X'0004' ERROR - DEFAULT REASON CODE IF NONE SET

Explanation: AIBREASN code in R0 = 0 when ERROR routine called.

Programmer Response: Trace back to caller of ERROR routine. This is an IMS internal error.

X'0008' ENTRY - INVLID FUNC PASSED TO QMRQ0 ENTRY

Explanation: DFSQMRQ0 was called with an invalid function code in R1.

Programmer Response: Internal error. Trace back to caller of DFSQMRQ0.

X'000C' GETLNB - SID PASSED IS ZERO

Explanation: Destination system identification (SYSID) or source SYSID of message being processed is zero.

Programmer Response: Locate destination SYSID (MSGMSOID) or source SYSID (MSGMSIID) in message. SYSIDs are extracted from the control block representing the resource (CNT for LTERMS, SMB for transactions) when the message was created. Verify resource was not changed across restart. Except for some internal system messages, SYSID=0 is invalid and should not occur. Possible IMS internal error.

X'0010' GETLNB - SID PASSED IS TOO HI VALUE

Explanation: Destination system identification (SYSID) or source SYSID of message being processed is higher than maximum SYSID defined on MSNAME macros at SYSGEN and stored in SCD at SCDSIDN.

Programmer Response: Locate destination SYSID (MSGMSOID) or source SYSID (MSGMSIID) in message. SYSIDs are extracted from the control block representing the resource (CNT for LTERMS, SMB for transactions) when the message was created. Max SYSID is determined from max SYSID in MSNAME macros at system generation and stored in the SCD at SCDSIDN. Verify that MSNAMES were not removed at system generation and SCDSIDN is correct.

X'0014' GETLNB - SID PASSED IS UNDEFINED TO SYSTEM

Explanation: Destination system identification (SYSID) or source SYSID of message being processed is not defined to system.

Programmer Response: Locate destination SYSID (MSGMSOID) or source SYSID (MSGMSIID) in message. SYSIDs are extracted from the control block representing the resource (CNT for LTERMS, SMB for transactions) when the message was created. To be valid, SYSID must be defined in an MSNAME macro at system generation.

X'0018' ENTRY - MSGQ DATA SET INVALID IMS RELEASE

Explanation: The message being inserted is from an IMS release not supported by this IMS release.

Programmer Response: Locate the I/O AREA. THE MRQ prefix is the first 24 bytes and contains the character string $MRQMSG at offset X'04'. The IMS release of the message is at offset X'0C' for 2 bytes (0310, 0410, and so on). This value is obtained from the type X'4001' checkpoint record by FMQSELCT. FMQSELCT locates the checkpoint ID record from the CHKPT input control statement. This data is passed to FMQINSRT and compared to the current IMS release at SSCDIMSR. The SCD address is in register 11.

Programmer Response: Verify that the message is being requeued from a supported IMS release. This is probably a user error.

X'001C'

Explanation: Reserved for future use.

X'0020'

Explanation: Reserved for future use.

X'0024'

Explanation: Reserved for future use.

X'1000' INSERT - INSERT PCB NOT MODIFIABLE

Explanation: Alternate PCB defined in MRQ PSB is not modifiable type.

Programmer Response: Verify that MODIFY=YES was coded on the PCB named ALTPCB01 for the MRQPSB.

MRQPSB is the default MRQ PSBNAME and might have been changed on the MRQPSBN= parameter of the MSGQUEUE macro at system generation.

X'1004' INSERT - 1ST ISRT NOT 1ST QUEUE BUFFER

Explanation: A new message is being inserted and the first queue buffer message flag (MSGFFRST) is not set on.

Programmer Response: Locate the message flags in the message prefix. If message is a first buffer then MSGFFRST should be set. Verify original message on log and input to FMQSELCT was correct. If not, this is an internal IMS error. If OK, message may have been handled incorrectly by FMQSELCT, FMQCANCL, or FMQINSRT.

X'1008' INSERT - CAN'T FIND RACF PREFIX SEGMENT

Explanation: Message was created with a RACF prefix, but RACF is not initialized.

Programmer Response: If the flag MSGC1RAC is set on and a RACF prefix segment with a code = 83 is not present, this is an internal IMS error.

X'100C' INSERT - MSC NOT GEN BUT MSC SEG PRESENT

Explanation: Message was created with an MSC prefix but MSC is not initialized.

Programmer Response: Locate the message and verify the MSC prefix is present and flag MSGC2MSC is set on. If so, MSC was invoked at system generation when message was created but is not available now. Flag SCDPDMUL is set on at system generation if MSC is invoked at system generation. Regenerate the system with MSC.

X'1010' INSERT - MSC NOT GEN BUT ISC SEG PRESENT

Explanation: Message was created with an ISC prefix but MSC is not initialized.

Programmer Response: Locate the message and verify the ISC prefix is present and flag MSGC2LU6 is set on. The ISC prefix segment item has a MSSSID code of 84. If so, MSC was invoked at system generation when the message was created but is not available now. Flag SCDPDMUL is set on at system generation if MSC is invoked at system generation. Regenerate the system with MSC.

X'1014' INSERT - FINDEST ERR FOR SOURCE=MSGIDSTN

Explanation: The local source name in the message at MSGIDSTN could not be found by the FINDEST routine.

Programmer Response: Locate the MSGIDSTN name in the message and verify that it is a valid local LTERM or MSNAME. If it is ETO, is invoked at system generation and name is a dynamic LTERM, verify that ETO is enabled. FINDEST parameter list used to locate the name is at PSTDCA.

X'1018' INSERT - MSGIDSTN BLOCK NOT CNT/LNB/QAB

Explanation: The control block returned by FINDEST, representing the source name at MSGIDSTN is not a CNT (LTERM), LNB (MSNAME), or QAB (LU 6.2 node).

Programmer Response: Locate the MSGIDSTN name in the message and verify that it is a valid LTERM, MSNAME, or LU 6.2 node. If it is an LU 6.2 node, then MSGIDSTN begins with FEFFFFFF and the NODE name is in the LU 6.2 prefix. Control block address is in REG1 in the REG14-12 area and the block is at QTPDST.

X'101C' INSERT - CAN'T FIND MSC SEGMENT MSGSIPEX

Explanation: Message flag indicates MSC prefix segment is present but segment cannot be located.

Programmer Response: Locate the message and verify the flag MSGC2MSC is set. If set, then MSC prefix segment with a code = 82 must be present. This is an internal IMS error.

X'1020' INSERT - FINDEST ERR FOR SOURCE=MSGMSINM

Explanation: The MSC source name in the message at MSGMSINM could not be found by the FINDEST routine.

Programmer Response: Locate the MSGMSINM name in the message and verify that it is a valid local LTERM. If ETO is invoked at system generation and name is a dynamic LTERM, verify that ETO is enabled.

The MSC LTERM name is only verified if the source SYSID in the message at MSGMSIID is local. Verify that the source SYSID was not changed from a remote SYSID to a local (check MSNAME macros).

X'1024' INSERT - FINDEST ERR FOR DEST = MSGODSTN

Explanation: The local destination name in the message at MSGODSTN could not found by the FINDEST routine.

Programmer Response: Locate the MSGODSTN name in the message and verify that it is a valid local LTERM, MSNAME, or local or remote TRANSACTION CODE. If it is ETO, is invoked at system generation and name is a dynamic LTERM, verify that ETO is enabled. FINDEST parameter list used to locate the name is at PSTDCA.

X'1028' INSERT - MSGODSTN BLOCK NOT EXPECTED CNT

Explanation: The control block returned by FINDEST, representing the destination name at MSGODSTN is not a CNT (LTERM) or MSC LNB (MSNAME).

Programmer Response: Locate the MSGODSTN name in the message and verify that it is a valid LTERM or MSNAME. The control block address is in REG1 in the REG14-12 area and the control block is at QTPDST.

X'102C' INSERT - MSG DEST FLAG NOT EXPECTED LTERM

Explanation: The message destination control block is a CNT type (either an LTERM or MSC MSNAME). However, the destination type flag in the message is not a CNT type.

Programmer Response: Locate the message destination type flag (MSGDFLG2) of the message and it should be a CNT type (X'82'=CNT type, X'81'=SMB type). If flag is X'81' then destination name at MSGODSTN in the message prefix was an SMB type when the message was originally created but now the resource name is a CNT type. The destination control block address is in REG1 in the REG14-12 area and the block is at QTPDST.

X'1030' INSERT - MSG DEST FLAG NOT EXPECTED TRAN

Explanation: The message destination type flag is expected to be an SMB type because the destination control block is an SMB.

Programmer Response: Locate the message destination type flag (MSGDFLG2) of the message and it should be an SMB type (X'81'=SMB type, X'82'=CNT type). If flag is X'82' then destination name at MSGODSTN in the message prefix was a CNT type (either a LTERM or MSNAME) when the message was created but now the resource name is an SMB type. The destination control block address is in REG1 in the REG14-12 area and the block is at QTPDST.

X'1034' INSERT - DEST BLOCK NOT EXPECTED SMB

Explanation: The control block returned by FINDEST, representing the source name at MSGODSTN is not an SMB (either a local or remote transaction code block).

Programmer Response: Locate the MSGODSTN name in the message and verify that it is a valid local or remote transaction code name. The control block address is in REG1 in the REG14-12 area and the block is at QTPDST.

X'1038' INSERT - ETO NEEDED BUT NOT SUPPORTED

Explanation: ETO was needed but was not available.

Programmer Response: This error is not currently set.

X'103C' INSERT - DEST LNB SID/DEST MSG SID NOMTCH

Explanation: The message is enqueued to an MSC logical link MSNAME and the destination SYSID of the message does not match the destination SYSID of the MSNAME.

Programmer Response: Locate the MSC destination name in the message (MSGMSONM in the MSC prefix). It should be an MSC MSNAME. The LNB control block that represents this MSNAME has a different destination SYSID than the message destination SYSID at MSGMSOID. Most probable cause is the MSNAME destination SYSID has been changed. The LNB control block address is in REG15 in the REG14-12 area and the block is at QTPDST.

X'1040' INSERT - FINDEST ERROR FOR DEST = MSGMSONM

Explanation: The MSC destination name in the message at MSGMSONM could not be found by the FINDEST routine.

Programmer Response: Locate the MSGMSONM name in the message and verify that it is a valid local LTERM, MSNAME, or local or remote TRANSACTION CODE. If it is ETO, it is invoked at system generation and name is a dynamic LTERM, verify that ETO is enabled. FINDEST parameter list used to locate the name is at PSTDCA.

X'1044' INSERT - MSC DEST BLOCK NOT EXPECTED CNT

Explanation: The control block returned by FINDEST, representing the source name at MSGMSONM is not an LTERM CNT.

Programmer Response: Locate the MSGMSONM name in the message prefix and verify it is a valid local LTERM. The CNT control block address returned by FINDEST is in REG1 in the REG14-12 area and the block is at QTPDST.

X'1048' INSERT - MSG DEST NOT EXPECTED TRANSACT

Explanation: The message destination type flag associated with the MSGODSTN name is expected to be an SMB type because the destination control block is an SMB.

Programmer Response: Locate the message destination type flag (MSGDFLG2) of the message and it is a 82. This indicates the MSGODSTN destination name was a CNT type when the original message was created. However, the resource control block returned by FINDEST returned an SMB type control block. Most likely cause is the destination was changed from an LTERM or MSNAME type to a transaction code type. The control block address is in REG1 in the REG14-12 area and the block is at QTPDST. The parameter list passed to FINDEST is in the PSTDCA area.

X'104C' INSERT - DEST SMB SID/DEST MSG SID NOMTCH

Explanation: The message is enqueued to a transaction code SMB and the destination SYSID of the message does not match the destination SYSID of the SMB.

Programmer Response: This error is not currently set.

X'1050' INSERT - DEST CONV BUT NO SPA SEG IN MSG

Explanation: The message destination is an IMS conversational transaction code but the message does not contain a scratch pad (SPA) segment.

Programmer Response: Locate the message destination name in the MSC prefix at MSGMSONM. This name is a conversational transaction code. The SMB address for the transaction code is in REG1 in the REG14-12 area and the SMB block is at QTPDST. The MSG2SPA flag in the MSC prefix should be set on to indicate the message contains a SPA; however, it is not set. Most likely cause is the transaction code was changed from nonconversational to conversational.

X'1054' INSERT - DEST NOT CONV BUT MSG HAS SPASEG

Explanation: The message flag MSG2SPA is set indicating a conversation SPA segment is included in the message and the destination transaction code is not an IMS conversation transaction code.

Programmer Response: Locate the MSG2SPA flag in the MSC prefix of the message and it should be set on. The transaction code is in the MSC prefix at MSGMSONM. REG1 in the REG14-12 area is the SMB address for the transaction code and it is a not an IMS conversational transaction code. The SMB block is at QTPDST. Most likely cause is the transaction code was changed from conversational to nonconversational.

X'1058' INSERT - DEST = BLANKS AT CALL QMGR TIME

Explanation: The destination in the modifiable TPPCB was not set.

Programmer Response: The message queue manager is being called to insert the message to a queue manager buffer and the destination name in the TPCB at TPCBTSYM has not been set. This is an IMS internal error.

X'105C' INSERT - DEST NAME INVALID AT CALLQMGR TIME

Explanation: The destination invalid flag in the TPPCB has not been reset.

Programmer Response: The message queue manager is being called to insert the message to a queue manager buffer and the destination invalid flag (TPCBSMBN) is still set on. This is an IMS internal error.

X'1060' INSERT - NON ZERO RC ON ISRT CALL TO QMGR

Explanation: The message queue manager was called to insert the message to a queue manager buffer and a nonzero return code was returned.

Programmer Response: The queue manager return code is in REG15 of the REG14-12 area. Most likely cause is the message queue buffer is too small to hold the message prefix and segment. Check the large message queue data set block size and determine if it has been reduced from the size when the message was originally created. The length of the message prefix and segment is contained in the first 2 bytes of the message in the I/O area. If the message queue block size is large enough, the message length is correct, and the message queue data sets are not full, then this is probably an IMS internal error.

X'1064' INSERT - MSG CONTAINS INVALID QUEUE NUM

Explanation: The queue number of the message is invalid.

Programmer Response: Locate the message queue number in the message prefix at MSGFLAGS (low order 4 bits of flag). A queue number greater than 5 is invalid. The queue number source will need to be determined. Some rules are:

- If the MRQ recovery mode is RECOVERDM or RECOVERAB and the source of the message is a 4002 DUMPQ or SNAPQ record, the queue number is obtained from the 4002 record by FMQSELCT.

- If the MRQ recovery mode is RECOVERDM or RECOVERAB and the source of the message is a 01 or 03 record, the queue number is obtained from the type 35 enqueue record by FMQSELCT.
- If the MRQ recover mode is REPROCESS, the queue number is 0 in the 01 or 03 record and should have been set by DFSQMRQ0 to 1 if destination is a transaction code or 4 for all other destination types.
- This is either an IMS or MRQ internal error.

X'1068' INSERT - MSGMSINM BLOCK NOT CNT TYPE

Explanation: The control block returned by FINDEST, representing the source name at MSGMSINM is not an LTERM CNT.

Programmer Response: Locate the MSGMSINM name in the message prefix and verify it is a valid local LTERM. The CNT control block address returned by FINDEST is in REG1 in the REG14-12 area and the block is at QTPDST.

X'106C' INSERT - DFSSLC CALL ERR FOR DST MSGMSONM

Explanation: An error was detected when attempting to locate the resource control block for the resource name at MSGMSONM in the message prefix.

Programmer Response: This is most likely an IMS internal error. The return code returned by the locate call is in REG15 of the REG14-12 area. The locate parameter list is in PSTDCA area.

X'1070' INSERT - DFSSLC CALL ERR FOR DST MSGIDSTM

Explanation: An error was detected when attempting to locate the resource control block for the resource name at MSGIDSTN in the message prefix.

Programmer Response: This is most likely an IMS internal error. The return code returned by the locate call is in REG15 of the REG14-12 area. The locate parameter list is in PSTDCA area.

X'1074' INSERT - DFSSLC CALL ERR FOR DST MSGMSINM

Explanation: An error was detected when attempting to locate the resource control block for the resource name at MSGMSINM in the message prefix.

Programmer Response: This is most likely an IMS internal error. The return code returned by the locate call is in REG15 of the REG14-12 area. The locate parameter list is in PSTDCA area.

X'1078' INSERT - DFSSLC CALL ERR FOR DST MSGODSTN

Explanation: An error was detected when attempting to locate the resource control block for the resource name at MSGODSTN in the message prefix.

Programmer Response: This is most likely an IMS internal error. The return code returned by the locate call is in REG15 of the REG14-12 area. The locate parameter list is in PSTDCA area.

X'107C' INSERT - APPC NEEDED BUT NOT SUPPORTED

Explanation: The message was determined to be an LU 6.2 APPC type; however, the APPC message prefix segment was not present or could not be located.

Programmer Response: Locate the message. The MSGC2APP flag should be set on indicating the message is an APPC type. The APPC prefix segment with a segment type flag (MSGSIID) of 85 should be present in the message prefix. This is most likely an IMS internal error.

X'1080' INSERT - MSG DEST = APPC SYNC = NON RECOV

Explanation: Message destination is an LU 6.2 synchronous logical unit (LU) name and is considered nonrecoverable.

Programmer Response: Locate the MSGODSTN name field in the message prefix and it should start with an FDFFFFFF indicating the destination of the message is an LU 6.2 (APPC) logical unit in LU 6.2 synchronous conversation mode. This message is nonrecoverable according to LU 6.2 protocol and is discarded by the MRQ processor (DFSQMRQ0). The LUNAME destination is in the APPC message prefix segment and is extracted and reported in the FMQINSRT messages discarded by destination report. This is a normal condition and is not considered to be an error.

X'1084' INSERT - MSG DEST = NON RECOV

Explanation: Message destination is nonrecoverable either because the destination transaction code name was defined as NORECOV or the message was received from an LU 6.2 LU in synchronous conversation mode (which implies nonrecoverable).

Programmer Response: Locate the MSGFLAGS byte in the message prefix of the message. MSGFNRQU should be set indicating the message is nonrecoverable. Some possible reasons are:

- If the message destination is local (system is not MSC or it is MSC and the destination SYSID at MSGMSOID in the MSC segment item is local) then check to see if destination name at MSGODSTN is a nonrecoverable transaction code.

- If the message destination is remote (system is MSC and the destination SYSID at MSGMSOID in the MSC segment item is remote) then check to see if destination name at MSGMSONM in the MSC prefix segment item is a nonrecoverable transaction code.
- If the source name in the message prefix at MSGIDSTN starts with an FDFFFFFF then the source of the message is an LU 6.2 (APPC) logical unit in LU 6.2 synchronous conversation mode. This message is nonrecoverable according to LU 6.2 protocol. The LUNAME destination is in the APPC message prefix segment and is extracted and reported in the FMQINSRT messages discarded by destination report.

This is a normal condition and is not considered to be an error.

X'1088' INSERT - MSG WAS CANCELED BY IMS

Explanation: The original message was canceled by IMS and was logged for accounting or message queue recovery purposes. The message text itself is not recovered.

Programmer Response: Locate the MSGFLAGS byte in the message prefix and MSGFCANC should be set on indicating the message had been canceled. The MSGODSTN field is the destination name of the canceled message. If MSC is invoked at system generation and an MSC segment item is present and the SYSID at MSGMSOID in the MSC prefix segment item is a remote SYSID, then MSGMSONM in the MSC prefix segment item is the remote destination name. One possible cause is an application program inserted the message and then abended or issued a ROLL or ROLB call. This is a normal condition and is not considered to be an error.

X'108C' INSERT - ERROR LOCATING APPC ASYNC DEST

Explanation: The destination name of the message was determined to be a LU 6.2 (APPC) asynchronous destination and a call to the IMS LU 6.2 interface routine encountered an error locating the LU destination.

Programmer Response: Locate the MSGODSTN destination name in the message prefix and it should start with an FEFFFFFF indicating the destination type is an LU 6.2 (APPC) asynchronous destination. The return code returned by the LU 6.2 interface is in REG15 in the REG14-12 area. The parameter list passed is in the PSTDCA area. The message should contain an LU 6.2 prefix item with a type code of 85 (MSGSIID=85). The LU 6.2 destination name is stored in the LU 6.2 prefix item. Check to see if APPC is correctly installed and enabled and the destination name is a LU 6.2 logical unit. Correct if not. Otherwise, this is most likely an IMS internal error.

X'1090' INSERT - MSGMRQF1 FLAG INVALID

Explanation: The MSGMRQF1 flag in the MRQ prefix passed to the IMS message requeuer processor (DFSQMRQ0) by the MRQ BMP routine (FMQINSRT) is invalid.

Programmer Response: The MSGMRQF1 flag byte is in the MRQ prefix segment (MSGMRQPF) and is in front of the prefix of the message being inserted. The flag byte should be zero or a multiple of X'4'. This is either an IMS or MRQ internal error.

X'1094' INSERT - MSC DEST BLOCK NOT EXPECTED LNB

Explanation: The destination of the message was determined to be an MSC MSNAME resource. However, the destination control block found by FINDEST was not an LNB.

Programmer Response: Locate the message and it should have an MSC prefix segment item with a segment code of 82 (MSGSIID=82) and the destination SYSID in MSGMSOID in the MSC segment item should be remote. MSGODSTN is the MSNAME of the message destination and it should be an LNB control block. REG15 in the REG14-12 area is the address of the expected LNB and the LNB is at QTPDST. Most likely cause is the destination MSNAME was changed to an LTERM name or transaction code.

X'1098' INSERT - SOURCE/DEST = DFSAPPC INVALID

Explanation: Destination name of DFSAPPC is invalid.

Programmer Response: This error is currently not being set.

X'109C' INSERT - LU6.2 SCD EXTEN INVALID/NOTAVAIL

Explanation: The message was determined to be an LU 6.2 (APPC) type. However, the APPC SCD extension could not be located.

Programmer Response: Locate the message and MSGCFLG2 byte of the message prefix segment should be set on indicating an LU 6.2 segment is present (MSGC2APP is set on), or the destination name at MSGODSTN or MSGMSONM is DFSAPPC. Field SCDLSCD in the SCD was zero. This is either an IMS internal error or APPC is not correctly installed.

X'10A0' INSERT - MSG NOT VALID 01/03 TYPE

Explanation: The message being passed by FMQINSRT is not a valid type 01 or 03 message.

Programmer Response: Locate the message and verify the MSGLCODE byte is either a 01 or a 03, and the message prefix includes at least a basic segment prefix item (first hex 14 bytes) and a system segment prefix item (prefix segment item following the basic prefix segment, MSGSIID = 81), and the MSGDFLG2 flag byte is either an 81 (transaction code type destination), or a 82 (LTERM, MSNAME, or USERID type of destination). This is most likely an IMS or MRQ internal error. The original message input to FMQSELCT should be located and examined.

X'10A4' INSERT - INTERNAL IMS MESSAGE

Explanation: The message being passed by FMQINSRT is an internal IMS message that is not recoverable.

Programmer Response: Locate the message in the I/O area and verify the destination name at MSGODSTN or MSGMSONM is an internal IMS destination. Current internal destination messages are: MSVERIFY system LNB. MSGODSTN/MSGOMSNM begins with the characters MSN and the destination control block at QTPDST is a system LNB (CNT3QSYS flag is set on). REG15 or REG1 in the REG14-12 area is the address of the LNB. This is normal and is not considered to be an error.

X'10A8' INSERT - SOURCE/DEST NAME CHANGED

Explanation: The name in the control block representing the source name of the message (LTERM name) or the destination name of the message (LTERM or TRANCODE name) does not match the name in the message.

Programmer Response: The control block representing either the source LTERM or destination LTERM or TRANCODE is pointed to by register 14 in the register save area. The message is in the I/O area and is also pointed to by register 6. The name in the control block at offset X'1C' does not match either the source field (MSGIDSTN) or destination field (MSGODSTN) of the message. This is an internal IMS failure.

X'10AC' INSERT - DFSLUMIF BLDPRE ERROR

Explanation: A nonzero return code was returned by the IMS APPC LUM services routine while trying to build a new APPC prefix for an APPC message.

Programmer Response: The APPC message being processed is in the I/O area and is also pointed to by register 6 in the register save area. The nonzero return code from the LUM services routine is in register 15. This is an internal IMS failure.

X'10B0' INIT - ERROR GETTING DFSPOOL STORAGE

Explanation: A DFSPOOL call received a nonzero return code attempting to get storage from the HIOP storage pool for the QMRQWORK area.

Programmer Response: Register 15 contains the return code from the DFSPOOL call. This is either an internal error, or there is not enough storage in the IMS control region private area.

X'10B4' INIT - ERROR GETTING AN AWE

Explanation: A DFSBCB GET for an AWE block received a nonzero return code.

Programmer Response: Register 15 contains the return code from the DFSBCB GET call. This is either an internal error, or there is not enough storage in the IMS control region private area.

X'10B8' INSERT - NO EXTENDED PREFIX PRESENT

Explanation: The message being requeued was expected to contain an extended prefix segment (MSGC2EPH=1), but none existed (QMRWEPHP=0).

Programmer Response: Analyze the message and its prefix segments. The address of QMRQWORK is in register 5; the message address is in register 6. If the message being processed is from IMS release 510 or a later release, this prefix segment should exist. If it is from a release earlier than 510, this prefix segment should not exist. This is most likely an IMS internal error.

X'10BC' INIT - ERROR INIT/ADDRESSING QMRQWORK

Explanation: An error occurred while getting the QMRQWORK area and initializing it with the current message information.

Programmer Response: Look for a previous type X'6701'-MRQE error record that indicates another more specific error. This error is logged when the caller (INSERT) receives control back from QMRQINIT and register 15 is nonzero. QMRQINIT logs a X'6701'-MRQE record when the specific error is detected.

X'10C0' INIT - CAN'T FIND RACF SEGMENT MSGSORAC

Explanation: The message flag indicates a RACF prefix segment is present, but the segment cannot be located.

Programmer Response: Locate the message and verify that flag MSGxxxx is set. If set, a RACF prefix segment with a code of X'83' must be present. This is an internal IMS error.

X'10C4' **INIT - CAN'T FIND LU6.1 SEGMENT MSGSILU6**

Explanation: The message flag indicates an LU6.1 prefix segment is present, but the segment cannot be located.

Programmer Response: Locate the message and verify that flag MSGxxxx is set. If set, an LU6.1 prefix segment with a code of X'84' must be present. This is an internal IMS error.

X'10C8' **INIT - CAN'T FIND APPC SEGMENT MSGSOAP0**

Explanation: The message flag indicates an APPC prefix segment is present, but the segment cannot be located.

Programmer Response: Locate the message and verify that flag MSGxxxx is set. If set, an APPC prefix segment with a code of X'85' must be present. This is an internal IMS error.

X'10CC' **INIT - CAN'T FIND EPH SEGMENT MSGSIEPH**

Explanation: The message flag indicates an EPH prefix segment is present, but the segment cannot be located.

Programmer Response: Locate the message and verify that flag MSGxxxx is set. If set, an EPH prefix segment with a code of X'86' must be present. This is an internal IMS error.

X'10D0' **INIT - CAN'T FIND APPC SEGMENT MSGSIAP0**

Explanation: The message flag indicates an APPC prefix segment is present, but the segment cannot be located.

Programmer Response: Locate the message and verify that flag MSGxxxx is set. If set, an APPC prefix segment with a code of X'87' must be present. This is an internal IMS error.

X'10D4' **INIT - CAN'T FIND SEC SEGMENT MSGSISEC**

Explanation: The message flag indicates a SEC prefix segment is present, but the segment cannot be located.

Programmer Response: Locate the message and verify that flag MSGxxxx is set. If set, a SEC prefix segment with a code of X'88' must be present. This is an internal IMS error.

X'10D8' **INIT - CAN'T FIND WLM SEGMENT MSGSIWLM**

Explanation: The message flag indicates a WLM prefix segment is present, but the segment cannot be located.

Programmer Response: Locate the message and verify that flag MSGxxxx is set. If set, a WLM prefix segment with a code of X'88' must be present. This is an internal IMS error.

X'10DC' **INIT - CAN'T FIND SYS EXT SEGMENT MSGSISEX**

Explanation: The message flag indicates a SYS EXT prefix segment is present, but the segment cannot be located.

Programmer Response: Locate the message and verify that flag MSGxxxx is set. If set, a SYS EXT prefix segment with a code of X'88' must be present. This is an internal IMS error.

X'10E0' **INIT - CAN'T FIND MSC EXT SEGMENT MSGSIMEX**

Explanation: The message flag indicates an MSC EXT prefix segment is present, but the segment cannot be located.

Programmer Response: Locate the message and verify that flag MSGxxxx is set. If set, an MSC EXT prefix segment with a code of X'88' must be present. This is an internal IMS error.

X'10E4' **ISRT - OTMA MESSAGES NOT SUPPORTED**

Explanation: The IMS release message that is being requeued either does not support OTMA messages, or the OTMA feature is not defined.

Programmer Response: Locate flag MSGFLAGA in the QMRQWORK area to determine the release of the IMS systems that are the source and destination of the message. The IMS release must be 510 or a later release.

X'10E8' **ISRT - MSC/APPC MESSAGE NOT SUPPORTED**

Explanation: The message is a remote MSC message that originated from an APPC LU6.2 session and is not supported on this release.

Programmer Response: Locate flag QMRWFLGA in the QMRQWORK area and determine the release of the IMS system that is the destination of the message. It must be release 510 or a later release. The destination SID in the message prefix (message prefix pointed to by register 6) is remote, as indicated by QMRWFLG6 in the QMRQWORK area. The problem is probably caused by the destination of the message changing from local to remote, or by requeuing a MSC/APPC message from an IMS release that is 510 or a later release. The IMS release originating the message is also set in QMRWLAGA. The address of QMRQWORK is in register 5.

X'10EC' ISRT - MESSAGE REROUT NOT SUPPORTED

Explanation: DFSQMRQ0 is being called with a reroute function that is not supported in this IMS release.

Programmer Response: This is an internal IMS error. Trace back to the caller of DFSQMRQ0.

X'10F0' ISRT - MSC SEG ITEM NOT PRESENT

Explanation: The destination is a remote transaction, but the message does not have an MSC segment item.

Programmer Response: The transaction changed from local to remote after the original message was built.

X'2000' PURGE - PURGE PCB NOT MODIFIABLE

Explanation: Alternate PCB defined in MRQ PSB is not modifiable type.

Programmer Response: Verify that MODIFY=YES was coded on the PCB named ALTPCB01 for the MRQPSB.

MRQPSB is the default MRQ PSBNAME and may have been changed on the MRQPSBN= parameter of the MSGQUEUE macro at system generation.

X'2004' PURGE - PURGE PCB DEST INVALID

Explanation: The message is being purged (enqueued to a temporary destination) and the temporary destination name has not been set to valid.

Programmer Response: The destination invalid flag (TPCBSMBN) in flag byte TPCBCODE is set on. This flag should have been reset during insert processing. If a queue manager buffer (QMBA) is allocated, the message being processed should be in this buffer. Otherwise, the message might have to be located on the SCRAPLOG data set where it is discarded by FMQINSRT. The time stamp (date/time) of the message being processed is stored in the PST at PSTPRE1 and can be used to locate the message on the SCRAPLOG or the original message input to FMQSELCT. This is an internal IMS or MRQ error.

X'2008' PURGE - PURGE PCB DEST SET TO BLANKS

Explanation: The message is being purged (enqueued to a temporary destination) and the temporary destination name is blanks.

Programmer Response: The destination name in the TPCB at TPCBTSYM is blanks (hex 40s). This field should have been set to the destination name of the message during insert processing. If a queue manager buffer (QMBA) is allocated, the message being processed should be in this buffer. Otherwise, the message might have to be located on the SCRAPLOG data set where it is discarded by FMQINSRT. The time stamp (date/time) of the message being processed is stored in the PST at PSTPRE1 and can be used to locate the message on the SCRAPLOG or the original message input to FMQSELCT. This is an internal IMS or MRQ error.

X'200C' PURGE - PURGE DEST CTL BLK ADDR ZERO

Explanation: The message is being purged (enqueued to a temporary destination) and the temporary destination control block address in the TPPCB is zero.

Programmer Response: The destination name control block address is in the TPCB at TPCBCNT and is referred to as the QTPDST address. This field should have been set to the address of destination name control block (address of either the CNT, LNB, or SMB) during insert processing. If a queue manager buffer (QMBA) is allocated, the message being processed should be in this buffer. Otherwise, the message may have to be located on the SCRAPLOG data set where it is discarded by FMQINSRT. The time stamp (date/time) of the message being processed is stored in the PST at PSTPRE1 and can be used to locate the message on the SCRAPLOG or the original message input to FMQSELCT. This is an internal IMS or MRQ error.

X'2010' PURGE - PURGE DEST NAME = DFS INVALID

Explanation: The message is being purged (enqueued to a temporary destination) and the temporary destination name of the message starts with the reserved characters DFS.

Programmer Response: The destination name in the TPCB at TPCBTSYM starts with the characters DFS and is not a DFSAPPC destination message or other internal IMS destination. This is invalid. If a queue manager buffer (QMBA) is allocated, the message being processed should be in this buffer. Otherwise the message may have to be located on the SCRAPLOG data set where it is discarded by FMQINSRT. The time stamp (date/time) of the message being processed is stored in the PST at PSTPRE1 and can be used to locate the message on the SCRAPLOG or the original message input to FMQSELCT. This is most likely an internal IMS error.

X'2014' PURGE - PURGE INQUIRY DEST NOT SIGNED ON

Explanation: The message is being purged (enqueued to a temporary destination) and the temporary destination name of the message is an inquiry type LTERM.

Programmer Response: The destination name in the TPCBTSYM is an inquiry type LTERM destination and is not signed on. The destination control block CNT is in REG6 in the REG14-12 area and the CNT2INQ flag is set on (destination is inquiry type). The CNT control block is at QTPDST. The CTB is in REG7 of the REG14-12 area and CTB1DIAL and CTB1SIGN are set off (terminal is not signed on).

Messages destined to an inquiry LTERM that is not signed on are discarded according to protocol. This is considered to be normal operation.

X'2018' PURGE - PURGE NON 0 RC ON QMGR ENQ CALL

Explanation: The message is being purged (enqueued to a temporary destination) and a nonzero return code was received from the message queue manager on the enqueue call.

Programmer Response: The message queue manager return code is in REG15 of the REG14-12 area. The message queue buffer is in the QMBA area. This is most likely an internal IMS error.

X'201C' PURGE - PURGE I/O AREA INVALID

Explanation: The I/O area passed to the IMS MRQ processor by FMQINSRT on the PURG call is invalid.

Programmer Response: The I/O area passed on the PURG call does not begin with a valid MRQ prefix segment (MSGMRQPF). This is an internal MRQ FMQINSRT error.

X'2020' PURGE - PURGE MSGMRQF1 FLAG INVALID

Explanation: The MSGMRQF1 flag in the MRQ prefix passed to the IMS message requeuer processor (DFSQMRQ0) by the MRQ BMP routine (FMQINSRT) is invalid.

Programmer Response: The MSGMRQF1 flag byte is in the MRQ prefix segment (MSGMRQPF). MSGMRQPF segment starts at the beginning of the I/O area. The flag byte should be a multiple of X'4'. This is either an IMS or MRQ internal error.

X'2024' PURGE - DEST BLK=DFSAPPC BUT MSG NOT APPC

Explanation: The message is being purged (enqueued to a temporary destination) and the destination name is DFSAPPC. However, the destination resource type is not an LU 6.2 (APPC) destination.

Programmer Response: The resource name control block in REG6 in the REG14-12 area contains a name of DFSAPPC but the resource type flag in the TPPCB at flag byte TPPCBFLG was not set to type = APPC (TPPCB62 is not set on). The DFSAPPC CNT block is at QTPDST. This is an internal IMS error.

X'3000' SETPRFX - MESSAGE PREFIX SIZE INVALID

Explanation: Either the total prefix or one or more of the prefix segments has an invalid length.

Programmer Response: Locate the message being inserted in the I/O area. The segment address is in REG1 of the REG14-12 area. The total prefix size is at offset 10 in the message. The current prefix segment address of the prefix segment being checked is in REG7 of the REG14-12 area. The prefix segment length is in the first 2 bytes. The prefix ID (MSGSIID) is in the third byte. Locate this ID in the QLOGMSG DSECT and verify the size.

If the message is from a supported IMS release, this is probably an internal IMS error.

X'3004' SETPRFX ERROR REASON CODE

Explanation: Reserved for future use.

X'3008' SETPRFX ERROR REASON CODE

Explanation: Reserved for future use.

X'4000' CPYPRFX - PREFIX SIZE NOT SIZE EXPECT

Explanation: The message queue manager failed to obtain a message prefix the same size as that of the original message.

Programmer Response: Locate the message being inserted in the I/O area. Field MSGPRFLL in the message prefix is the length of the original message prefix. Field QSAPPLTH in the QSAPWKAD area contains the length of the new message prefix. They should be equal. This is an internal IMS error.

X'4004' CPYPRFX - CAN'T FIND MSC SEGMENT MSGSIPEX

Explanation: Message flag indicates MSC prefix segment is present but segment cannot be located.

Programmer Response: Locate the message and verify the flag MSGC2MSC is set. If set, then MSC prefix segment with a code=82 must be present. REG1 in the REG14-12 area is the address of the prefix being copied. This is an internal IMS error.

X'4008' CPYPRFX ERROR REASON CODE

Explanation: Reserved for future use.

X'400C' CPYPRFX ERROR REASON CODE

Explanation: Reserved for future use.

X'5000' CANCEL - NON ZERO RC ON CANCEL CALL TO QMGR

Explanation: A nonzero return code was returned by the message queue manager while attempting to cancel a message queue buffer that is being discarded (message is being scrapped).

Programmer Response: An error was detected while inserting a message to the message queue and cleanup processing is being performed. The original error has already been logged in a prior type 6701-MRQE log record and the queue buffer area is being released (canceled). The queue manager return code on the cancel call is in REG15 of the REG14-12 area. This is an internal IMS error.

X'5004' CANCEL ERROR REASON CODE

Explanation: Reserved for future use.

X'5008' CANCEL ERROR REASON CODE

Explanation: Reserved for future use.

X'500C' CANCEL ERROR REASON CODE

Explanation: Reserved for future use.

X'6000' LOGIC ERROR REASON CODE

Explanation: Reserved for future use.

X'6004' FMQINSRT - LOGREC TYPE NOT 4002, 01, OR 03

Explanation: The FMQINSRT BMP program read a log record that was not a valid type 4002 (DUMPQ or SNAPQ), 01 (input), or 03 (output) record, and discarded the record to the SCRAPLOG data set.

Programmer Response: This error is detected by the FMQINSRT routine and is passed to the message requeuer processor to perform cleanup and log the error in a 6701-MRQE record. The SCRAPLOG record written by FMQINSRT will need to be located to determine its validity. The record may need to be traced back to the log data set input to FMQSELCT. The QMBA area may contain part or all of the message being inserted when the invalid record was detected. This is either an IMS or MRQ internal error.

X'6008' FMQINSRT - NO SECONDARY LOGREC WHEN EXPECTED

Explanation: A message was being inserted that spanned multiple message queue buffers and one of the secondary buffers could not be located.

Programmer Response: This error is detected by the FMQINSRT routine and is passed to the message requeuer processor to perform cleanup and log the error in a 6701-MRQE record. The SCRAPLOG record written by FMQINSRT needs to be located to reconstruct the chain of message buffers. The record may need to be traced back to the log data set input to FMQSELCT. The QMBA area may contain part or all of the message being inserted. This is either an IMS or MRQ internal error.

X'600C' FMQINSRT - SECONDARY LOGREC DEST INVALID

Explanation: A message was being inserted that spanned multiple message queue buffers and one of the secondary buffers in the chain being processed by FMQINSRT did not have the same destination name.

Programmer Response: This error is detected by the FMQINSRT routine and is passed to the message requeuer processor to perform cleanup and log the error in a 6701-MRQE record. The SCRAPLOG record written by FMQINSRT will need to be located to determine its validity and reconstruct the message buffer chain. The record may need to be traced back to the log data set input to FMQSELCT. This is either an IMS or MRQ internal error.

X'6010' FMQINSRT ERROR REASON CODE

Explanation: Reserved for future use.

X'6014' FMQINSRT ERROR REASON CODE

Explanation: Reserved for future use.

X'6018' FMQINSRT ERROR REASON CODE

Explanation: Reserved for future use.

X'7004' XLATPFX - CAN'T FIND SYS EXT SEGMENT MSGSISEX

Explanation: The message flag indicates that the system EXT prefix segment is present, but the segment cannot be located.

Programmer Response: Locate the message and verify that flag MSGESEX is set. If set, an MSC EXT prefix segment with a code of X'8A' must be present. The message being built that caused the error is pointed to by register 6. This is an internal IMS error.

Appendix C. Module-to-Function-to-Subfunction List

For an explanation of the functions and subfunctions, see “IMS Functions and Subfunctions” on page 517.

Modules with the identification of DSP apply to IMS Database Recovery Control.

Modules with the identification of DXR apply to the Internal Resource Lock Manager (IRLM).

Module	Function	Subfunction
DBFAALD0	FP	DIAG
DBFABAL0	FP	DIAG
DBFACDI0	FP	DIAG
DBFACNT0	FP	DIAG
DBFADCC0	FP	DIAG
DBFADMA0	FP	DIAG
DBFADMC0	FP	DIAG
DBFADMH0	FP	DIAG
DBFADUMP	FP	DIAG
DBFAEMH0	FP	DIAG
DBFAESR0	FP	DIAG
DBFAHSD0	FP	DIAG
DBFAHSO0	FP	DIAG
DBFAHSR0	FP	DIAG
DBFAIDS0	FP	DIAG
DBFALOC0	FP	DIAG
DBFAMRM0	FP	DIAG
DBFAMSD0	FP	DIAG
DBFAPCB0	FP	DIAG
DBFAPSC0	FP	DIAG
DBFAPST0	FP	DIAG
DBFARCT0	FP	DIAG
DBFARDA0	FP	CMD
DBFARDB0	FP	CMD
DBFARDC0	FP	CMD
DBFARD10	FP	CMD
DBFARD20	FP	CMD
DBFARD30	FP	CMD
DBFARD40	FP	CMD
DBFARD50	FP	CMD
DBFASCD0	FP	DIAG
DBFASRB0	FP	DIAG
DBFATRM0	SYS	INIT
DBFAUXR0	FP	DIAG
DBFAXCR0	FP	DIAG
DBFBADR0	FP	INIT
DBFBBIN0	FP	MSDB
DBFBCHG0	FP	MSDB

Module	Function	Subfunction
DBFBCL10	FP	MSDB
DBFBCNT0	FP	EMH
DBFBDLT0	FP	MSDB
DBFBENQ0	FP	MSDB
DBFBFLD0	FP	MSDB
DBFBGET0	FP	MSDB
DBFBINC0	FP	MSDB
DBFBNUB0	FP	CNTRL
DBFBNXT0	FP	MSDB
DBFBRPL0	FP	MSDB
DBFBSEQ0	FP	MSDB
DBFBSRT0	FP	MSDB
DBFBUPB0	FP	MSDB
DBFBVAL0	FP	MSDB
DBFBVFY0	FP	MSDB
DBFBXTR0	FP	MSDB
DBFCARP0	FP	SHRDQ
DBFCBHL0	FP	CNTRL
DBFCDAR0	FP	CMD
DBFCDDA0	FP	CMD
DBFCDDB0	FP	CMD
DBFCDPL0	FP	CMD
DBFCDPS0	FP	CMD
DBFCDQB0	FP	CMD
DBFCDRC0	FP	CMD
DBFCDSR0	FP	CMD
DBFCDVS0		
DBFCEMH0	FP	CKPT
DBFCGAB0	FP	DEDB
DBFCHKP0	FP	CKPT
DBFCHK10	FP	CKPT
DBFCHK20	FP	CKPT
DBFCHK30	FP	CKPT
DBFCMP00	FP	DEDB
DBFCMP10	FP	DEDB
DBFCPID0	FP	CKPT
DBFCPRC0	FP	CMD
DBFCPY00	FP	CNTRL
DBFCQR10	FP	SHRDQ
DBFCSTS0	FP	CNTRL
DBFCST00	FP	CNTRL
DBFDBAC0	FP	CMD
DBFDBAU0	FP	DEDB
DBFDBDL0	FP	MSDB
DBFDBDP0	FP	MSDB
DBFDBDR0	FP	UTIL

Module	Function	Subfunction
DBFDBDS0	FP	UTIL
DBFDBDT0	FP	UTIL
DBFDBDU0	FP	MSDB
DBFDBDW0	FP	UTIL
DBFDBDZ0	FP	UTIL
DBFDBFM0	FP	MSDB
DBFDBF00	FP	CNTRL
DBFDBF10	FP	CNTRL
DBFDBIF0	FP	MSDB
DBFDBIL0	FP	MSDB
DBFDBLP0	FP	UTIL
DBFDBLS0	FP	RSTRT
DBFDBMA0	FP	UTIL
DBFDBMB0	FP	UTIL
DBFDBMC0	FP	UTIL
DBFDBMD0	FP	UTIL
DBFDBME0	FP	UTIL
DBFDBMF0	FP	UTIL
DBFDBMG0	FP	UTIL
DBFDBMH0	FP	UTIL
DBFDBMK0	FP	UTIL
DBFDBML0	FP	UTIL
DBFDBMM0	FP	UTIL
DBFDBMN0	FP	UTIL
DBFDBMP0	FP	UTIL
DBFDBMQ0	FP	UTIL
DBFDBMR0	FP	UTIL
DBFDBMV0	FP	UTIL
DBFDBMX0	FP	UTIL
DBFDBPV0	FP	DEDB
DBFDBTC0	FP	MSDB
DBFDBUN0	FP	RSTRT
DBFDCADD	FP	DEDB
DBFDCAP0	FP	DEDB
DBFDCREM	FP	DEDB
DBFDEBSC	FP	DEDB
DBFDEBUG	FP	DEDB
DBFDEDB0	FP	DEDB
DBFDEVT0	FP	DEDB
DBFDIDT0	FP	CNTRL
DBFDLA30	FP	CNTRL
DBFDLB00	FP	UTIL
DBFDLG20	FP	LOG
DBFDLOG0	FP	LOG
DBFDLSR0	FP	RSTRT
DBFDRIS0	FP	CNTRL

Module	Function	Subfunction
DBFDRSC0	FP	CNTRL
DBFDSRP0	FP	CNTRL
DBFDTCR0	FP	DEDB
DBFDTXO0	FP	LOCK
DBFDVBI0	FP	INIT
DBFEACL0	FP	TKO
DBFEAIS0	FP	TKO
DBFECLS0	FP	TKO
DBFEHSH0	FP	TKO
DBFELOCK	FP	LOCK
DBFEMH00	FP	EMH
DBFEPSB0	FP	INIT
DBFERAU0	FP	RSTRT
DBFERCF0	FP	RSTRT
DBFERDB0	FP	RSTRT
DBFERMG0	FP	RSTRT
DBFERMSA	FP	RSTRT
DBFEROC0	FP	RSTRT
DBFERST0	FP	RSTRT
DBFERSY0	FP	RSTRT
DBFERSY1	FP	RSTRT
DBFERS10	FP	RSTRT
DBFERS20	FP	RSTRT
DBFERS30	FP	RSTRT
DBFE2CI0	FP	RSTRT
DBFFATC0	FP	CNTRL
DBFFATW	FP	CNTRL
DBFFCNT0	FP	EMH
DBFFEMH0	FP	EMH
DBFFENT0	FP	EMH
DBFFFP00	FP	CNTRL
DBFFORH0	FP	I/O
DBFFORI0	FP	I/O
DBFFPPR0	FP	CNTRL
DBFHAGU0	FP	EMH
DBFHBDS0	FP	TKO
DBFHCHG0	FP	EMH
DBFHCIR0	FP	TKO
DBFHCL00	FP	EMH
DBFHCTK0	FP	TKO
DBFHDC40	FP	DEDB
DBFHDC44	FP	DEDB
DBFHDEP0	FP	TKO
DBFHDMP0	FP	MSDB
DBFHEMH0	FP	EMH
DBFHGN00	FP	EMH

Module	Function	Subfunction
DBFHGU10	FP	EMH
DBFHIEL0	FP	EMH
DBFHINI0	FP	INIT
DBFHLOD0	FP	EMH
DBFHQMI0	FP	EMH
DBFHRLB0	FP	EMH
DBFHRTR0	FP	EMH
DBFHSRT0	FP	EMH
DBFHTMG0	FP	CNTRL
DBFIBTS0	FP	RSTRT
DBFIBUF0	FP	CNTRL
DBFICIR0	FP	INIT
DBFICI10	FP	INIT
DBFICLI0	FP	INIT
DBFICLJ0	FP	CMD
DBFICL20	FP	MSDB
DBFICL40	DC	CMD
DBFIFIX0	FP	INIT
DBFIIN30	FP	CNTRL
DBFILQS0	FP	SHRDQ
DBFINI10	FP	INIT
DBFINI20	FP	INIT
DBFINI30	FP	INIT
DBFINI40	FP	INIT
DBFINTE0	FP	INIT
DBFINTP0	FP	INIT
DBFINTS0	FP	INIT
DBFIPQS0	FP	SHRDQ
DBFIRC10	FP	CNTRL
DBFISRB0	FP	INIT
DBFLHCK0	FP	LOCK
DBFLHSH0	FP	CNTRL
DBFLINK2	FP	UTIL
DBFLIRL0	FP	LOCK
DBFLRH00	FP	CNTRL
DBFLRLS0	FP	CNTRL
DBFMADR0	FP	DEDB
DBFMBED0	FP	CNTRL
DBFMBFL9	FP	DEDB
DBFMBMM9	FP	DEDB
DBFMCCV9	FP	DEDB
DBFMCLBS	FP	DEDB
DBFMCLES	FP	DEDB
DBFMCLX0	FP	DEDB
DBFMCRP9	FP	DEDB
DBFMCSS9	FP	DEDB

Module	Function	Subfunction
DBFMCTL0	FP	DEDB
DBFMDA00	FP	CNTRL
DBFMDBQ0	FP	DEDB
DBFMDIE0	FP	DEDB
DBFMDLT0	FP	DEDB
DBFMDPT9	FP	DEDB
DBFMDRA9	FP	DEDB
DBFMDRB0	FP	DEDB
DBFMDRX0	FP	DEDB
DBFMDSG9	FP	DEDB
DBFMEQE0	FP	DEDB
DBFMERE0	FP	DEDB
DBFMER00	FP	DEDB
DBFMFLG0	FP	DEDB
DBFMFSE0	FP	DEDB
DBFMGAP0	FP	DEDB
DBFMGFD0	FP	DEDB
DBFMGLA9	FP	DEDB
DBFMGNR0	FP	DEDB
DBFMGNX0	FP	DEDB
DBFMGPD0	FP	DEDB
DBFMGPF0	FP	DEDB
DBFMGRF0	FP	DEDB
DBFMGUX0	FP	DEDB
DBFMGXC0	FP	CNTRL
DBFMHEX0	FP	DEDB
DBFMIOE0	FP	I/O
DBFMIOS0	FP	I/O
DBFMIRC9	FP	DEDB
DBFMIRT0	FP	DEDB
DBFMISL9	FP	DEDB
DBFMLCL0	FP	DEDB
DBFMLEV0	FP	DEDB
DBFMLOG0	FP	DEDB
DBFMLOP0	FP	I/O
DBFMLTE2	FP	DEDB
DBFMMIT0	FP	DEDB
DBFMOCI0	FP	DEDB
DBFMOCL0	FP	DEDB
DBFMOPE0	FP	DEDB
DBFMOPR0	FP	DEDB
DBFMOVE0	FP	DEDB
DBFMPCC9	FP	DEDB
DBFMPCL0	FP	DEDB
DBFMPED9	FP	DEDB
DBFMPEI9	FP	DEDB

Module	Function	Subfunction
DBFMPER9	FP	DEDB
DBFMPGO0	FP	DEDB
DBFMPIO9	FP	DEDB
DBFMPIRS	FP	DEDB
DBFMPOP0	FP	I/O
DBFMPOS0	FP	DEDB
DBFMPSG9	FP	DEDB
DBFMPSI9	FP	DEDB
DBFMPUG0	FP	DEDB
DBFMRA1S	FP	DEDB
DBFMRA2S	FP	DEDB
DBFMRBU0	FP	DEDB
DBFMRCCS	FP	DEDB
DBFMRCPS	FP	DEDB
DBFMRCU0	FP	DEDB
DBFMRDCS	FP	DEDB
DBFMRDDS	FP	DEDB
DBFMRDPS	FP	DEDB
DBFMRDTS	FP	DEDB
DBFMRPU0	FP	DEDB
DBFMRPX0	FP	DEDB
DBFMRQC0	FP	DEDB
DBFMRUC0	FP	DEDB
DBFMSDBT	FP	MSDB
DBFMSDBW	FP	MSDB
DBFMSDB0	FP	MSDB
DBFMSDP0	FP	DEDB
DBFMSDSN	FP	MSDB
DBFMSEG0	FP	DEDB
DBFMSERS	FP	DEDB
DBFMSFI9	FP	DEDB
DBFMSFO9	FP	DEDB
DBFMSGA0	FP	DEDB
DBFMSIM9	FP	DEDB
DBFMSPC0	FP	DEDB
DBFMSRB0	FP	DEDB
DBFMSRH0	FP	DEDB
DBFMSRT0	FP	DEDB
DBFMSSA9	FP	DEDB
DBFMSSC9	FP	DEDB
DBFMSSD9	FP	DEDB
DBFMSSG9	FP	DEDB
DBFMSSI9	FP	DEDB
DBFMSSP9	FP	DEDB
DBFMSSR9	FP	DEDB
DBFMSTP0	FP	DEDB

Module	Function	Subfunction
DBFMSVC9	FP	DEDB
DBFMTME0	FP	DEDB
DBFMUHE0	FP	DEDB
DBFMUHE1	FP	DEDB
DBFMUTR0	FP	RSTRT
DBFMVAPS	FP	DEDB
DBFMVSN9	FP	DEDB
DBFMWTO0	FP	CNTRL
DBFNALC0	FP	DEDB
DBFNCBS0	FP	LOCK
DBFNDC00	FP	RSTRT
DBFNEQE0	FP	DEDB
DBFNOTM0	FP	LOCK
DBFNOTX0	FP	LOCK
DBFNRST0	FP	RSTRT
DBFNRS10	FP	RSTRT
DBFNRS20	FP	RSTRT
DBFPADR0	FP	CNTRL
DBFPALC0	FP	I/O
DBFPAPB0	FP	DEDB
DBFPARDL	FP	DEDB
DBFPCAA0	FP	INIT
DBFPCHM0	FP	LOCK
DBFPDHS0	FP	CMD
DBFPDNA0	FP	DEDB
DBFPEAT0	FP	CNTRL
DBFPENQ0	FP	LOCK
DBFPFAB0	FP	DEDB
DBFPFDS0	FP	I/O
DBFPFPB0	FP	DEDB
DBFPGAB0	FP	DEDB
DBFPGAP0	FP	DEDB
DBFPGDS0	FP	I/O
DBFPHI00	FP	INIT
DBFPHI10	FP	INIT
DBFPHI20	FP	INIT
DBFPHI30	FP	INIT
DBFPHI40	FP	INIT
DBFPHST0	FP	CNTRL
DBFPICE0	FP	DEDB
DBFPICS0	FP	DEDB
DBFPICT0	FP	CNTRL
DBFPIEX0	FP	CNTRL
DBFPIOS0	FP	I/O
DBFPMSG0	FP	INIT
DBFPRAB0	FP	DEDB

Module	Function	Subfunction
DBFPSET0	FP	DEDB
DBFPTIC0	FP	CNTRL
DBFPULI0	FP	LOCK
DBFPUXC0	FP	LOCK
DBFPUXR0	FP	LOCK
DBFPVTS0	FP	DEDB
DBFRESX0	FP	CNTRL
DBFRMRC0	FP	CMD
DBFSADR0	FP	DEDB
DBFSAMA1	FP	IVP
DBFSAMA2	FP	IVP
DBFSAMA3	FP	IVP
DBFSAMD1	FP	IVP
DBFSAMD2	FP	IVP
DBFSAMD3	FP	IVP
DBFSAMD4	FP	IVP
DBFSAMF1	FP	IVP
DBFSAMP1	FP	IVP
DBFSAMP2	FP	IVP
DBFSAMP3	FP	IVP
DBFSAMP4	FP	IVP
DBFSBLK0	FP	CNTRL
DBFSBP10	FP	MSDB
DBFSDEQ0	FP	CNTRL
DBFSEQS0	FP	SHRDQ
DBFSEVT0	FP	SHRDQ
DBFSGAB0	FP	DEDB
DBFSFAB0	FP	DEDB
DBFSHDQ0	FP	DEDB
DBFSHSP0	FP	EMH
DBFSIC10	FP	DEDB
DBFSINF0	FP	SHRDQ
DBFSLEEP	FP	CNTRL
DBFSLGE0	FP	LOG
DBFSLGE1	FP	LOG
DBFSLGE2	FP	LOG
DBFSLG20	FP	LOG
DBFSLM62	FP	EMH
DBFSLOG0	FP	LOG
DBFSMP10	FP	DEDB
DBFSPIX0	FP	DEDB
DBFSQ030	FP	SHRDQ
DBFSTAP0	FP	RSTRT
DBFSUSX0	FP	CNTRL
DBFSYN00	FP	CNTRL
DBFSYN10	FP	CNTRL

Module	Function	Subfunction
DBFSYN20	FP	CNTRL
DBFSYP20	FP	CNTRL
DBFTAFC9	FP	DIAG
DBFTATC9	FP	DIAG
DBFTBIS9	FP	DIAG
DBFTBLT9	FP	DIAG
DBFTBMIS	FP	DIAG
DBFTBMI9	FP	DIAG
DBFTCMT9	FP	DIAG
DBFTCOT9	FP	DIAG
DBFTCTLU	FP	CNTRL
DBFTDEB9	FP	DIAG
DBFTDERS	FP	DIAG
DBFTDRT9	FP	DIAG
DBFTERM0	FP	INIT
DBFTFTO9	FP	DIAG
DBFTIR1S	FP	DIAG
DBFTOCH0	FP	TKO
DBFTOFN0	FP	TKO
DBFTOPU0	FP	TKO
DBFTORS0	FP	TKO
DBFTRAB9	FP	DIAG
DBFTRACE	FP	DIAG
DBFTRACI	FP	DIAG
DBFTRAK0	FP	MSDB
DBFTRCC9	FP	DIAG
DBFTRCO9	FP	DIAG
DBFTRIN9	FP	DIAG
DBFTRLG9	FP	DIAG
DBFTROC0	FP	DIAG
DBFTRRT9	FP	DIAG
DBFTRSO9	FP	DIAG
DBFTRTF9	FP	DIAG
DBFTRXL9	FP	DIAG
DBFTSIE9	FP	DIAG
DBFTSTS9	FP	DIAG
DBFTVIA9	FP	DIAG
DBFT24B0	FP	CNTRL
DBFUAMB0	FP	UTIL
DBFUBUG0	FP	UTIL
DBFUDLB0	FP	UTIL
DBFUHIC0	FP	UTIL
DBFULTA0	FP	UTIL
DBFUMAC9	FP	UTIL
DBFUMAF0	FP	UTIL
DBFUMAI0	FP	UTIL

Module	Function	Subfunction
DBFUMAL0	FP	UTIL
DBFUMAN0	FP	UTIL
DBFUMAV0	FP	UTIL
DBFUMCAS	FP	UTIL
DBFUMCB9	FP	UTIL
DBFUMCC9	FP	UTIL
DBFUMCF9	FP	UTIL
DBFUMCI9	FP	UTIL
DBFUMCL0	FP	UTIL
DBFUMCP9	FP	UTIL
DBFUMCS9	FP	UTIL
DBFUMCT9	FP	UTIL
DBFUMCU9	FP	UTIL
DBFUMCV0	FP	UTIL
DBFUMCW9	FP	UTIL
DBFUMDAS	FP	UTIL
DBFUMDA9	FP	UTIL
DBFUMDES	FP	UTIL
DBFUMDF0	FP	UTIL
DBFUMDIS	FP	UTIL
DBFUMDL0	FP	UTIL
DBFUMDP0	FP	UTIL
DBFUMDRS	FP	UTIL
DBFUMDR0	FP	UTIL
DBFUMDS0	FP	UTIL
DBFUMER0	FP	UTIL
DBFUMEUS	FP	UTIL
DBFUMFB9	FP	UTIL
DBFUMFL0	FP	UTIL
DBFUMFR9	FP	UTIL
DBFUMFT0	FP	UTIL
DBFUMGB9	FP	UTIL
DBFUMGS9	FP	UTIL
DBFUMHV0	FP	UTIL
DBFUMIL9	FP	UTIL
DBFUMIM9	FP	UTIL
DBFUMIN0	FP	UTIL
DBFUMIN9	FP	UTIL
DBFUMMH0	FP	UTIL
DBFUMMSS	FP	UTIL
DBFUMMS0	FP	UTIL
DBFUMMT0	FP	UTIL
DBFUMNO0	FP	UTIL
DBFUMOP0	FP	UTIL
DBFUMOS9	FP	UTIL
DBFUMPA0	FP	UTIL

Module	Function	Subfunction
DBFUMPI0	FP	UTIL
DBFUMPR0	FP	UTIL
DBFUMPR9	FP	UTIL
DBFUMPV0	FP	UTIL
DBFUMQS0	FP	UTIL
DBFUMRBS	FP	UTIL
DBFUMRDS	FP	UTIL
DBFUMRD9	FP	UTIL
DBFUMRE9	FP	UTIL
DBFUMRI0	FP	UTIL
DBFUMRT0	FP	UTIL
DBFUMRV0	FP	UTIL
DBFUMSC0	FP	UTIL
DBFUMSE0	FP	UTIL
DBFUMSL9	FP	UTIL
DBFUMSP0	FP	UTIL
DBFUMTC0	FP	UTIL
DBFUMTQ9	FP	UTIL
DBFUMTR0	FP	UTIL
DBFUMT8S	FP	UTIL
DBFUMWB9	FP	UTIL
DBFUMWL0	FP	UTIL
DBFUMWQ9	FP	UTIL
DBFUMWR9	FP	UTIL
DBFUMWS0	FP	UTIL
DBFUMWT0	FP	UTIL
DBFUMZE9	FP	UTIL
DBFUNAL0	FP	UTIL
DBFUS470	FP	UTIL
DBFVIDS0	FP	DEDB
DBFVOCI0	FP	DEDB
DBFVPRO0	FP	DEDB
DBFVSOP0	FP	DEDB
DBFVSOW0	FP	DEDB
DBFVSPL0	FP	DEDB
DBFVSRO0	FP	DEDB
DBFVXCS0	FP	DEDB
DBFVXOE0	FP	DEDB
DBFVXOI0	FP	DEDB
DBFVXOW0	FP	DEDB
DBFVXPL0	FP	DEDB
DBFWAKEU	FP	UTIL
DFSAFMDM	SYS	DIAG
DFSAFMD0	SYS	CNTRL
DFSAFMP0	SYS	DIAG
DFSAFMT0	SYS	CNTRL

Module	Function	Subfunction
DFSAIPR0	SYS	INTRF
DFSALOG0	SYS	CNTRL
DFSALSC0	SYS	DIAG
DFSALUC0	SYS	DIAG
DFSAMFS0	SYS	CNTRL
DFSAMSG0	SYS	CNTRL
DFSAOE00	SYS	AOI
DFSAOSC0	SYS	ISI
DFSAMSN0	SYS	DIAG
DFSAOSF0	DB	ACSMTH
DFSAOSI0	SYS	CNTRL
DFSAOSM0	SYS	CNTRL
DFSAOS10	DB	ACSMTH
DFSAOS60	DB	ACSMTH
DFSAOS70	DB	ACSMTH
DFSAOS80	DB	ACSMTH
DFSAOUE0	DC	AOI
DFSAPIQ0	SYS	CNTRL
DFSAPI00	SYS	INTRF
DFSAPRC0	SYS	CNTRL
DFSAPRT0	SYS	DIAG
DFSAPSB0	SYS	CNTRL
DFSAPST0	SYS	CNTRL
DFSAPS00	SYS	CNTRL
DFSAPS10	SYS	DIAG
DFSAP360	DC	CNTRL
DFSAQMR0	SYS	CNTRL
DFSARLMD	SYS	DIAG
DFSARLM0	SYS	CNTRL
DFSARST0	SYS	CNTRL
DFSARW00	DC	LMGR
DFSASAP0	SYS	DIAG
DFSASBA0	SYS	CNTRL
DFSASBC0	SYS	CNTRL
DFSASBP0	SYS	CNTRL
DFSASBR0	SYS	CNTRL
DFSASB10	SYS	CNTRL
DFSASB20	SYS	CNTRL
DFSASB30	SYS	CNTRL
DFSASB40	SYS	CNTRL
DFSASCD0	SYS	CNTRL
DFSASDE0	SYS	DIAG
DFSASK00	SYS	SCHD
DFSASLT0	DC	CTRL
DFSASMB0	SYS	CNTRL
DFSASMF0	SYS	DIAG

Module	Function	Subfunction
DFSASMV0	SYS	DIAG
DFSASPQ0	SYS	CNTRL
DFSASSA0	SYS	CNTRL
DFSASSS0	SYS	DIAG
DFSASTA0	SYS	CNTRL
DFSASTG0	SYS	DIAG
DFSASV10	DC	LMGR
DFSASV20	DC	LMGR
DFSASYM0	SYS	DIAG
DFSASYS0	SYS	CNTRL
DFSASY10	SYS	DIAG
DFSATIM0	SYS	CNTRL
DFSATRA0	SYS	CNTRL
DFSATRC0	SYS	DIAG
DFSATRY0	SYS	CNTRL
DFSAUCE0	DB	INTRF
DFSAUEH0	SYS	CNTRL
DFSAUTO0	SYS	DIAG
DFSBACK0	UTIL	DB
DFSBACM0	UTIL	DB
DFSBBLD0	DB	INTRF
DFSBBLK0	SYS	CNTRL
DFSBBLK1	SYS	CNTRL
DFSBBO00	UTIL	DB
DFSBCB00	SYS	SMGR
DFSBCB30	SYS	SMGR
DFSBCB60	SYS	SMGR
DFSBCB61	SYS	SMGR
DFSBCB90	SYS	SMGR
DFSBCKI0	SYS	INIT
DFSBCK00	SYS	INIT
DFSBDMY0	SYS	INIT
DFSBIND0	DB	INTRF
DFSBINT0	DB	INTRF
DFSBML00	SYS	CNTRL
DFSBR140	SYS	CNTRL
DFSBSCK0	DC	LMGR
DFSCAUT0	DC	CNTRL
DFSCBTA0	DC	LMGR
DFSCBTB0	DC	LMGR
DFSCBTC0	DC	LMGR
DFSCBTD0	DC	LMGR
DFSCBTE0	DC	LMGR
DFSCBTF0	DC	LMGR
DFSCBTG0	DC	LMGR
DFSCBTH0	DC	LMGR

Module	Function	Subfunction
DFSCBTJ0	DC	LMGR
DFSCBT00	SYS	SMGR
DFSCBT10	SYS	SMGR
DFSCBT20	SYS	SMGR
DFSCBT30	SYS	SMGR
DFSCBT40	SYS	SMGR
DFSCBT50	SYS	SMGR
DFSCDMP0	SYS	DBCTL
DFSCDSX0	DC	LMGR
DFSCD600	DC	LMGR
DFSCD610	DC	LMGR
DFSCD620	DC	LMGR
DFSCEQS0	SYS	SHRDQ
DFSCESP0	DC	LMGR
DFSCEVT0	SYS	SHRDQ
DFSCFEA0	DC	MFS
DFSCFEI0	DC	MFS
DFSCFEM0	DC	MFS
DFSCFEO0	DC	MFS
DFSCFEP0	DC	MFS
DFSCFEQ0	DC	MFS
DFSCFES0	DC	MFS
DFSCFEX0	DC	MFS
DFSCFEZ0	DC	CNTRL
DFSCFE00	DC	MFS
DFSCFE10	DC	MFS
DFSCFE80	DC	MFS
DFSCFE90	DC	MFS
DFSCFRT0	DC	MFS
DFSCINB0	SYS	INIT
DFSCIOA0	DC	CNTRL
DFSCIOB0	DC	CNTRL
DFSCIO20	SYS	SHRDQ
DFSCIO30	SYS	SHRDQ
DFSCIR00	SYS	DISP
DFSCKWD0	DC	CMD
DFSCLMA0	DC	CNTRL
DFSCLMO0	DC	CNTRL
DFSCLMR0	DC	CNTRL
DFSCLMR2	DC	CNTRL
DFSCLM00	SYS	CNTRL
DFSCLM10	SYS	CNTRL
DFSCLM20	SYS	CNTRL
DFSCMCP0	MSC	CTC
DFSCMCT0	MSC	CTC
DFSCMCX0	MSC	CTC

Module	Function	Subfunction
DFSCMC00	MSC	CTC
DFSCMC10	MSC	CTC
DFSCMC20	MSC	CTC
DFSCMC40	MSC	CTC
DFSCMC50	MSC	CTC
DFSCMDX0	DC	CMD
DFSCMD30	DC	TPCALL
DFSCMD60	SYS	AOI
DFSCMI00	MSC	CMD
DFSCMLA0	MSC	CMD
DFSCMLB0	MSC	CMD
DFSCMLR0	MSC	CNTRL
DFSCML70	MSC	CMD
DFSCMMP0	MSC	MTM
DFSCMMU0	MSC	MTM
DFSCMMX0	MSC	MTM
DFSCMM20	MSC	MTM
DFSCMPR0	MSC	CNTRL
DFSCMPX0	DB	DBCALL
DFSCMR00	MSC	CNTRL
DFSCMSA0	MSC	CNTRL
DFSCMSB0	MSC	CNTRL
DFSCMSD0	MSC	CNTRL
DFSCMSE0	MSC	CNTRL
DFSCMSF0	MSC	CNTRL
DFSCMSH0	MSC	CNTRL
DFSCMSI0	MSC	CNTRL
DFSCMSM0	MSC	CNTRL
DFSCMSS0	MSC	CNTRL
DFSCMST0	SYS	CHKRT
DFSCMSV0	MSC	CNTRL
DFSCMSW0	MSC	CNTRL
DFSCMSY0	MSC	CNTRL
DFSCMS00	MSC	CNTRL
DFSCMS30	MSC	CNTRL
DFSCMS60	MSC	CNTRL
DFSCMS70	MSC	CNTRL
DFSCMS80	MSC	CNTRL
DFSCMTI0	DC	CNTRL
DFSCMTR0	MSC	CNTRL
DFSCMT00	DC	CNTRL
DFSCMT10	DC	CNTRL
DFSCMT20	DC	CNTRL
DFSCMT30	DC	CNTRL
DFSCMT40	DC	CNTRL
DFSCMT50	DC	CNTRL

Module	Function	Subfunction
DFSCMVA0	MSC	VTAM
DFSCMVC0	MSC	VTAM
DFSCMVR0	MSC	VTAM
DFSCMV20	MSC	VTAM
DFSCM1A0	MSC	VTAM
DFSCM1C0	MSC	VTAM
DFSCM1D0	MSC	VTAM
DFSCM1E0	MSC	VTAM
DFSCM1F0	MSC	VTAM
DFSCM2A0	MSC	VTAM
DFSCM2B0	MSC	VTAM
DFSCM2E0	MSC	VTAM
DFSCM3A0	MSC	VTAM
DFSCM4A0	MSC	VTAM
DFSCM4F0	MSC	VTAM
DFSCM4G0	MSC	VTAM
DFSCM4H0	MSC	VTAM
DFSCM4J0	MSC	VTAM
DFSCM4K0	MSC	VTAM
DFSCM4L0	MSC	VTAM
DFSCM4M0	MSC	VTAM
DFSCM4X0	MSC	VTAM
DFSCM7A0	MSC	VTAM
DFSCM7B0	MSC	VTAM
DFSCM7D0	MSC	VTAM
DFSCM7V0	MSC	VTAM
DFSCM7W0	MSC	VTAM
DFSCM7X0	MSC	VTAM
DFSCM7Y0	MSC	VTAM
DFSCM7Z0	MSC	VTAM
DFSCNS00	SYS	INIT
DFSCNTE0	DC	CNTRL
DFSCNVT0	SYS	CNTRL
DFSCNXA0	DC	LMGR
DFSCOFC0	DC	MFS
DFSCOMP0	DC	MFS
DFSCONA0	DC	CONV
DFSCONE0	DC	CONV
DFSCONG0	DC	CONV
DFSCONM0	DC	CONV
DFSCONP0	DC	CONV
DFSCONU0	DC	CONV
DFSCON00	DC	CONV
DFSCON10	DC	CONV
DFSCON20	DC	CONV
DFSCPCP0	SYS	CHKRT

Module	Function	Subfunction
DFSCPDD0	DC	OLC
DFSCPDM0	DC	OLC
DFSCPIN0	DC	LMGR
DFSCPIR0	SYS	SYSCALL
DFSCPPD0	DC	OLC
DFSCPPS0	DC	OLC
DFSCPSM0	DC	OLC
DFSCPY00	DB	ANAL
DFSCPY30	DB	ANAL
DFSCPY50	SYS	SYSCALL
DFSCPY70	SYS	CHKPT
DFSCREP0	SYS	CHKRT
DFSCRPB0	SYS	CHKRT
DFSCRPC0	SYS	CHKRT
DFSCRPD0	SYS	CHKRT
DFSCRPQ0	SYS	CHKRT
DFSCRPV0	SYS	CHKRT
DFSCRSA0	DC	LMGR
DFSCRSB0	SYS	CHKRT
DFSCRSC0	DC	LMGR
DFSCRSD0	DC	LMGR
DFSCRSE0	DC	LMGR
DFSCRSF0	DC	LMGR
DFSCRSH0	DC	LMGR
DFSCRSL0	DC	LMGR
DFSCRSM0	DC	LMGR
DFSCRSN0	DC	LMGR
DFSCRSO0	DC	LMGR
DFSCRSP0	SYS	CHKRT
DFSCRSR0	DC	LMGR
DFSCRSS0	DC	LMGR
DFSCRST0	DC	LMGR
DFSCRSU0	DC	LMGR
DFSCRSV0	DC	LMGR
DFSCRSW0	DC	LMGR
DFSCRSX0	DC	LMGR
DFSCRS10	DC	LMGR
DFSCRS20	DC	LMGR
DFSCRS40	DC	LMGR
DFSCRS50	DC	LMGR
DFSCRS60	DC	LMGR
DFSCRS70	DC	LMGR
DFSCRS80	DC	LMGR
DFSCR2I0	DC	LMGR
DFSCR2K0	DC	LMGR
DFSCR2Y0	DC	LMGR

Module	Function	Subfunction
DFSCR2Z0	DC	LMGR
DFSCSEA0	DC	LMGR
DFSCSEG0	DC	LMGR
DFSCSF10	SYS	CALLSERV
DFSCSF20	SYS	CALLSERV
DFSCSF30	SYS	CALLSERV
DFSCSGN0	DC	CNTRL
DFSCSIE0	SYS	CALLSERV
DFSCSI00	SYS	CALLSERV
DFSCSI00	SYS	CNTRL
DFSCSMB0	DC	CNTRL
DFSCSPI0	DC	LMGR
DFSCSS00	SYS	INIT
DFSCST00	SYS	CNTRL
DFSCSUB0	DC	CNTRL
DFSCS3G0	DC	LMGR
DFSCS3J0	DC	LMGR
DFSCS3P0	DC	LMGR
DFSCS3Q0	DC	LMGR
DFSCS7A0	DC	LMGR
DFSCS7B0	DC	LMGR
DFSCS7C0	DC	LMGR
DFSCS7D0	DC	LMGR
DFSCS7G0	DC	LMGR
DFSCS7I0	DC	LMGR
DFSCS7L0	DC	LMGR
DFSCS7P0	DC	LMGR
DFSCS7T0	DC	LMGR
DFSCS7U0	DC	LMGR
DFSCTCEQ	MSC	CTC
DFSCTIM0	SYS	CNTRL
DFSCTRN0	DC	CNTRL
DFSCTTO0	DC	LMGR
DFSCVCB0	DC	LMGR
DFSCVCC0	DC	LMGR
DFSCVCD0	DC	LMGR
DFSCVCE0	DC	LMGR
DFSCVCF0	DC	LMGR
DFSCVCG0	DC	LMGR
DFSCVCI0	DC	LMGR
DFSCVCK0	DC	LMGR
DFSCVCL0	DC	LMGR
DFSCVCN0	DC	LMGR
DFSCVCO0	DC	LMGR
DFSCVCP0	DC	LMGR
DFSCVCQ0	DC	LMGR

Module	Function	Subfunction
DFSCVCR0	DC	LMGR
DFSCVCS0	DC	LMGR
DFSCVCT0	DC	LMGR
DFSCVCV0	DC	LMGR
DFSCVEA0	DC	LMGR
DFSCVEB0	DC	LMGR
DFSCVEC0	DC	LMGR
DFSCVED0	DC	LMGR
DFSCVEE0	DC	LMGR
DFSCVEF0	DC	LMGR
DFSCVEG0	DC	LMGR
DFSCVEH0	DC	LMGR
DFSCVEI0	DC	LMGR
DFSCVEJ0	DC	LMGR
DFSCVEK0	DC	LMGR
DFSCVEL0	DC	LMGR
DFSCVEM0	DC	LMGR
DFSCVEN0	DC	LMGR
DFSCVEO0	DC	LMGR
DFSCVEP0	DC	LMGR
DFSCVEQ0	DC	LMGR
DFSCVER0	DC	LMGR
DFSCVES0	DC	LMGR
DFSCVET0	DC	LMGR
DFSCVFA0	DC	LMGR
DFSCVFC0	DC	LMGR
DFSCVFD0	DC	LMGR
DFSCVFG0	DC	LMGR
DFSCVFH0	DC	LMGR
DFSCVFI0	DC	LMGR
DFSCVFJ0	DC	LMGR
DFSCVFM0	DC	LMGR
DFSCVFN0	DC	LMGR
DFSCVFP0	DC	LMGR
DFSCVFQ0	DC	LMGR
DFSCVFR0	DC	LMGR
DFSCVFS0	DC	LMGR
DFSCVFX0	DC	LMGR
DFSCVFY0	DC	LMGR
DFSCVFZ0	DC	LMGR
DFSCVF10	DC	LMGR
DFSCVF30	DC	LMGR
DFSCVF40	DC	LMGR
DFSCVF60	DC	LMGR
DFSCVF70	DC	LMGR
DFSCVGA0	DC	LMGR

Module	Function	Subfunction
DFSCVGB0	DC	LMGR
DFSCVGC0	DC	LMGR
DFSCVGD0	DC	LMGR
DFSCVGE0	DC	LMGR
DFSCVGF0	DC	LMGR
DFSCVGG0	DC	LMGR
DFSCVGH0	DC	LMGR
DFSCVGI0	DC	LMGR
DFSCVGJ0	DC	LMGR
DFSCVGK0	DC	LMGR
DFSCVGL0	DC	LMGR
DFSCVGM0	DC	LMGR
DFSCVGN0	DC	LMGR
DFSCVGO0	DC	LMGR
DFSCVGP0	DC	LMGR
DFSCVGQ0	DC	LMGR
DFSCVHA0	DC	LMGR
DFSCVHB0	DC	LMGR
DFSCVHC0	DC	LMGR
DFSCVHD0	DC	LMGR
DFSCVHE0	DC	LMGR
DFSCVHF0	DC	LMGR
DFSCVHH0	DC	LMGR
DFSCVHI0	DC	LMGR
DFSCVHK0	DC	LMGR
DFSCVHL0	DC	LMGR
DFSCVHM0	DC	LMGR
DFSCVHN0	DC	LMGR
DFSCVHP0	DC	LMGR
DFSCVHQ0	DC	LMGR
DFSCVHR0	DC	LMGR
DFSCVHS0	DC	LMGR
DFSCVHT0	DC	LMGR
DFSCVHX0	DC	LMGR
DFSCVHZ0	DC	LMGR
DFSCVH60	DC	LMGR
DFSCVH70	DC	LMGR
DFSCVJB0	DC	LMGR
DFSCVJK0	DC	LMGR
DFSCVJL0	DC	LMGR
DFSCVJM0	DC	LMGR
DFSCVJO0	DC	LMGR
DFSCVJR0	DC	LMGR
DFSCVLG0	DC	LMGR
DFSCVRA0	DC	LMGR
DFSCVRB0	DC	LMGR

Module	Function	Subfunction
DFSCVRC0	DC	LMGR
DFSCVRF0	DC	LMGR
DFSCVRG0	DC	LMGR
DFSCVRH0	DC	LMGR
DFSCVRJ0	DC	LMGR
DFSCVRK0	DC	LMGR
DFSCVRL0	DC	LMGR
DFSCVRM0	DC	LMGR
DFSCVRN0	DC	LMGR
DFSCVRO0	DC	LMGR
DFSCVRP0	DC	LMGR
DFSCVRR0	DC	LMGR
DFSCVRS0	DC	LMGR
DFSCVRT0	DC	LMGR
DFSCVRY0	DC	LMGR
DFSCVRZ0	DC	LMGR
DFSCVTM0	SYS	CNTRL
DFSCWU00	SYS	DISP
DFSDABN0	SYS	SCHD
DFSDAPL0	DB	INTRF
DFSDASB0	SYS	DBCTL
DFSDASC0	SYS	DBCTL
DFSDASD0	SYS	DBCTL
DFSDASG0	SYS	DBCTL
DFSDASI0	SYS	DBCTL
DFSDASP0	SYS	DBCTL
DFSDASR0	SYS	DBCTL
DFSDASS0	SYS	DBCTL
DFSDAST0	SYS	DBCTL
DFSDAST0	SYS	SCHD
DFSDBAU0	DB	INTRF
DFSDBAU0	SYS	INIT
DFSDBCTG	SYS	DBCTL
DFSDBCTL	SYS	DBCTL
DFSDBDR0	DC	CMD
DFSDBH10	DB	CMGR
DFSDBH20	DB	CMGR
DFSDBH30	DB	CMGR
DFSDBH40	DB	CMGR
DFSDBIE0	SYS	CHKRT
DFSDBLB0	DB	INTRF
DFSDBLD0	DB	INTRF
DFSDBLI0	DB	INTRF
DFSDBLM0	SYS	SCHD
DFSDBLN0	DB	INTRF
DFSDBLP0	DB	INTRF

Module	Function	Subfunction
DFSDBLR0	DB	INTRF
DFSDBSM0	DB	CMGR
DFSDCFC0	DB	CMGR
DFSDCFR0	DB	CMGR
DFSDCLM0	DC	LMGR
DFSDCPY0	DB	ANAL
DFSDDLE0	DB	DBCALL
DFSDDLSA	UTIL	TSTTOOL
DFSDDLSI	UTIL	TSTTOOL
DFSDDLS1	UTIL	TSTTOOL
DFSDDLS2	UTIL	TSTTOOL
DFSDDLS3	UTIL	TSTTOOL
DFSDDLS4	UTIL	TSTTOOL
DFSDDLS5	UTIL	TSTTOOL
DFSDDLS6	UTIL	TSTTOOL
DFSDDLS7	UTIL	TSTTOOL
DFSDDLS8	UTIL	TSTTOOL
DFSDDLS9	UTIL	TSTTOOL
DFSDDLT0	UTIL	TSTTOOL
DFSDDUI0	DB	CMGR
DFSDECP0	DB	INTRF
DFSDENF0	DB	CMGR
DFSDFLS0	DB	ANAL
DFSDHD00	DB	CMGR
DFSDINB0	SYS	DBCTL
DFSDLAS0	DB	ANAL
DFSDLA00	DB	ANAL
DFSDLA30	DC	TPCALL
DFSDLA50	SYS	SYSCALL
DFSDLBI0	DB	DBCALL
DFSDLBL0	DB	INTRF
DFSDLBN0	SYS	INIT
DFSDLB00	DB	INTRF
DFSDLB10	DB	INTRF
DFSDLB20	DB	INTRF
DFSDLB30	DB	INTRF
DFSDLB40	DB	INTRF
DFSDLB50	DB	INTRF
DFSDLB60	DB	INTRF
DFSDLB70	DB	INTRF
DFSDLB80	DB	INTRF
DFSDLDC0	DB	DBCALL
DFSDLDD0	DB	DBCALL
DFSDLDR0	DB	DBCALL
DFSDLDW0	DB	DBCALL
DFSDLD00	DB	DBCALL

Module	Function	Subfunction
DFSDLICS	SYS	CNTRL
DFSDLKX0	SYS	CNTRL
DFSDLN00	SYS	INIT
DFSDLOC0	DB	CMGR
DFSDLOV0	DB	CMGR
DFSDLPF0	SYS	CNTRL
DFSDLPR0	SYS	INTRF
DFSDLR00	DB	DBCALL
DFSDLTR0	SYS	CNTRL
DFSDMG10	SYS	DMMGR
DFSDMG20	SYS	DMMGR
DFSDMG30	SYS	DMMGR
DFSDMG40	SYS	DMMGR
DFSDMG50	SYS	DMMGR
DFSDMIF0	SYS	DMMGR
DFSDMIQ0	SYS	DMMGR
DFSDMAW0	DB	CMGR
DFSDMSG0	DB	INTRF
DFSDNSC0	DC	LMGR
DFSDNS20	DC	LMGR
DFSDNS30	DC	LMGR
DFSDN010	DC	LMGR
DFSDN020	DC	LMGR
DFSDN030	DC	LMGR
DFSDN040	DC	LMGR
DFSDN050	DC	LMGR
DFSDN060	DC	LMGR
DFSDN070	DC	LMGR
DFSDN080	DC	LMGR
DFSDN090	DC	LMGR
DFSDN100	DC	LMGR
DFSDN110	DC	LMGR
DFSDN120	DC	LMGR
DFSDN130	DC	LMGR
DFSDN140	DC	LMGR
DFSDN150	DC	LMGR
DFSDN160	DC	LMGR
DFSDN170	DC	LMGR
DFSDN190	DC	LMGR
DFSDN230	DC	LMGR
DFSDN240	DC	LMGR
DFSDN250	DC	LMGR
DFSDN260	DC	LMGR
DFSDN270	DC	LMGR
DFSDN280	DC	LMGR
DFSDN290	DC	LMGR

Module	Function	Subfunction
DFSDN520	MSC	CNTRL
DFSDN530	MSC	CNTRL
DFSDN540	MSC	CNTRL
DFSDN550	MSC	CNTRL
DFSDOBI0	SYS	INIT
DFSDPDM0	SYS	SCHD
DFSDPRH0	DB	INTRF
DFSDPSB0	UTIL	DB
DFSDQMG0	SYS	DBCTL
DFSDRCL0	SYS	CNTRL
DFSDRID0	SYS	DBCTL
DFSDRID0	SYS	CHKRT
DFSDRIS0	SYS	DBCTL
DFSDRIS0	SYS	CHKRT
DFSDSC00	SYS	CHKRT
DFSDSEH0	UTIL	DB
DFSDSPI0	SYS	DISP
DFSDSPS0	SYS	DISP
DFSDSPX0	SYS	DISP
DFSDSSI0	SYS	DBCTL
DFSDSST0	DB	CMGR
DFSDSTA0	SYS	DBCTL
DFSDSTP0	SYS	DBCTL
DFSDS010	DC	LMGR
DFSDS020	DC	LMGR
DFSDS030	DC	LMGR
DFSDS040	DC	LMGR
DFSDS050	DC	LMGR
DFSDS060	DC	LMGR
DFSDS070	DC	LMGR
DFSDTTA0	SYS	SCHD
DFSDUMYC	SYS	CNTRL
DFSDUMYE	SYS	CNTRL
DFSDUMYR	SYS	CNTRL
DFSDVBH0	DB	CMGR
DFSDVBI0	SYS	INIT
DFSDVSM0	DB	CMGR
DFSDXES0	DB	CMGR
DFSDXMT0	DB	CMGR
DFSDYA00	SYS	INIT
DFSECP10	DB	INTRF
DFSECP20	DB	INTRF
DFSEIPB0	DB	INTRF
DFSERA10	UTIL	SYS
DFSERA20	SYS	LOG
DFSERA30	UTIL	SYS

Module	Function	Subfunction
DFSERA40	UTIL	SYS
DFSERA50	SYS	LOG
DFSERA60	SYS	CNTRL
DFSERA70	UTIL	DB
DFSESAB0	SYS	ESS
DFSESCL0	SYS	ESS
DFSESCT0	SYS	ESS
DFSESD10	SYS	ESS
DFSESD20	SYS	ESS
DFSESD30	SYS	ESS
DFSESD40	SYS	ESS
DFSESD50	SYS	ESS
DFSESD60	SYS	ESS
DFSESD70	SYS	ESS
DFSESD80	SYS	ESS
DFSESD90	SYS	ESS
DFSESFM0	SYS	ESS
DFSESGL0	SYS	ESS
DFSESI00	SYS	ESS
DFSESI20	SYS	ESS
DFSESI30	SYS	ESS
DFSESI40	SYS	ESS
DFSESI50	SYS	ESS
DFSESI60	SYS	ESS
DFSESI70	SYS	ESS
DFSESMG0	SYS	ESS
DFSESPL0	SYS	ESS
DFSESPR0	SYS	ESS
DFSESP10	SYS	ESS
DFSESP20	SYS	ESS
DFSESSO0	SYS	ESS
DFSESS10	SYS	ESS
DFSESS20	SYS	ESS
DFSESS30	SYS	ESS
DFSESS40	SYS	ESS
DFSFAB00	SYS	CNTRL
DFSFAB10	SYS	CNTRL
DFSFCMP0	SYS	SMGR
DFSFCST0	SYS	CNTRL
DFSFCTT0	SYS	INIT
DFSFCTX0	SYS	CNTRL
DFSFDDL0	SYS	SMGR
DFSFDDT0	SYS	CNTRL
DFSFDEQ	DC	MFS
DFSFDIR0	SYS	INIT
DFSFDLB0	SYS	LOG

Module	Function	Subfunction
DFSFDLD0	SYS	LOG
DFSFDLE0	SYS	LOG
DFSFDLF0	SYS	LOG
DFSFDLG0	SYS	LOG
DFSFDLI0	SYS	CNTRL
DFSFDLN0	SYS	LOG
DFSFDLO0	SYS	LOG
DFSFDLP0	SYS	LOG
DFSFDLQ0	SYS	LOG
DFSFDLR0	SYS	LOG
DFSFDLS0	SYS	LOG
DFSFDLT0	SYS	LOG
DFSFDLU0	SYS	LOG
DFSFDLV0	SYS	LOG
DFSFDLW0	SYS	LOG
DFSFDLX0	SYS	LOG
DFSFDLY0	SYS	LOG
DFSFDLZ0	SYS	LOG
DFSFDMP0	SYS	CNTRL
DFSFDSC0	DC	MFS
DFSFDYA0	SYS	INTRF
DFSFEBJ0	DC	FES
DFSFESI0	SYS	ESS
DFSFESP0	SYS	ESS
DFSFES00	DC	FES
DFSFES20	SYS	ESS
DFSFFET0	DC	MFS
DFSFFRH0	DC	MFS
DFSFHSH0	DC	MFS
DFSFLLG0	SYS	LOG
DFSFLOAT	FP	DEDB
DFSFLST0	SYS	LOG
DFSFLTP0	UTIL	LOG
DFSFMOD0	SYS	CNTRL
DFSFMOD0	SYS	INIT
DFSFPAT0	SYS	DRA
DFSFPGS0	DC	MFS
DFSFPMM0	DC	MFS
DFSFPRA0	SYS	DRA
DFSFRDS0	SYS	CHKRT
DFSFRLWA	SYS	LOG
DFSFRSP0	DB	CMGR
DFSFRST0	SYS	CHKRT
DFSFSQ10	SYS	SHRDQ
DFSFSQ20	SYS	SHRDQ
DFSFSTM0	SYS	CNTRL

Module	Function	Subfunction
DFSFSWA0	SYS	CNTRL
DFSFTCF0	DC	TCO
DFSFTIM0	SYS	CNTRL
DFSFTIN0	SYS	INIT
DFSFTRA0	SYS	CNTRL
DFSFTRM0	DC	LMGR
DFSFVS10	SYS	CNTRL
DFSFXC10	SYS	CNTRL
DFSFXC30	SYS	CHKRT
DFSFXC40	SYS	CHKRT
DFSFXC50	SYS	CHKRT
DFSGENFL	FP	UTIL
DFSGESB0	SYS	ESS
DFSGGSP0	DB	CMGR
DFSHASH0	SYS	CNTRL
DFSHAVM0	DC	LMGR
DFSHAV00	DC	CNTRL
DFSHAV10	DC	CNTRL
DFSHAV20	DC	CNTRL
DFSHAV30	DC	CNTRL
DFSHAV40	DC	CNTRL
DFSHAV70	DC	CNTRL
DFSHCEX0	DC	CNTRL
DFSHCI00	DC	CNTRL
DFSHCI10	DC	CNTRL
DFSHCLG0	DC	CNTRL
DFSHCMS0	IXRF	TRK
DFSHCSW0	IXRF	TKO
DFSHDAI0	IXRF	TRK
DFSHDCL0	IXRF	TKO
DFSHDC10	DB	CMGR
DFSHDC20	DB	CMGR
DFSHDC30	DB	CMGR
DFSHDC40	DB	CMGR
DFSHDEP0	SYS	CNTRL
DFSHIC40	DC	CMD
DFSHINT0	SYS	INIT
DFSHIN10	SYS	INIT
DFSHLIN0	SYS	INIT
DFSHLOG0	DC	CMD
DFSHLTK0	IXRF	TRK
DFSHMFS0	IXRF	TRK
DFSHPTK0	IXRF	TRK
DFSHQMV0	IXRF	TKO
DFSHRAL0	IXRF	TKO
DFSHRCL0	IXRF	TRK

Module	Function	Subfunction
DFSHRDB0	IXRF	TRK
DFSHREQ0	IXRF	TKO
DFSHSRV0	SYS	CNTRL
DFSHSSF0	SYS	CNTRL
DFSHTIM0	IXRF	TRK
DFSHTKO0	IXRF	TRK
DFSHTKR0	IXRF	TKO
DFSHTRM0	SYS	CNTRL
DFSHVIO0	IXRF	TKO
DFSICAT0	UTIL	MFS
DFSICA10	DC	CMD
DFSICA20	DC	CMD
DFSICIO0	DC	CNTRL
DFSICLA0	DC	CMD
DFSICLB0	DC	CNTRL
DFSICLC0	DC	CMD
DFSICLD0	DC	CMD
DFSICLE0	DC	CMD
DFSICLF0	DC	CNTRL
DFSICLG0	DC	CMD
DFSICLH0	DC	CMD
DFSICLI0	DC	CMD
DFSICLJ0	DC	CMD
DFSICLK0	DC	CMD
DFSICLL0	SYS	CNTRL
DFSICLL1	SYS	CNTRL
DFSICLM0	DC	CMD
DFSICLN0	DC	CMD
DFSICLP0	DC	CMD
DFSICLQ0	DC	CMD
DFSICLR0	DC	CNTRL
DFSICLS0	DC	CNTRL
DFSICLT0	DC	CNTRL
DFSICLU0	DC	CMD
DFSICLV0	DC	CMD
DFSICLW0	DC	CMD
DFSICLX0	DC	CNTRL
DFSICLY0	DC	CMD
DFSICLZ0	DC	CMD
DFSICL10	DC	CMD
DFSICL20	DC	CMD
DFSICL30	DC	CMD
DFSICL40	DC	CMD
DFSICL50	DC	CMD
DFSICL60	DC	CMD
DFSICL70	DC	CMD

Module	Function	Subfunction
DFSICL80	DC	CMD
DFSICL90	DC	CMD
DFSICM00	DC	CMD
DFSICQ10	SYS	SHRDQ
DFSICQ20	SYS	SHRDQ
DFSICQ30	SYS	SHRDQ
DFSICSC0	DC	CMD
DFSICUR0	DC	CNTRL
DFSICVA0	DC	CMD
DFSICVD0	DC	CMD
DFSICVE0	DC	CMD
DFSICVF0	DC	CMD
DFSICV10	DC	CMD
DFSICV20	DC	CMD
DFSICV30	DC	CMD
DFSICV40	DC	CMD
DFSICV50	DC	CMD
DFSICV60	DC	CMD
DFSICV70	DC	CMD
DFSICV80	DC	CMD
DFSICV90	DC	CMD
DFSICWA2	DC	CMD
DFSIC410	DC	CMD
DFSIC420	DC	CMD
DFSIC430	DC	CMD
DFSIC440	DC	CMD
DFSIC450	DC	CMD
DFSIC460	DC	CMD
DFSIC470	DC	CMD
DFSIC480	DC	CMD
DFSIDDP0	UTIL	MFS
DFSIDPA0	DC	CMD
DFSIDPR0	SYS	SHRDQ
DFSIDPB0	DC	CMD
DFSIDPC0	DC	CMD
DFSIDPD0	DC	CMD
DFSIDPE0	DC	CMD
DFSIDPF0	DC	CMD
DFSIDPG0	DC	CMD
DFSIDPH0	DC	CMD
DFSIDPI0	DC	CMD
DFSIDPJ0	DC	CMD
DFSIDPK0	DC	CMD
DFSIDPL0	DC	CMD
DFSIDP00	DC	CMD
DFSIDPQ0	DC	CMD

Module	Function	Subfunction
DFSIDPS0	SYS	SHRDQ
DFSIDPT0	SYS	SHRDQ
DFSIDP10	DC	CMD
DFSIDP20	DC	CMD
DFSIDP30	DC	CMD
DFSIDP40	DC	CMD
DFSIDP50	DC	CMD
DFSIDP60	DC	CMD
DFSIDP70	DC	CMD
DFSIDP80	DC	CMD
DFSIDP90	DC	CMD
DFSIDSP0	SYS	DISP
DFSIESI0	SYS	INIT
DFSIFIX0	SYS	INIT
DFSIIDE0	SYS	QMGR
DFSIIDM0	SYS	INIT
DFSIIEN0	SYS	QMGR
DFSIIMS0	MSC	CNTRL
DFSIINB0	SYS	INIT
DFSIIND0	SYS	INIT
DFSIING0	SYS	INIT
DFSIINL0	SYS	INIT
DFSIINM0	SYS	INIT
DFSIINQ0	SYS	INIT
DFSIINS0	SYS	INIT
DFSIINV0	SYS	INIT
DFSIIN10	SYS	INIT
DFSIIN30	SYS	INIT
DFSIIOC0	SYS	INIT
DFSIIO30	SYS	INIT
DFSII150	SYS	INIT
DFSILMT0	ILS	INIT
DFSILNK0	SYS	CNTRL
DFSILQS0	SYS	SHRDQ
DFSILTA0	UTIL	SYS
DFSILTXT	ILS	SER
DFSIL010	ILS	CMD
DFSIL110	ILS	INIT
DFSIL210	ILS	INIT
DFSIL220	ILS	INIT
DFSIL230	ILS	CNTRL
DFSIL240	ILS	INIT
DFSIL250	ILS	CNTRL
DFSIL300	ILS	CONV
DFSIL310	ILS	CONV
DFSIL320	ILS	CONV

Module	Function	Subfunction
DFSIL330	ILS	SER
DFSIL340	ILS	LOG
DFSIL350	ILS	LOG
DFSIL390	ILS	CMD
DFSIL400	ILS	SER
DFSIL500	ILS	SER
DFSIL510	ILS	SER
DFSIMBD0	SYS	SCHD
DFSIMBE0	SYS	SCHD
DFSIMNT0	SYS	LOG
DFSIMP00	UTIL	SYS
DFSIMP10	UTIL	SYS
DFSIMP20	UTIL	SYS
DFSIMP30	UTIL	SYS
DFSINDX0	UTIL	MFS
DFSINTRA	SYS	CNTRL
DFSIOBP0	DB	ACSMTH
DFSIPCP0	SYS	CHKRT
DFSIPOL0	DC	LMGR
DFSIPST0	SYS	CNTRL
DFSIRAC0	DC	CNTRL
DFSIRD10	DC	CMD
DFSIRST0	SYS	CHRKT
DFSISCN0	UTIL	MFS
DFSISC00	DC	CNTRL
DFSISIS0	SYS	SCHD
DFSISI00	SYS	ISI
DFSISI10	SYS	ISI
DFSISI20	SYS	DBCTL
DFSISMI0	SYS	INIT
DFSISMN0	SYS	SMGR
DFSISTS0	UTIL	SYS
DFSIST20	UTIL	SYS
DFSIST30	UTIL	SYS
DFSIST40	UTIL	SYS
DFSISUB0	UTIL	MFS
DFSITQS0	SYS	SHRDQ
DFSI7770	DC	LMGR
DFSKBDP0	SYS	DISP
DFSKDP00	SYS	INIT
DFSKDS10	SYS	DISP
DFSKDS20	SYS	DISP
DFSKEYT0	SYS	CNTRL
DFSKLDLI	SYS	CNTRL
DFSKLSO0	SYS	INIT
DFSKLSPT	SYS	CNTRL

Module	Function	Subfunction
DFSKMPX0	DB	INTRF
DFSKPXT0	SYS	CNTRL
DFSLATE0	SYS	CNTRL
DFSLAWE0	SYS	CNTRL
DFSLBLM0	SYS	SCHD
DFSLGD00	UTIL	SYS
DFSLIE00	SYS	INTRF
DFSLIE20	SYS	INTRF
DFSLI000	DB	INTRF
DFSLLCL0	DB	CMGR
DFSLMGR0	SYS	CNTRL
DFSLRHC0	SYS	CNTRL
DFSLRH00	SYS	CNTRL
DFSLSAB0	SYS	SMGR
DFSLSM00	DC	CNTRL
DFSLTMG0	UTIL	MSC
DFSMAID0	UTIL	DB
DFSMDA00	SYS	CNTRL
DFSMDA10	SYS	CNTRL
DFSME000	DC	LMGR
DFSME127	DC	LMGR
DFSMINI0	SYS	INIT
DFSMMLC0	DB	CMGR
DFSMMUD0	DB	CMGR
DFSMNTB0	SYS	LOG
DFSMNTR0	SYS	LOG
DFSMNZ00	SYS	SHRDQ
DFSMODE0	SYS	CNTRL
DFSMODF0	SYS	CNTRL
DFSMODS0	SYS	CNTRL
DFSMODU0	SYS	CNTRL
DFSMPOS0	SYS	CNTRL
DFSMRCL0	SYS	CNTRL
DFSMRTR0	DC	LMGR
DFSMTMA0	MSC	MTM
DFSMVRC0	SYS	CNTRL
DFSNNIC0	DB	CMGR
DFSNOTB0	DB	CMGR
DFSNOTI0	SYS	CNTRL
DFSNOTX0	SYS	CNTRL
DFSOCMT0	SYS	CNTRL
DFSOFMD0	UTIL	SYS
DFSO7770	DC	LMGR
DFSPAGE0	DC	LMGR
DFSPARSE	DC	CNTRL
DFSPAT00	SYS	DRA

Module	Function	Subfunction
DFSPAT20	SYS	DRA
DFSPAUL0	DB	INTRF
DFSPCCC0	SYS	INIT
DFSPCC20	SYS	CNTRL
DFSPCC30	SYS	CNTRL
DFSPCIB0	DC	MFS
DFSPCR00	SYS	INTRF
DFSPCSH0	DB	CMGR
DFSPDLI0	SYS	DRA
DFSPGLD0	SYS	CNTRL
DFSPIEX0	SYS	CNTRL
DFSPINI0	SYS	DRA
DFSPIRP0	UTIL	SYS
DFSPIXT0	DC	LMGR
DFSPLAT0	SYS	DRA
DFSPLDL0	SYS	INIT
DFSPLDR0	SYS	INIT
DFSPLDT0	SYS	INIT
DFSPLOAD	SYS	CNTRL
DFSPLPP0	SYS	INIT
DFSPMSG0	SYS	DRA
DFSPNRT0	SYS	DRA
DFSPREC0	UTIL	DB
DFSPPTK0	SYS	DRA
DFSPRABC	UTIL	DB
DFSPRA10	SYS	DRA
DFSPRA20	SYS	DRA
DFSPRA30	SYS	DRA
DFSPRA40	SYS	DRA
DFSPRA50	SYS	DRA
DFSPRA60	SYS	DRA
DFSPRCHK	UTIL	DB
DFSPRCLN	UTIL	DB
DFSPRCT1	UTIL	DB
DFSPRCT2	UTIL	DB
DFSPRC10	SYS	DRA
DFSPRDBD	UTIL	DB
DFSPREQ0	SYS	DMMGR
DFSPRERR	UTIL	DB
DFSPRE00	UTIL	SYS
DFSPRE05	UTIL	SYS
DFSPRE10	UTIL	SYS
DFSPRE20	UTIL	SYS
DFSPRE30	UTIL	SYS
DFSPRE40	UTIL	SYS
DFSPRE50	UTIL	SYS

Module	Function	Subfunction
DFSPRE60	UTIL	SYS
DFSPRE70	UTIL	SYS
DFSPRE80	UTIL	SYS
DFSPRH00	SYS	INTRF
DFSPRIMS	UTIL	DB
DFSPRMS0	SYS	CNTRL
DFSPRNT0	UTIL	SYS
DFSPRPAR	UTIL	DB
DFSPRPSB	UTIL	DB
DFSPRPX0	SYS	SCHD
DFSPRRA0	SYS	DRA
DFSPRRC0	SYS	DRA
DFSPRRD0	SYS	DBCTL
DFSPRREP	UTIL	DB
DFSPRRE0	SYS	DBCTL
DFSPRRG0	SYS	CNTRL
DFSPRSCC	UTIL	DB
DFSPRSDI	SUR	none
DFSPRSDI	UTIL	DB
DFSPRSDS	SUR	none
DFSPRSDS	UTIL	DB
DFSPRSER	SUR	none
DFSPRSER	UTIL	DB
DFSPRSFR	SUR	none
DFSPRSFR	UTIL	DB
DFSPRSIM	SUR	none
DFSPRSIM	UTIL	DB
DFSPRSPA	SUR	none
DFSPRSPA	UTIL	DB
DFSPRSRF	UTIL	DB
DFSPRSTC	UTIL	DB
DFSPRSTO	SUR	none
DFSPRSTO	UTIL	DB
DFSPRSTW	UTIL	DB
DFSPRSUR	SUR	none
DFSPRSUR	UTIL	DB
DFSPRUPD	UTIL	DB
DFSPRURC	UTIL	DB
DFSPRWFM	UTIL	DB
DFSPR000	DB	INTRF
DFSPSCH0	SYS	DRA
DFSPSDB0	SYS	CNTRL
DFSPSE00	DB	DBCALL
DFSPSEL0	DB	CMGR
DFSPSM10	DB	CMGR
DFSPSNP0	SYS	DRA

Module	Function	Subfunction
DFSPSTB0	SYS	CNTRL
DFSPSYN0	SYS	DRA
DFSPTCH0	SYS	CNTRL
DFSPTRA0	SYS	DRA
DFSPTTH0	SYS	DRA
DFSPUSC0	SYS	DRA
DFSPZP00	SYS	DRA
DFSQBFM0	SYS	QMGR
DFSQCP00	SYS	CHKRT
DFSQCQS0	SYS	SHRDQ
DFSQC010	SYS	QMGR
DFSQC020	SYS	QMGR
DFSQC030	SYS	QMGR
DFSQC040	SYS	QMGR
DFSQC050	SYS	QMGR
DFSQC060	SYS	QMGR
DFSQC070	SYS	QMGR
DFSQC080	SYS	QMGR
DFSQDOC0	SYS	CNTRL
DFSQDQ00	SYS	QMGR
DFSQENQ0	SYS	QMGR
DFSQEQ00	SYS	QMGR
DFSQFIX0	SYS	QMGR
DFSQGU00	SYS	QMGR
DFSQIS00	SYS	QMGR
DFSQLOG0	SYS	QMGR
DFSQMGR0	SYS	QMGR
DFSQMRQ0	SYS	QMGR
DFSQMRT0	SYS	QMGR
DFSQNP00	SYS	QMGR
DFSQRH00	SYS	QMGR
DFSQRL00	SYS	QMGR
DFSQRST0	SYS	QMGR
DFSQSAB0	SYS	SMGR
DFSQSPC0	SYS	QMGR
DFSQUEI0	SYS	QMGR
DFSQXF00	SYS	QMGR
DFSRBCP0	SYS	CHKRT
DFSRBLB0	SYS	CHKRT
DFSRBOI0	SYS	CHKRT
DFSRCHB0	DB	CMGR
DFSRCP00	SYS	CHKRT
DFSRCP10	SYS	CHKRT
DFSRCP30	SYS	CHKRT
DFSRCP40	SYS	CHKRT
DFSRCQM0	SYS	CNTRL

Module	Function	Subfunction
DFSRCQR0	SYS	CNTRL
DFSRCRT0	SYS	CHKRT
DFSRDBC0	UTIL	DB
DFSRDBL0	SYS	LOG
DFSRDBP0	SYS	CHKRT
DFSRDSH0	SYS	CNTRL
DFSRDS00	SYS	CHKRT
DFSRDS10	SYS	CHKRT
DFSRDUP0	SYS	CHKRT
DFSRDY00	SYS	CHKRT
DFSRED20	SYS	CNTRL
DFSRELP0	SYS	INIT
DFSREPS0	DC	CMD
DFSREP00	SYS	DISP
DFSRESP0	SYS	ESS
DFSRESX0	SYS	CNTRL
DFSRHSH0	SYS	CNTRL
DFSRLCC0	SYS	CHKRT
DFSRLMP0	SYS	CNTRL
DFSRLP00	SYS	CHKRT
DFSRMDD0	DC	OLC
DFSRMDM0	DC	OLC
DFSRMPD0	DC	OLC
DFSRMPS0	DC	OLC
DFSRMSM0	DC	OLC
DFSRMS00	DC	OLC
DFSRRAE0	SYS	CNTRL
DFSRRA00	SYS	CNTRL
DFSRRA10	SYS	CNTRL
DFSRRA20	SYS	CNTRL
DFSRRA30	SYS	CNTRL
DFSRRA40	SYS	CNTRL
DFSRRA50	SYS	CNTRL
DFSRRA60	SYS	DBCTL
DFSRRA70	SYS	CNTRL
DFSRRA80	SYS	CNTRL
DFSRRA90	SYS	CNTRL
DFSRRC00	SYS	CNTRL
DFSRRC10	SYS	CNTRL
DFSRRC40	SYS	CNTRL
DFSRREF0	SYS	INIT
DFSRRHM0	DB	CMGR
DFSRRHP0	DB	CMGR
DFSRSDDM	DC	CNTRL
DFSRSLST	DC	CNTRL
DFSRST00	SYS	CHKRT

Module	Function	Subfunction
DFSRTM00	SYS	CNTRL
DFSSABN0	SYS	SCHD
DFSSAM01	SYS	IVP
DFSSAM02	SYS	IVP
DFSSAM03	SYS	IVP
DFSSAM04	SYS	IVP
DFSSAM05	SYS	IVP
DFSSAM06	SYS	IVP
DFSSAM07	SYS	IVP
DFSSAM08	SYS	IVP
DFSSAM11	SYS	IVP
DFSSAM12	SYS	IVP
DFSSAM13	SYS	IVP
DFSSAM15	SYS	IVP
DFSSAM16	SYS	IVP
DFSSAM17	SYS	IVP
DFSSAM18	SYS	IVP
DFSSBCA0	DB	CMGR
DFSSBCI0	DB	CMGR
DFSSBCN0	DB	CMGR
DFSSBCQ0	DB	CMGR
DFSSBCR0	DB	CMGR
DFSSBCW0	DB	CMGR
DFSSBEV0	DB	CMGR
DFSSBEX0	DB	CMGR
DFSSBGI0	DB	CMGR
DFSSBGM0	DB	CMGR
DFSSBHD0	DB	CMGR
DFSSBIC0	DB	CMGR
DFSSBID0	DB	CMGR
DFSSBIE0	DB	CMGR
DFSSBIL0	DB	CMGR
DFSSBIO0	DB	CMGR
DFSSBIP0	DB	CMGR
DFSSBIS0	DB	CMGR
DFSSBIT0	DB	CMGR
DFSSBIX0	DB	CMGR
DFSSBI00	DB	CMGR
DFSSBI10	DB	CMGR
DFSSBLK0	SYS	SMGR
DFSSBMP0	SYS	SCHD
DFSSBOP0	DB	CMGR
DFSSBSN0	DB	CMGR
DFSSBSP0	DB	CMGR
DFSSBSR0	DB	CMGR
DFSSBTD0	DB	CMGR

Module	Function	Subfunction
DFSSBT00	DB	CMGR
DFSSBT10	DB	CMGR
DFSSBWC0	DB	CMGR
DFSSCBT0	SYS	SMGR
DFSSCHP0	SYS	SCHD
DFSSCHQ0	SYS	SCHD
DFSSCHR0	SYS	SCHD
DFSSDAB0	SYS	CNTRL
DFSSDA10	SYS	CNTRL
DFSSDA20	SYS	CNTRL
DFSSDA30	SYS	DBCTL
DFSSDCI0	SYS	SHRDQ
DFSSDLA0	DB	INTRF
DFSSDLB0	DB	INTRF
DFSSDLC0	SYS	INIT
DFSSDL20	SYS	INIT
DFSSDL30	SYS	CNTRL
DFSSDL40	SYS	CNTRL
DFSSDL60	DB	INTRF
DFSSDL70	SYS	INIT
DFSSDL80	SYS	INIT
DFSSDL90	SYS	INIT
DFSSEQS0	SYS	SHRDQ
DFSSEVT0	SYS	SHRDQ
DFSSIDT0	SYS	SMGR
DFSSIDX0	SYS	CNTRL
DFSSIMC0	SYS	SMGR
DFSSIML0	DC	LMGR
DFSSINF0	SYS	SHRDQ
DFSSLAT0	SYS	SMGR
DFSSMIC0	SYS	SCHD
DFSSMSC0	SYS	SCHD
DFSSMUP0	UTIL	SYS
DFSSPF00	SYS	SMGR
DFSSPIN0	SYS	CNTRL
DFSSPST0	SYS	CNTRL
DFSSQCP0	SYS	SHRDQ
DFSSQI00	SYS	SHRDQ
DFSSQI10	SYS	SHRDQ
DFSSQI20	SYS	SHRDQ
DFSSQOF0	SYS	SHRDQ
DFSSQ000	SYS	SHRDQ
DFSSQ010	SYS	SHRDQ
DFSSQ020	SYS	SHRDQ
DFSSQ030	SYS	SHRDQ
DFSSQ040	SYS	SHRDQ

Module	Function	Subfunction
DFSSRPR0	SYS	INTRF
DFSSSAP0	SYS	CNTRL
DFSSSITP	SYS	CNTRL
DFSSSMU0	SYS	SMGR
DFSSSREQ	SYS	CNTRL
DFSSS000	UTIL	SYS
DFSSS010	UTIL	SYS
DFSSS020	UTIL	SYS
DFSSS030	UTIL	SYS
DFSSS040	UTIL	SYS
DFSSS050	UTIL	SYS
DFSSS060	UTIL	SYS
DFSSTAT0	SYS	LOG
DFSSTAX0	SYS	CNTRL
DFSSTM00	SYS	SMGR
DFSSTOP0	DC	CMD
DFSSTS10	SYS	CNTRL
DFSSUSX0	SYS	CNTRL
DFSSUT04	SYS	IVP
DFSSVT00	SYS	SMGR
DFSSYI40	SYS	CHKPT
DFSSYI50	SYS	CHKPT
DFSS3741	DC	LMGR
DFSS7770	DC	LMGR
DFSTAUTH	DC	CNTRL
DFSTBLD0	DC	TCO
DFSTDDM0	DC	TCO
DFSTDLI0	DC	TCO
DFSTERM0	SYS	CNTRL
DFSTEXP0	SYS	CNTRL
DFSTINI0	DC	TCO
DFSTMAD0	SYS	SCHD
DFSTMAP0	SYS	SCHD
DFSTMAS0	SYS	SCHD
DFSTMCD0	SYS	INIT
DFSTMED0	SYS	SCHD
DFSTMEI0	SYS	INIT
DFSTMII0	SYS	CNTRL
DFSTMOD0	SYS	INIT
DFSTMPR0	SYS	INTRF
DFSTMR00	SYS	CNTRL
DFSTMSB0	SYS	CNTRL
DFSTMSE0	SYS	CNTRL
DFSTMSG0	DC	TCO
DFSTMSS0	SYS	CNTRL
DFSTMS00	SYS	CHKRT

Module	Function	Subfunction
DFSTOBH0	DB	CMGR
DFSTOCL0	DB	CMGR
DFSTOCP0	DB	CMGR
DFSTODI0	DB	CMGR
DFSTODU0	DB	CMGR
DFSTOFM0	DB	CMGR
DFSTOFN0	DB	CMGR
DFSTOGM0	DB	CMGR
DFSTOLG0	DB	CMGR
DFSTOPR0	DB	CMGR
DFSTOPU0	DB	CMGR
DFSTORS0	DB	CMGR
DFSTPCP0	DC	TCO
DFSTPRO0	DC	TCO
DFSTRABK	SYS	CNTRL
DFSTRACE	SYS	CNTRL
DFSTRAE0	SYS	CNTRL
DFSTRAG0	SYS	DIAG
DFSTRA00	SYS	INIT
DFSTRA10	SYS	CNTRL
DFSTRA20	SYS	CNTRL
DFSTRA30	SYS	CNTRL
DFSTRA40	SYS	DIAG
DFSTRMOD	SYS	CNTRL
DFSTRM00	SYS	INIT
DFSTRTN0	DC	TCO
DFSTST00	SYS	CNTRL
DFSTTIM0	DC	TCO
DFSTVER0	DC	TCO
DFSTXIT0	DC	TCO
DFSUACB0	UTIL	DB
DFSUAMB0	UTIL	DB
DFSUARA0	UTIL	LOG
DFSUARC0	UTIL	LOG
DFSUARP0	UTIL	LOG
DFSUCCT0	UTIL	DB
DFSUCER0	UTIL	DB
DFSUCF00	UTIL	DB
DFSUCMN0	UTIL	DB
DFSUCPA0	UTIL	DB
DFSUCPB0	UTIL	DB
DFSUCPC0	UTIL	DB
DFSUCPD0	UTIL	DB
DFSUCPE0	UTIL	DB
DFSUCP00	UTIL	DB
DFSUCP10	UTIL	DB

Module	Function	Subfunction
DFSUCP20	UTIL	DB
DFSUCP30	UTIL	DB
DFSUCP40	UTIL	DB
DFSUCP50	UTIL	DB
DFSUCP60	UTIL	DB
DFSUCP70	UTIL	DB
DFSUCP80	UTIL	DB
DFSUCP90	UTIL	DB
DFSUCTR0	UTIL	DB
DFSUCUM0	UTIL	DB
DFSUC150	UTIL	DB
DFSUC350	UTIL	DB
DFSUC450	UTIL	DB
DFSUDMP0	UTIL	DB
DFSUDMT0	UTIL	DB
DFSUDUI0	UTIL	DB
DFSUEX10	UTIL	DB
DFSUICC0	UTIL	DB
DFSUICP0	UTIL	DB
DFSULG10	UTIL	LOG
DFSULG20	UTIL	LOG
DFSULG40	UTIL	LOG
DFSULG50	UTIL	LOG
DFSULTR0	UTIL	LOG
DFSUMGT0	UTIL	DB
DFSUMSG0	UTIL	DB
DFSUMSV0	UTIL	MSC
DFSUNPK0	UTIL	MFS
DFSUNUA0	UTIL	MFS
DFSUNUB0	UTIL	MFS
DFSUOCU0	UTIL	SYS
DFSUPAA0	UTIL	MFS
DFSUPAB0	UTIL	MFS
DFSUPAC0	UTIL	MFS
DFSUPAD0	UTIL	MFS
DFSUPAE0	UTIL	MFS
DFSUPAF0	UTIL	MFS
DFSUPAG0	UTIL	MFS
DFSUPAH0	UTIL	MFS
DFSUPAJ0	UTIL	MFS
DFSUPAK0	UTIL	MFS
DFSUPAL0	UTIL	MFS
DFSUPAM0	UTIL	MFS
DFSUPAN0	UTIL	MFS
DFSUPAP0	UTIL	MFS
DFSUPAQ0	UTIL	MFS

Module	Function	Subfunction
DFSUPAR0	UTIL	MFS
DFSUPAS0	UTIL	MFS
DFSUPAT0	UTIL	MFS
DFSUPAU0	UTIL	MFS
DFSUPAV0	UTIL	MFS
DFSUPAW0	UTIL	MFS
DFSUPAX0	UTIL	MFS
DFSUPAY0	UTIL	MFS
DFSUPAZ0	UTIL	MFS
DFSUPA00	UTIL	MFS
DFSUPA10	UTIL	MFS
DFSUPA20	UTIL	MFS
DFSUPA30	UTIL	MFS
DFSUPA60	UTIL	MFS
DFSUPA70	UTIL	MFS
DFSUPA80	UTIL	MFS
DFSUPA90	UTIL	MFS
DFSUPBA0	UTIL	MFS
DFSUPBB0	UTIL	MFS
DFSUPBE0	UTIL	MFS
DFSUPBF0	UTIL	MFS
DFSUPBH0	UTIL	MFS
DFSUPBJ0	UTIL	MFS
DFSUPBK0	UTIL	MFS
DFSUPBL0	UTIL	MFS
DFSUPBM0	UTIL	MFS
DFSUPBN0	UTIL	MFS
DFSUPBO0	UTIL	MFS
DFSUPBP0	UTIL	MFS
DFSUPBQ0	UTIL	MFS
DFSUPBZ0	UTIL	MFS
DFSUPB00	UTIL	MFS
DFSUPB10	UTIL	MFS
DFSUPB20	UTIL	MFS
DFSUPB30	UTIL	MFS
DFSUPB50	UTIL	MFS
DFSUPB60	UTIL	MFS
DFSUPB70	UTIL	MFS
DFSUPRT0	UTIL	SYS
DFSURDB0	UTIL	DB
DFSURGL0	UTIL	DB
DFSURGP0	UTIL	DB
DFSURGS0	UTIL	DB
DFSURGU0	UTIL	DB
DFSURG10	UTIL	DB
DFSURIO0	UTIL	DB

Module	Function	Subfunction
DFSURPR0	UTIL	DB
DFSURRL0	UTIL	DB
DFSURTR0	UTIL	DB
DFSURT00	UTIL	DB
DFSURUI0	UTIL	DB
DFSURUL0	UTIL	DB
DFSUSCH0	UTIL	DB
DFSUSE00	SYS	CNTRL
DFSUSE10	SYS	CNTRL
DFSUSE20	SYS	CNTRL
DFSUSRC0	UTIL	DB
DFSUSRXI	SYS	USEREXIT
DFSUSRX0	SYS	USEREXIT
DFSUTLA0	UTIL	MFS
DFSUTLB0	UTIL	MFS
DFSUTLC0	UTIL	MFS
DFSUTLD0	UTIL	MFS
DFSUTLE0	UTIL	MFS
DFSUTLF0	UTIL	MFS
DFSUTLG0	UTIL	MFS
DFSUTLH0	UTIL	MFS
DFSUTLJ0	UTIL	MFS
DFSUTLN0	UTIL	MFS
DFSUTLP0	UTIL	MFS
DFSUTLT0	UTIL	MFS
DFSUTLU0	UTIL	MFS
DFSUTLV0	UTIL	MFS
DFSUTLW0	UTIL	MFS
DFSUTLX0	UTIL	MFS
DFSUTLY0	UTIL	MFS
DFSUTLZ0	UTIL	MFS
DFSUTL00	UTIL	MFS
DFSUTL30	UTIL	MFS
DFSUTL40	UTIL	MFS
DFSUTL60	UTIL	MFS
DFSUTL70	UTIL	MFS
DFSUTL80	UTIL	MFS
DFSUTL90	UTIL	MFS
DFSUTR20	UTIL	SYS
DFSUTR30	UTIL	SYS
DFSUTSA0	UTIL	MFS
DFSUTSB0	UTIL	MFS
DFSUTSC0	UTIL	MFS
DFSUTSD0	UTIL	MFS
DFSUTSE0	UTIL	MFS
DFSUTSF0	UTIL	MFS

Module	Function	Subfunction
DFSUTSG0	UTIL	MFS
DFSUTSH0	UTIL	MFS
DFSUTSK0	UTIL	MFS
DFSUTSO0	UTIL	MFS
DFSUTSQ0	UTIL	MFS
DFSUTSR0	UTIL	MFS
DFSUT0A0	UTIL	MFS
DFSUT0I0	UTIL	MFS
DFSUT0T0	UTIL	MFS
DFSUT010	UTIL	MFS
DFSUT020	UTIL	MFS
DFSUT030	UTIL	MFS
DFSUT040	UTIL	MFS
DFSUT050	UTIL	MFS
DFSUT060	UTIL	MFS
DFSUT070	UTIL	MFS
DFSUT080	UTIL	MFS
DFSUT090	UTIL	MFS
DFSUT110	UTIL	MFS
DFSUT120	UTIL	MFS
DFSUT130	UTIL	MFS
DFSUT140	UTIL	MFS
DFSUT150	UTIL	MFS
DFSUT160	UTIL	MFS
DFSUT170	UTIL	MFS
DFSUT180	UTIL	MFS
DFSUT190	UTIL	MFS
DFSUT200	UTIL	MFS
DFSUT260	UTIL	MFS
DFSUT280	UTIL	MFS
DFSUT290	UTIL	MFS
DFSUT300	UTIL	MFS
DFSVBLK0	DC	LMGR
DFSVCI00	SYS	INIT
DFSVCI10	SYS	ISI
DFSVCPRM	SYS	ISI
DFSVC000	SYS	CNTRL
DFSVC200	SYS	CNTRL
DFSVES00	SYS	ESS
DFSVMPRS	DC	CNTRL
DFSVMSGE	DC	CNTRL
DFSVMT00	DC	CNTRL
DFSVMT10	DC	CNTRL
DFSVMT20	DC	CNTRL
DFSVMT30	DC	CNTRL
DFSVMT40	DC	CNTRL

Module	Function	Subfunction
DFSVTDDM	DC	CNTRL
DFSVTDIR	DC	CNTRL
DFSV4100	SYS	CNTRL
DFSV4200	SYS	CNTRL
DFSWRAP0	DC	LMGR
DFSXBAT0	SYS	INIT
DFSXBC60	SYS	SMGR
DFSXCB00	SYS	INIT
DFSXCFB0	DB	INIT
DFSXCIC0	SYS	INIT
DFSXCTL0	SYS	INIT
DFSXDBI0	SYS	INIT
DFSXDLG0	SYS	LOG
DFSXDLLL	SYS	INIT
DFSXDL00	SYS	INIT
DFSXDL10	SYS	INIT
DFSXDRC0	SYS	INIT
DFSXDSP0	SYS	INIT
DFSXDYA0	SYS	INIT
DFSXDYB0	SYS	CNTRL
DFSXESI0	SYS	INIT
DFSXFIX0	SYS	INIT
DFSXIOB0	SYS	INIT
DFSXLGI0	SYS	LOG
DFSXLGJ0	SYS	LOG
DFSXLIC0	SYS	INIT
DFSXLSM0	SYS	INIT
DFSXNCL0	SYS	INIT
DFSXOLDS	SYS	LOG
DFSXRBL0	SYS	INIT
DFSXRDS0	SYS	INIT
DFSXRIC0	SYS	INIT
DFSXRID0	SYS	INIT
DFSXRLM0	IRLM	INIT
DFSXRPS0	SYS	INIT
DFSXRST0	SYS	INIT
DFSXSQ10	SYS	SHRDQ
DFSXSQ20	SYS	SHRDQ
DFSXSTA0	SYS	INIT
DFSXSTM0	SYS	INIT
DFSXTRA0	SYS	DIAG
DFSYCM20	SYS	SHRDQ
DFSYNCL6	DC	LMGR
DFSZDC00	DB	INTRF
DFSZDI00	SYS	INIT
DFSZDI20	SYS	CNTRL

Module	Function	Subfunction
DFSZDI30	SYS	CNTRL
DFSZDI40	SYS	CNTRL
DFSZD110	DB	CMGR
DFSZD150	DB	ACSMTH
DFSZD210	DB	ACSMTH
DFSZD250	DB	ACSMTH
DFSZD310	DB	ACSMTH
DFSZD510	SYS	CNTRL
DFSZSC00	SYS	CHKRT
DFSZSR00	SYS	CHKRT
DFSZSR10	SYS	CHKRT
DFSZZZ97	SYS	CNTRL
DFSZZZ98	SYS	CNTRL
DFSZZZ99	SYS	CNTRL
DFS29800	DC	LMGR
DFS36010	DC	LMGR
DFS36140	DC	LMGR
DSPADS00	DBRC	EXIT
DSPADTIM	DBRC	SER
DSPALD00	DBRC	EXIT
DSPALD10	DBRC	EXIT
DSPALD20	DBRC	EXIT
DSPALD30	DBRC	EXIT
DSPAMS00	DBRC	SER
DSPARCST	DBRC	SER
DSPARC00	DBRC	EXIT
DSPARC10	DBRC	EXIT
DSPBUFFS	DBRC	SER
DSPCABN0	DBRC	SER
DSPCEXT0	DBRC	EXIT
DSPCEXT1	DBRC	EXIT
DSPCHKWD	DBRC	SER
DSPCINT0	DBRC	SER
DSPCRTR0	DBRC	CNTRL
DSPDBGST	DBRC	SER
DSPDBG00	DBRC	EXIT
DSPDBSWP	DBRC	EXIT
DSPDEQE	DBRC	SER
DSPDEQ00	DBRC	SER
DSPDIUST	DBRC	SER
DSPDIU00	DBRC	EXIT
DSPDLT00	DBRC	SER
DSPDSN00	DBRC	EXIT
DSPDTM	DBRC	SER
DSPDUHAA	DBRC	SER
DSPDUHAD	DBRC	SER

Module	Function	Subfunction
DSPDUHCR	DBRC	SER
DSPDUHDA	DBRC	SER
DSPDUHDL	DBRC	SER
DSPDUHDS	DBRC	SER
DSPDUHEA	DBRC	SER
DSPDUHIT	DBRC	SER
DSPDUHLA	DBRC	SER
DSPDUHLO	DBRC	SER
DSPEXHRS	DBRC	SER
DSPFLPCR	DBRC	SER
DSPFLPDS	DBRC	SER
DSPFLPGE	DBRC	SER
DSPFLTST	DBRC	SER
DSPFLT00	DBRC	EXIT
DSPFSIGN	DBRC	EXIT
DSPGDALC	DBRC	SER
DSPGFREE	DBRC	SER
DSPGPAR0	DBRC	SER
DSPHICBG	DBRC	EXIT
DSPHICED	DBRC	EXIT
DSPHIC00	DBRC	EXIT
DSPHSHAD	DBRC	SER
DSPHSHCR	DBRC	SER
DSPHSHDL	DBRC	SER
DSPHSHDS	DBRC	SER
DSPHSHIM	DBRC	SER
DSPHSHLO	DBRC	SER
DSPHSHMS	DBRC	SER
DSPHSHPC	DBRC	SER
DSPICP00	DBRC	SER
DSPIDB00	DBRC	EXIT
DSPJBMAI	DBRC	CMD
DSPJBSAL	DBRC	CMD
DSPJBSCA	DBRC	CMD
DSPJBSDB	DBRC	CMD
DSPJBSEL	DBRC	CMD
DSPJBSIC	DBRC	CMD
DSPJBSOL	DBRC	CMD
DSPJBSRL	DBRC	CMD
DSPJBSSL	DBRC	CMD
DSPJCARC	DBRC	CMD
DSPJCCAC	DBRC	CMD
DSPJCCLO	DBRC	CMD
DSPJCIMG	DBRC	CMD
DSPJCMAI	DBRC	CMD
DSPJCRCV	DBRC	CMD

Module	Function	Subfunction
DSPJCUSR	DBRC	CMD
DSPJDFLT	DBRC	CMD
DSPJGMEM	DBRC	CMD
DSPJKMGR	DBRC	CMD
DSPJRN00	DBRC	EXIT
DSPKWTBL	DBRC	SER
DSPLGRAD	DBRC	SER
DSPLGRDL	DBRC	SER
DSPLINK1	DBRC	SER
DSPLINK2	DBRC	SER
DSPLOADR	DBRC	SER
DSPLRCST	DBRC	SER
DSPLRC00	DBRC	EXIT
DSPLRC10	DBRC	EXIT
DSPNORSR	DBRC	SER
DSPOFRCD	DBRC	SER
DSPOFRLR	DBRC	SER
DSPOFRMD	DBRC	SER
DSPOLDST	DBRC	SER
DSPOLD00	DBRC	EXIT
DSPOLD10	DBRC	EXIT
DSPPTKOV	DBRC	EXIT
DSPQARCL	DBRC	EXIT
DSPQHPTK	DBRC	EXIT
DSPQLOGS	DBRC	EXIT
DSPQNLOG	DBRC	EXIT
DSPQOFRL	DBRC	EXIT
DSPQSGLS	DBRC	EXIT
DSPQTLOG	DBRC	EXIT
DSPRCDCP	DBRC	SER
DSPRCDEX	DBRC	SER
DSPRESET	DBRC	CMD
DSPRSV00	DBRC	SER
DSPSAUCE	DBRC	EXIT
DSPSAUTH	DBRC	EXIT
DSPSBOER	DBRC	EXIT
DSPSDBUA	DBRC	EXIT
DSPSGCMD	DBRC	EXIT
DSPSIOER	DBRC	EXIT
DSPSLSCS	DBRC	SER
DSPSLSPV	DBRC	SER
DSPSLSSV	DBRC	SER
DSPSOPEN	DBRC	EXIT
DSPSSFA0	DBRC	EXIT
DSPSSFN0	DBRC	EXIT
DSPSSIGN	DBRC	EXIT

Module	Function	Subfunction
DSPSSNRE	DBRC	SER
DSPSTATS	DBRC	EXIT
DSPSTFRE	DBRC	SER
DSPSTGET	DBRC	SER
DSPTAUTH	DBRC	EXIT
DSPTDBUA	DBRC	EXIT
DSPTEOFR	DBRC	EXIT
DSPTETRK	DBRC	EXIT
DSPTIME0	DBRC	SER
DSPTLG00	DBRC	EXIT
DSPTLTRN	DBRC	EXIT
DSPTNUSI	DBRC	EXIT
DSPTQALL	DBRC	EXIT
DSPTQUIT	DBRC	EXIT
DSPTRACE	DBRC	SER
DSPTREPL	DBRC	EXIT
DSPTSCOV	DBRC	SER
DSPTSIGN	DBRC	EXIT
DSPTSSST	DBRC	SER
DSPTSTCM	DBRC	EXIT
DSPTTKOV	DBRC	EXIT
DSPUALL0	DBRC	SER
DSPUAUTH	DBRC	EXIT
DSPUBKST	DBRC	SER
DSPUBK00	DBRC	EXIT
DSPUBK10	DBRC	EXIT
DSPUBK20	DBRC	EXIT
DSPUBU00	DBRC	CMD
DSPUCAIN	DBRC	SER
DSPUCAST	DBRC	SER
DSPUCA00	DBRC	EXIT
DSPUCLA0	DBRC	SER
DSPUCLCA	DBRC	SER
DSPUCP40	DBRC	SER
DSPUDB00	DBRC	SER
DSPUDV00	DBRC	SER
DSPUEX00	DBRC	SER
DSPUGP00	DBRC	SER
DSPUICST	DBRC	SER
DSPUIC00	DBRC	EXIT
DSPUIN00	DBRC	SER
DSPUNQ00	DBRC	SER
DSPURCLC	DBRC	SER
DSPURCMH	DBRC	SER
DSPURCM1	DBRC	SER
DSPURCM2	DBRC	SER

Module	Function	Subfunction
DSPURCM3	DBRC	SER
DSPURCM4	DBRC	SER
DSPURCM5	DBRC	SER
DSPURCM6	DBRC	SER
DSPURCM7	DBRC	SER
DSPURCM8	DBRC	SER
DSPURCM9	DBRC	SER
DSPURC00	DBRC	CMD
DSPURDLB	DBRC	CMD
DSPURD00	DBRC	CMD
DSPURD10	DBRC	CMD
DSPURD20	DBRC	CMD
DSPURD40	DBRC	CMD
DSPURD50	DBRC	CMD
DSPURD60	DBRC	CMD
DSPURD65	DBRC	CMD
DSPURD70	DBRC	CMD
DSPURI00	DBRC	SER
DSPURI10	DBRC	SER
DSPURI20	DBRC	SER
DSPURI30	DBRC	SER
DSPURI40	DBRC	SER
DSPURLBK	DBRC	CMD
DSPURLB2	DBRC	CMD
DSPURLB3	DBRC	CMD
DSPURLB4	DBRC	CMD
DSPURLCO	DBRC	CMD
DSPURLGR	DBRC	CMD
DSPURL00	DBRC	CMD
DSPURMBK	DBRC	CMD
DSPURMCG	DBRC	CMD
DSPURM00	DBRC	CMD
DSPURM10	DBRC	CMD
DSPURM20	DBRC	CMD
DSPURM30	DBRC	CMD
DSPURM35	DBRC	CMD
DSPURM40	DBRC	CMD
DSPURM45	DBRC	CMD
DSPURM50	DBRC	CMD
DSPURM60	DBRC	CMD
DSPURM70	DBRC	CMD
DSPURM80	DBRC	CMD
DSPURM90	DBRC	CMD
DSPURNST	DBRC	SER
DSPURN00	DBRC	EXIT
DSPURPBK	DBRC	CMD

Module	Function	Subfunction
DSPURPDB	DBRC	CMD
DSPURPDD	DBRC	CMD
DSPURPDG	DBRC	CMD
DSPURPHI	DBRC	CMD
DSPURPLB	DBRC	CMD
DSPURPOL	DBRC	CMD
DSPURPSS	DBRC	CMD
DSPURP00	DBRC	CMD
DSPURSLB	DBRC	CMD
DSPURSST	DBRC	SER
DSPURS00	DBRC	CMD
DSPURS10	DBRC	CMD
DSPURS20	DBRC	CMD
DSPURS30	DBRC	CMD
DSPURTBK	DBRC	CMD
DSPURT00	DBRC	CMD
DSPURT10	DBRC	CMD
DSPURT20	DBRC	CMD
DSPURT30	DBRC	CMD
DSPURT50	DBRC	CMD
DSPURT55	DBRC	CMD
DSPURT70	DBRC	CMD
DSPURT80	DBRC	CMD
DSPURT85	DBRC	CMD
DSPURT95	DBRC	CMD
DSPURUEP	DBRC	CMD
DSPURUPD	DBRC	SER
DSPURUST	DBRC	SER
DSPURU00	DBRC	CMD
DSPURU10	DBRC	CMD
DSPURU20	DBRC	CMD
DSPURVIN	DBRC	SER
DSPURVST	DBRC	SER
DSPURVTN	DBRC	EXIT
DSPURVTR	DBRC	EXIT
DSPURV00	DBRC	EXIT
DSPURXCS	DBRC	CMD
DSPURXEP	DBRC	SER
DSPURXST	DBRC	SER
DSPURX00	DBRC	CMD
DSPUSTBS	DBRC	SER
DSPUSTB2	DBRC	SER
DSPUSTCA	DBRC	SER
DSPUSTIC	DBRC	SER
DSPUSTRV	DBRC	SER
DSPUST00	DBRC	SER

Module	Function	Subfunction
DSPUTM00	DBRC	SER
DSPUVF00	DBRC	SER
DSPVMTBL	DBRC	SER
DSPXCPTN	DBRC	SER
DSP00MVS	DBRC	SER
HMDUSRF2	DBRC	SER

IMS Functions and Subfunctions

IMS is comprised of many functions, some of which are subdivided into smaller pieces called subfunctions. Table 193 shows the function and subfunction list.

Table 193. IMS Functions and Subfunctions

Function	Subfunction
DB—Database	DBS/INTRF—Database Application/Scheduling Interface DB/ANAL—Database Call Analyzer DB/DBCALL—Database Call Action Processing DB/CMGR—Database Call Resource Management DB/ACSMTH—Database Access Method Interface
DC—Data Communication	DC/CNTRL—Data Communication Control DC/LMGR—Data Communication Line Management DC/MFS—Data Communication Message Format Services DC/CMD—Data Communication Command Processing DC/CONV—Data Communication Conversational Processing DC/TPCALL—Data Communication DL/I Telecommunications Call Processing DC/ISC—Data Communication Intersystem Communication Processing DC/APPC—Data Communication Advanced Program to Program Communication Processing

Table 193. IMS Functions and Subfunctions (continued)

Function	Subfunction
SYS—System Service	SYS/AOI—System Automated Operator Interface SYS/INIT—System Service Initialization SYS/CNTRL—System Service Control SYS/DMMGR—System Service Directed Message Manager SYS/QMGR—System Service Message Queue Management SYS/SCHD—System Service Scheduling SYS/ISI—System Service Resource Access Security SYS/SMGR—System Service Storage Management SYS/LOG—System Service Logging SYS/CHKPT—System Service Checkpoint Restart Processing SYS/ESS—System Service External Subsystem Support SYS/DBCTL—System Service Database Control Processing SYS/DRA—System Service Database Resource Adapter Processing SYS/SHRDQ—System Service Shared Queues
UTIL—Utilities	UTIL/DB—Database Utilities UTIL/MFS—Message Format Service Utilities UTIL/MSC—Multiple Systems Coupling Utilities UTIL/SYS—System Service Utilities UTIL/DBRC—Database Recovery Control Utilities UTIL/TSTTOOL—Application Call Test Tool

Table 193. IMS Functions and Subfunctions (continued)

Function	Subfunction
FP—Fast Path	FP/CNTRL—Fast Path Control FP/EMH—Fast Path Expedited Message Handling Call Analyzer FP/MSDB—Fast Path Main Storage Database Call Analyzer FP/DEDB—Fast Path Data Entry Database Processing FP/UTIL—Fast Path Utilities and System Definition FP/CMD—Fast Path Command Analyzer FP/INIT—Fast Path Initialization FP/CKPT—Fast Path Checkpoint Processing FP/TKO—XRF Takeover Processing for a Fast Path Environment FP/DIAG—Fast Path Diagnostic Routines FP/LOCK—Fast Path Locking and Notify Logic FP/RSTRT—Fast Path Restart Processing FP/LOG—Fast Path Logging FP/I/O—Fast Path I/O Processing
MSC—Multiple Systems Coupling	MSC/CNTRL—Multiple Systems Coupling Control Processor MSC/CTC—Multiple Systems Coupling Channel-to-Channel Link and Access Method MSC/MTM—Multiple Systems Coupling Main Storage-to-Main Storage Link and Access Method MSC/VTAM—Multiple Systems Coupling Synchronous Data Link Control (SDLC) Communications Link MSC/CMD—Multiple Systems Coupling MSVERIFY Command Message and MSASSIGN Command Processing
SUR—Surveyor Feature	none

Table 193. IMS Functions and Subfunctions (continued)

Function	Subfunction
IRLM—Internal Resource Lock Manager	IRLM/REQ—Internal Resource Lock Manager Request Handler IRLM/DEADLK—Internal Resource Lock Manager Deadlock Detection IRLM/PTB—Internal Resource Lock Manager Pass-the-Buck Processing IRLM/STRMGR—Internal Resource Lock Manager Storage Manager IRLM/MCP—Internal Resource Lock Manager Modify Command Processor IRLM/INIT—Internal Resource Lock Manager Initialization IRLM/TERM—Internal Resource Lock Manager Termination IRLM/FMTDMP—Internal Resource Lock Manager Dump and GTF Format Support
IXRF—Alternate IMS in XRF Complex	IXRF/TRK—XRF-related Tracking IXRF/TAKE—XRF-related Takeover
DBRC—Database Recovery Control	DBRC/EXIT—Database Recovery Control Exit Processing DBRC/CNTRL—Database Recovery Control Processor DBRC/CMD—Database Recovery Control Command Processing DBRC/SER—Database Recovery Control Services
ILS—Isolated Log Send	ILS/INIT—Isolated Log Send Initialization Processing ILS/LOG—Isolated Log Send Log Processing ILS/CMD—Isolated Log Send Command Processing ILS/SER—Isolated Log Send Services ILS/CONV—Isolated Log Send Conversation Processing ILS/CNTRL—Isolated Log Send Control Processing

Appendix D. Save-Area-ID-to-Module Cross-Reference Table

Entries in which the SAVID and the module name are the same have been omitted from this table.

SAVIDs with the identification of DXR only apply to the Internal Resource Lock Manager (IRLM).

SAVID	Module
ACLB0	DFSACLB0
ACNT0	DFSACNT0
ACON0	DFSACON0
ACTB0	DFSACTB0
ADBCxxxx	DFSADBC0
ADCC0	DFSADCC0
ADLDxxxx	DFSADLD0
ADL30	DFSADL30
AFMD0	DFSAFMD0
ALLLINV	DFSFXC50
AMDUSRF6	AMDUSRF6
AMFS0	DFSAMFS0
APPEND1	DFSCLMR0
APSTxxxx	DFSAPST0
AP1xxxx	DFSAAP10
AP2xxxx	DFSAAP20
ASMB0	DFSASMB0
ASTA	DFSASTA0
ASYN1000	DFSRST00
ATTRLKUP	DFSIDPB0
AUEH0	DFSAUEH0
BATSHRTN	DFSRRA00
BLDNCB	DFSDBDR0
BR14	DFSIMNT0
BTAxxxx	DFSCBTA0
BTBxxxx	DFSCBTB0
BTCxxxx	DFSCBTC0
BTDxxxx	DFSCBTD0
BTExxxx	DFSCBTE0
BTFxxxx	DFSCBTF0
BTGxxxx	DFSCBTG0
BTHxxxx	DFSCBTH0
BTJxxxx	DFSCBTJ0
BUILDMSG	DFSFXC50
CALCPSDB	DFSDLDC0
CALLBH	DFSDLDC0
CALLLRH	DFSDLDC0
CALLSM	DFSDLDC0
CATA215	DFSICAT0
CAUTxxxx	DFSCAUT0

SAVID	Module
CAxxxx	DFSICA10
CD1xxxx	DFSCVCD0
CD3xxxx	DFSCVCD0
CE2xxxx	DFSCVCE0
CE6xxxx	DFSCVCE0
CE7xxxx	DFSCVCE0
CFEIMOVE	DFSCFEI0
CHECKSDS	DFSRST00
CHKFMT	DFSUTSK0
CHKLPFLC	DFSDLDD0
CHKSEGFC	DFSDLDD0
CHNGDATA	DFSDLDR0
CINBxxxx	DFSCINB0
CIOAxxxx	DFSCIOA0
CIOA1	DFSCIOA0
CIOA2	DFSCIOA0
CIOA4	DFSCIOA0
CIOB1	DFSCIOB0
CIOB2	DFSCIOB0
CIOB3	DFSCIOB0
CIOB4	DFSCIOB0
CIOB5	DFSCIOB0
CIOB6	DFSCIOB0
CIOB7	DFSCIOB0
CIOC	DFSICIO0
CISHRLOC	DBFMGXC0
CIEXCLOC	DBFMGXC0
CLBxxxx	DFSICLB0
CLEANUP	DFSRDBP0
CLFxxxx	DFSICLF0
CLMRGTBF	DFSCLMR0
CLMRGTWK	DFSCLMR0
CLMRINI0	DFSCLMR0
CLMRINT0	DFSCLMR0
CLMRI020	DFSCLMR0
CLMRKEY0	DFSCLMR0
CLMRPED0	DFSCLMR0
CLM1RTNE	DFSRST00
CLOSERTN	DFSRST00
CLR	DFSICLR0
CLR1	DFSICLR0
CLSxxxx	DFSICLS0
CLTxxxx	DFSICLT0
CLXxxxx	DFSICLX0
CL71-312	DFSICL70
CMBPC	DFSCMBP0

SAVID	Module
CMBPD	DFSCMBP0
CMBPQ	DFSCMBP0
CMBPR	DFSCMBP0
CMBPW	DFSCMBP0
CMBP1	DFSCMBP0
CMBP3	DFSCMBP0
CMCPC	DFSCMCP0
CMCPD	DFSCMCP0
CMCPS	DFSCMCP0
CMCPW	DFSCMCP0
CMCP1	DFSCMCP0
CMI01	DFSCMI00
CMI02	DFSCMI00
CMI03	DFSCMI00
CMI04	DFSCMI00
CMI05	DFSCMI00
CMI06	DFSCMI00
CMLRxxxx	DFSCMLR0
CMMPC	DFSCMMP0
CMMPD	DFSCMMP0
CMMPR	DFSCMMP0
CMMPW	DFSCMMP0
CMMP1	DFSCMMP0
CMPxxxx	DFSCMPR0
CMSW1	DFSCMSW0
CMS30xxx	DFSCMS30
CMS31	DFSCMS30
CMS32	DFSCMS30
CMS33	DFSCMS30
CMS34	DFSCMS30
CMS35	DFSCMS30
CMS36	DFSCMS30
CMS37	DFSCMS30
CMS71	DFSCMS70
CMS81	DFSCMS80
CMTxxxx	DFSCMTR0
CN1xxxx	DFSCVCN0
COMDECOM	DFSDLDC0
COMFUNC	DFSUTR20
COMIWAIT	DFSUTR20
COMSUM	DFSUTR20
COND	DFSCON20
CONG0	DFSCONG0
CONIEXIR	DFSCON10
CONLxxxx	DFSCON10
CONMRWEX	DFSCONM0

SAVID	Module
CONS0	DFSCON00
CONT0	DFSCON20
CONU	DFSCONU0
CONUNPK	DFSCPY00
CONXRWEX	DFSCON20
CO1xxxx	DFSCVCO0
CO2xxxx	DFSCVCO0
CO3xxxx	DFSCVCO0
CO4xxxx	DFSCVCO0
CO5xxx	DFSCVCO0
CPIN1	DFSCPIN0
CPIN2	DFSCPIN0
CPRDATA	DFSDLDR0
CPRKEY	DFSDLDW0
CP0xxxx	DFSRCP00
CP1xxxx	DFSRCP10
CQ1xxxx	DFSCVCQ0
CSGxxxx	DFSCSGN0
CSPxxxx	DFSCSPI0
CSS0xxxx	DFSCSS00
CTIM0	DFSCTIM0
CTIM1	DFSCTIM0
CTLETXR	DFSFMOD0
CTRxxxx	DFSCTRN0
CTTA207	DFSCTTO0
CURxxxx	DFSICUR0
CVCSxxx	DFSCVCS0
CVEExxx	DFSCVEE0
CVEKxxx	DFSCVEK0
CVELxxx	DFSCVEL0
CVEN	DFSCVEN0
CVERxxxx	DFSCVER0
CVER1	DFSCVER0
CVER2	DFSCVER0
CVFxxxx	DFSCVEF0
CVHI	DFSCVHIO
CVHL0	DFSCVHL0
CVHN	DFSCVHN0
C2170xx	DFSC2170
C32xxxx	DFSC3270
DBFARDR1	DBFARDR0
DBFARDR2	DBFARDR0
DBFARDR3	DBFARDR0
DBFARDR4	DBFARDR0
DBFARDR5	DBFARDR0
DBFCHKP1	DBFCHKP0

SAVID	Module
DBFCHKP2	DBFCHKP0
DBFDLB10	DBFDLB00
DBFDLOG0	DBFARDR0
DBFGIVEB	DBFIRC10
DBFI	DBFAAFP0
DBFI	DBFCOPYR
DBFICL21	DBFICL20
DBFICL22	DBFICL20
DBFICL41	DBFICL40
DBFINTP0	DBFINTE0
DBFINTS0	DBFINTE0
DBFINTT0	DBFINTE0
DBFLKSCP	DBFDBAU0
DBFMLOP1	DBFMLOP0
DBFMLOP2	DBFMLOP0
DBFMLOP3	DBFMLOP0
DBFMLOP4	DBFMLOP0
DBFMLOP5	DBFMLOP0
DBFRESX0	DBFREXX0
DBFSTAP0	DBFERST0
DBFTIME0	DBFCHKP0
DBFTRACE	DBFTRAK0
DBFVLOC0	DBFMGXC0
DBFXSL00	DBFXSL30
DBFXWU00	DBFXWU30
DBH10	DFSDBH10
DBH4	DFSDBH40
DBOPEN	DFSSBHD0
DBRCETXR	DFSFMOD0
DCINIT	DFSIINB0
DDPxxxx	DFSIDDP0
DEBUGS	DFSDLDC0
DECRECTR	DFSDLDD0
DELSCAN	DFSDLDD0
DEPOPEN	DFSREP00
DFSAEXIT	DFSABND0
DFSAIMOD	DFSABND0
DFSALIN	DFS0ALIN
DFSAQMR	DFSAQMR0
DFSBLDAQ	DFSDBAU0
DFSBLSER	DFSMMUD0
DFSCFEL0	DFSCFEA0
DFSCHGAU	DFSDBAU0
DFSCMDL0	DFSICV50
DFSCM7C5	DFSCM7A0
DFSCNDC0	DFSCNXA0

SAVID	Module
DFSCTIM1	DFSCTIM0
DFSCTLETXR	DFSFMOD0
DFSDBDR1	DFSDBDR0
DFSDBRCXIT	DFSFMOD0
DFSDLGETXR	DFSFMOD0
DFSDPDB0	DFSDPRH0
DFSDPDE0	DFSDPRH0
DFSDPIO0	DFSDPRH0
DFSDPRQ0	DFSDPRH0
DFSDPTM0	DFSDPRH0
DFSDYAETXR	DFSFMOD0
DFSDYAETXR	DFSSDL20
DFSESETXR	DFSFMOD0
DFSFDEQ0	DFSFFRH0
DFSFDSCH	DFSFPRF0
DFSFENQ0	DFSFFRH0
DFSFIND	DFSPRE00
DFSFLAT0	DFSFUNL0
DFSFLRCX	DFSFUNL0
DFSFLRC0	DFSFUNL0
DFSFMODETXR	DFSFMOD0
DFSFREMN	DFSPRE00
DFSFRES0	DFSFFRH0
DFSGDSNM	DFSDBAU0
DFSGETMN	DFSPRE00
DFSICL21	DFSICL20
DFSICL22	DFSICL20
DFSICL31	DFSICL30
DFSICL61	DFSICL60
DFSICL72	DFSICL70
DFSICL73	DFSICL70
DFSICL74	DFSICL70
DFSICV41	DFSICV40
DFSICV51	DFSICV50
DFSIFBUF	DFSDBSM0
DFSIFMT0	DFSERA10
DFSIINIT	DFSKBDP0
DFSIINIT	DFSREP00
DFSINECB	DFSIDSP0
DFSIPCP0	DFSIPCP0
DFSIPCP1	DFSIPCP0
DFSIPCP2	DFSIPCP0
DFSIPEXT	DFSREP00
DFSIPOST	DFSIDSP0
DFSIPOST	DFSKBDP0
DFSIPOSX	DFSKBDP0

SAVID	Module
DFSIPOSX	DFSIDSP0
DFSIPOTC	DFSREP00
DFSIRABN	DFSIDSP0
DFSISDSW	DFSREP00
DFSISERW	DFSIDSP0
DFSISUSP	DFSREP00
DFSISWIT	DFSIDSP0
DFSIWAIT	DFSIDSP0
DFSIWAIT	DFSKBDP0
DFSIXCTL	DFSIDSP0
DFSIXMIT	DFSREP00
DFSIXMPT	DFSREP00
DFSKCKXM	DFSREP00
DFSKETXR	DFSKLS00
DFSKINPC	DFSREP00
DFSKLSDC	DFSKLS00
DFSKOSWT	DFSIDSP0
DFSKOSWT	DFSKBDP0
DFSKPOTL	DFSIDSP0
DFSKPOTL	DFSKBDP0
DFSKPTB0	DFSREP00
DFSKPXTC	DFSREP00
DFSKSETL	DFSREP00
DFSKSETL	DFSKBDP0
DFSKSGN0	DFSIDSP0
DFSKTECB	DFSREP00
DFSKTRMX	DFSIDSP0
DFSKXMSW	DFSREP00
DFSMSG00	DFSDBAU0
DFSNDXRF	DFSDBAU0
DFSNEP&O	DBFESCD
DFSNOF&O	DBFESCD
DFSNOTE	DFSPRE00
DFSOBND0	DFSABND0
DFSPLRD0	DFSPLDR0
DFSPOINT	DFSPRE00
DFSQCP10	DFSQCP00
DFSRCP20	DFSRST00
DFSRDSETXR	DFSFMOD0
DFSREAD	DFSPRE00
DFSRSTETXR	DFSFMOD0
DFSSBCAA	DFSSBCA0
DFSSBCAE	DFSSBCA0
DFSSBEV1	DFSSBEV0
DFSSBEV2	DFSSBEV0
DFSSBID1	DFSSBID0

SAVID	Module
DFSSBID3	DFSSBID0
DFSSBTD1	DFSSBTD0
DFSSCHDQ	DFSSCHQ0
DFSSCHEQ	DFSSCHQ0
DFSSTACK	DFSPRE00
DFSSTAP0	DFSRST00
DFSSTCETXR	DFSFMOD0
DFSTERM	DFSXRPS0
DFSTOAD0	DBFTOFN0
DFSTSTCN	DFSTST00
DFSTSTER	DFSTST00
DFSTSTFB	DFSTST00
DFSTSTSD	DFSTST00
DFSTSTTM	DFSTST00
DFSUARC0	DFSUARA0
DFSUARP0	DFSUARA0
DFSUIA10	DFSUCPA0
DFSUICA0	DFSUCP60
DFSUIDR0	DFSUCP60
DFSUIDU0	DFSUCP60
DFSUIIL0	DFSUCPA0
DFSUIIM0	DFSUCP60
DFSUIPR0	DFSUCPA0
DFSUIPU0	DFSUCPA0
DFSUIRV0	DFSUCP60
DFSUISN0	DFSUCPA0
DFSUISR0	DFSUCP60
DFSUISU0	DFSUCP60
DFSUIZB0	DFSUCPA0
DFSUIZM0	DFSUCPA0
DFSUTILA	DFSDBAU0
DFSV410A-SRB	DFSASK00
DFSV410A-SRB	DFSDASI0
DFSXLSD0	DFSKLSO0
DFSXRBLI	DFSXRBL0
DFSXRBLP	DFSXRBL0
DFSXRBLR	DFSXRBL0
DIRMAINT	DFSUTSG0
DISPRINT	DFSUTR20
DISPRINT	DFSUTR30
DLAMSG3303	DFSDLA00
DLASTAT	DFSDLA00
DLA3ACNT	DFSDLA30
DLA3LOG	DFSDLA30
DLA3MOVE	DFSDLA30
DLA3OPTL	DFSDLA30

SAVID	Module
DLA30000	DFSDLA30
DLA32000	DFSDLA30
DLBIxxxx	DFSDLBI0
DLDC	DFSDLDC0
DLDEFER	DFSRLP00
DLGETXR	DFSFMOD0
DLICALL	DFSUTR20
DLICALL	DFSUTR30
DLTRxxxx	DFSDLTR0
DMBDFND	DFSDBDR0
DMBDSCN	DFSDBDR0
DNSC1	DFSDNSC0
DOSETL	DFSDLDC0
DT1xxxx	DFSDN110
DYAETXR	DFSFMOD0
EACLLOCK	DBFEACL0
EACLSTOP	DBFEACL0
EACLUNLK	DBFEACL0
EDIT2980	DFS29800
EDTxxxx	DFSDN110
EDTxxxx	DFS36140
EEQE00	DFSSDLB0
EMSG2012	DBFEACL0
ENTRY36A	DFSIDPB0
ERA5xxxx	DFSERA50
ERSM3886	DBFERS20
ESPxxxx	DFSCESP0
ESSETXR	DFSFMOD0
ESSETXR	DFSIESI0
FAILURES	DFSUTR20
FBKENQ	DFSDLDW0
FD1xxxx	DFSCVFD0
FD2xxxx	DFSCVFD0
FELxxxx	DFSCFEA0
FEO1xxxx	DFSCFEO0
FEO2xxxx	DFSCFEO0
FEPxxxx	DFSCFEP0
FESxxxx	DFSCFES0
FEZ1	DFSCFEZ0
FINDRPST	DFSQRST0
FINDSTRT	DFSDLDW0
FINISHMG	DFSDBDR0
FIRSTREP	DFSUTR20
FIRSTREP	DFSUTR30
FIXDSG	DFSSDLB0
FIXDSG2	DFSSDLB0

SAVID	Module
FI1xxxx	DFSCVFI0
FLOCK00	DBFTOPU0
FN1xxxx	DFSCVFN0
FN2xxxx	DFSCVFN0
FN3xxxx	DFSCVFN0
FPDEFER	DFSRLP00
FPETXR	DFSFMOD0
FP1xxxx	DFSCVFP0
FP2xxxx	DFSCVFP0
FQ1xxxx	DFSCVFQ0
FQ2xxxx	DFSCVFQ0
FQ3xxxx	DFSCVFQ0
FQ4xxxx	DFSCVFQ0
FREEIT	DFSTRTN0
FREEWA	DFSDLDC0
FREWAREA	DFSRST00
FRSPC	DFSDLDD0
FTRxxxx	DFSFTRM0
FXCESS	DFSFXC30
FX1xxxx	DFSCVFX0
FX2xxxx	DFSCVFX0
FX3xxxx	DFSCVFX0
FX5xxxx	DFSCVFX0
FX6xxxx	DFSCVFX0
FY1xxxx	DFSCVFY0
FZ1xxxx	DFSCVFZ0
FZ2xxxx	DFSCVFZ0
FZ3xxxx	DFSCVFZ0
F12xxxx	DFSCVF10
F13xxxx	DFSCVF10
F14xxxx	DFSCVF10
GETELA	DFSTRTN0
GETLAT	DFSDLDC0
GETMCTB	DFSCRPV0
GETMSGA	DFSTRTN0
GETPARNT	DFSDLDW0
GETPDIR	DFSUTSA0
GETPTR	DFSDLDC0
GETRAP	DFSDLDC0
GETRPST	DFSRST00
GETRTNE	DFSRST00
GETSEG	DFSDLDW0
GETSYMT	DFSUTSR0
GETUNIQ	DFSDLDW0
GETWKA	DFSDLDW0
GLOCK00	DBFTOPU0

SAVID	Module
HCSUBRT2	DFSCVHC0
HCSUBRT3	DFSCVHC0
HDAI2000	DFSHDAI0
HDAI3000	DFSHDAI0
HDAI4000	DFSHDAI0
HDAI5000	DFSHDAI0
HDAI6000	DFSHDAI0
HDAI7000	DFSHDAI0
HDAI8000	DFSHDAI0
HDAI9000	DFSHDAI0
HDC4xxxx	DFSHDC40
HD0-DATE-TIME	DFSCVHD0
HE0-DATE-TIME	DFSCVHE0
HF0-DATE-TIME	DFSCVHF0
HGU1TMR	DBFHGU10
HM0-DATE-TIME	DFSCVHM0
HTRKBUF	DFSHPTK0
ICIOOPCK	DFSICIO0
ICIOQMGR	DFSICIO0
ICIOVTAM	DFSICIO0
ICLHEXRW	DFSICLH0
ICLS1	DFSICLS0
ICL40X10	DFSIC440
ICL40X20	DFSIC440
ICL40X30	DFSIC440
ICL40X30	DFSIC460
ICL7FIND	DFSICL70
ICRERTNE	DFSRST00
IDENTRTN	DFSRST00
IDExxxx	DFSIIDE0
IDSONRTN	DFSRRA00
IENxxxx	DFSIIEN0
IINL-1.2	DFSIINL0
IMSxxxx	DFSIIMS0
INITPSDB	DFSDLDW0
INTDBLNT	DFSSDL30
INTDVBH2	DFSSDL30
INTERA20	DFSSDL30
INTERNALTRACE	DFSINTRA
INTFXC50	DFSSDL30
INVDBRC	DFSDBDR0
IO3xxxx	DFSIIO30
IRACxxxx	DFSIRAC0
ITLJxxxx	DFSUTLJ0
I15xxxx	DFSII150
I779090	DFSI7770

SAVID	Module
JCBSET	DFSDBLM0
KEYCALL	DFSDBDR0
KLSMETXR	DFSFMOD0
LAHxxxx	DFSDN070
LA3xxxx	DFSDLA30
LDSECRTN	DFSRST00
LEOVSER	DFSFPLG0
LMOxxxx	DFSCLMO0
LMSRBEP	DFSLMGR0
LNK1300	DFSILNK0
LOGDLT	DFSDLDD0
LR1xxxx	DFSICLR0
LSDBSRCH	DFSDLDW0
MARKSDB	DFSDLDD0
MBDxxxx	DFSCMBD0
MBJxxxx	DFSCMBJ0
MBPxxxx	DFSCMBP0
MBTxxxx	DFSCMBT0
MBUxxxx	DFSCMBU0
MBXxxxx	DFSCMBX0
MCPxxxx	DFSCMCP0
MCTxxxx	DFSCMCT0
MCXxxxx	DFSCMCX0
MC0xxxx	DFSCMC00
MC1xxxx	DFSCMC10
MC2xxxx	DFSCMC20
MC4xxxx	DFSCMC40
MC5xxxx	DFSCMC50
ME0xxxx	DFSME000
ME127	DFSME127
MFSWORK	DFSCFEO0
MLAxxxx	DFSCMLA0
MLOPAUTH	DBFMLOP0
MMPxxxx	DFSCMMP0
MMUD	DFSMMUD0
MMUxxxx	DFSCMMU0
MMXxxxx	DFSCMMX0
MM20xxxx	DFSCMM20
MNTB0	DFSMNTB0
MODETXR	DFSFMOD0
MONREPT	DFSUTR20
MOVRTNE	DFSRST00
MPLC	DFSMMLC0
MPOS1	DFSMPOS0
MPPENQ00	DFSSMSC0
MP1xxxx	DFSIMP10

SAVID	Module
MP2xxxx	DFSIMP20
MR0xxxx	DFSCMR00
MSAxxxx	DFSCMSA0
MSBxxxx	DFSCMSB0
MSCREP	DFSUTR20
MSDxxxx	DFSCMSD0
MSExxxx	DFSCMSE0
MSFxxxx	DFSCMSF0
MSGRTNE	DFSRRA00
MSHxxxx	DFSCMSH0
MSIxxxx	DFSCMSI0
MSMxxxx	DFSCMSM0
MSQREP	DFSUTR20
MSSxxxx	DFSCMSS0
MSUMREP	DFSUTR20
MSVxxxx	DFSCMSV0
MSWxxxx	DFSCMSW0
MSYxxxx	DFSCMSY0
MS0xxxx	DFSCMS00
MS1	DFSCMS00
MS6xxxx	DFSCMS60
MS7xxxx	DFSCMS70
MS8xxxx	DFSCMS80
MTMA	DFSMTMA0
MTMWORKA	DFSMTMA0
NDXxxxx	DFSINDX0
NEWAWE	DFSDBDR0
NOCORE	DFSERA20
NPKxxxx	DFSUNPK0
NPKxxxx	DFSUTLW0
NSCxxxx	DFSDNSC0
NS2xxxx	DFSDNS20
NS3xxxx	DFSDNS30
NXAxxxx	DFSCNXA0
N01xxxx	DFSDN010
N02xxxx	DFSDN020
N03xxxx	DFSDN030
N04xxxx	DFSDN040
N05xxxx	DFSDN050
N06xxxx	DFSDN060
N07xxxx	DFSDN070
N08xxxx	DFSDN080
N09xxxx	DFSDN090
N10xxxx	DFSDN100
N11xxxx	DFSDN110
N12xxxx	DFSDN120

SAVID	Module
N13xxxx	DFSDN130
N14xxxx	DFSDN140
N15xxxx	DFSDN150
N16xxxx	DFSDN160
N17xxxx	DFSDN170
N18xxxx	DFSDN180
N19xxxx	DFSDN190
N23xxxx	DFSDN230
N24xxxx	DFSDN240
N25xxxx	DFSDN250
N260xxxx	DFSDN260
N27xxxx	DFSDN270
N28xxxx	DFSDN280
N52xxxx	DFSDN520
N53xxxx	DFSDN530
N54xxxx	DFSDN540
OLICCALL	DFSDBAU0
OPCLSRTN	DFSRST00
OPENACB	DFSIINB0
OVHDREPT	DFSUTR30
O779090	DFSO7770
PAAxxxx	DFSUPAA0
PABxxxx	DFSUPAB0
PACxxxx	DFSUPAC0
PADxxxx	DFSUPAD0
PAFxxxx	DFSUPAF0
PAGxxxx	DFSPAGE0
PAGxxxx	DFSUPAG0
PAHxxxx	DFSUPAH0
PAJxxxx	DFSUPAJ0
PAKxxxx	DFSUPAK0
PAMxxxx	DFSUPAM0
PANxxxx	DFSUPAN0
PAPxxxx	DFSUPAP0
PAQxxxx	DFSUPAQ0
PARTEDIT	DFSSUT04
PARxxxx	DFSUPAR0
PASxxxx	DFSUPAS0
PATxxxx	DFSUPAT0
PAUxxxx	DFSUPAU0
PAVxxxx	DFSUPAV0
PAWxxxx	DFSUPAW0
PAXxxxx	DFSUPAX0
PAYxxxx	DFSUPAY0
PA6xxxx	DFSUPA60
PA7xxxx	DFSUPA70

SAVID	Module
PA8xxxx	DFSUPA80
PA9xxxx	DFSUPA90
PBAxxxx	DFSUPBA0
PBBxxxx	DFSUPBB0
PBExxxx	DFSUPBE0
PBFxxxx	DFSUPBF0
PBHxxxx	DFSUPBH0
PBJxxxx	DFSUPBJ0
PBKxxxx	DFSUPBK0
PBMxxxx	DFSUPBM0
PBNxxxx	DFSUPBN0
PBOxxxx	DFSUPBO0
PBZxxxx	DFSUPBZ0
PB0xxxx	DFSUPB00
PB1xxxx	DFSUPB10
PB2xxxx	DFSUPB20
PB3xxxx	DFSUPB30
PB5xxxx	DFSUPB50
PB6xxxx	DFSUPB60
PB7xxxx	DFSUPB70
PCC20ESTAE	DFSPCC20
PCC20TIMER	DFSPCC20
PDMxxxx	DFSDPDM0
PE1CPINV	DFS1CINV
PE1CPPUR	DFS1CPUR
PE1PINV	DFS1PINV
PE1PPUR	DFS1PPUR
PE2CORD	DFS2CORD
PE2ORDR	DFS2PORD
PE3CPPUR	DFS3CPUR
PE3PPUR	DFS3PPUR
PE4CNINQ	DFS4CNAM
PE4CODEL	DFS4CDEL
PE4COINQ	DFS4CINQ
PE4CORDR	DFS4CNEW
PE4NINQ	DFS4PNAM
PE4ODEL	DFS4PDEL
PE4OINQ	DFS4PINQ
PE4ORDL	DFS4PNEW
PGMBYREG	DFSUTR20
PIXxxxx	DFSPIXT0
POLxxxx	DFSIPOL0
PRELDRTN	DFSRRA00
PRIOROLD	DFSULG10
PRNTPDIR	DFSUTL40
PRNTPDIR	DFSUTSK0

SAVID	Module
PROCDVCT	DFSUTSK0
PROCINDX	DFSUTSK0
PROCPDIR	DFSUTSK0
PROFILE	DFSUTR20
PROGIO	DFSUTR20
PROGIO	DFSUTR30
PROGSUM	DFSUTR20
PROXO	DFSISI00
PRTMSG	DFSUACB0
PUT	DFSSBI00
QBFM	DFSQBFM0
QENTRTNE	DFSRST00
QFIXxxxx	DFSQFIX0
QLOG	DFSQLOG0
QMGR	DFSQMGR0
QRSTxxxx	DFSQRST0
QUEI	DFSQUEI0
RAPxxxx	DFSWRAP0
RCX	DFSRRC10
RDIxxxx	DFSTRD10
RDSETXR	DFSFMOD0
REGIWAIT	DFSUTR20
REGNSUM	DFSUTR20
RELLAT	DFSDLDC0
RELOC	DFSUTLV0
RELRPST	DFSRST00
REPOSDL	DFSIDP10
REPOSDL	DFSIDP70
REPOSDL	DFSIDP80
REPOSDL	DFSIDP90
REPOSDL	DFSIDPB0
RESPTYP	DFSIDPB0
RETPOST	DFSFXC10
RPLENQ	DFSDLDR0
RPSBW000	DFSFXC50
RPSB1000	DFSFXC50
RRA1	DFSRRA10
RRA2	DFSRRA20
RRA5	DFSRRA50
RSAxxxx	DFSCRSA0
RSBxxxx	DFSCRSB0
RSCxxxx	DFSCRSC0
RSDxxxx	DFSCRSD0
RSExxxx	DFSCRSE0
RSFxxxx	DFSCRSF0
RSHxxxx	DFSCRSH0

SAVID	Module
RSMxxxx	DFSCRSM0
RSNxxxx	DFSCRSN0
RSOxxxx	DFSCRSO0
RSRxxxx	DFSCRSR0
RSSxxxx	DFSCRSS0
RSTETXR	DFSFMOD0
RSTxxxx	DFSCRST0
RSUxxxx	DFSCRSU0
RSVxxxx	DFSCRSV0
RSWxxxx	DFSCRSW0
RSXxxxx	DFSCRSX0
RS1xxxx	DFSCRS10
RS2xxxx	DFSCRS20
RS4xxxx	DFSCRS40
RS5xxxx	DFSCRS50
RS6xxxx	DFSCRS60
RS7xxxx	DFSCRS70
RS8xxxx	DFSCRS80
RW0xxxx	DFSARW00
R2Ixxxx	DFSCR2I0
R2Kxxxx	DFSCR2K0
R2Yxxxx	DFSCR2Y0
R2Zxxxx	DFSCR2Z0
SAM08	DFSSAM08
SCANCLBB	DFSIDP70
SCANCLBB	DFSIDP90
SCANDMB	DFSDLDD0
SCANRPST	DFSQRST0
SCKC	DFSBSCK0
SEAxxxx	DFSCSEA0
SEGLOCK	DBFMGXC0
SEJxxxx	DFSCS3J0
SELECT	DFSERA10
SENDMSG	DFSFXC50
SETDMBD	DFSDBDR0
SETLIM	DFSDLDC0
SETPPDE	DBFVSOP0
SETSCALL	DFSDLA00
SIGNONRT	DFSRST00
SLOG	DFSSMSC0
SMIxxxx	DFSISMI0
SMSC1000	DFSSMSC0
SMSC2000	DFSSMSC0
SMSC3000	DFSSMSC0
SMSC4000	DFSSMSC0
SMSC5000	DFSSMSC0

SAVID	Module
SMSC6000	DFSSMSC0
SMSC7000	DFSSMSC0
SMSC8000	DFSSMSC0
SMSC9000	DFSSMSC0
SPINRTNE	DFSRRA00
SPYVALRT	DFSRRA00
SRLxxxx	DFSCRSL0
STAEXIT	DBFFATC0
STCETXR	DFSFMOD0
STGFMT	DFSERA10
STIMREXT	DBFFATC0
STOWCODE	DFSUTSH0
STRTSRV0	DFSRST00
STRTSURV	DFSRLP00
ST3xxxx	DFSIST30
ST4xxxx	DFSIST40
SUBxxxx	DFSCSUB0
SUBxxxx	DFSISUB0
SUDSG	DFSDLDC0
SURVAL00	DFSRST00
SURVST00	DFSRST00
SVCRTNE	DFSRRA00
SYM-TBL-INSRT	DFSUTSR0
SYNCLOW	DFSFXC50
S01xxxx	DFSDS010
S02xxxx	DFSDS020
S03xxxx	DFSDS030
S04xxxx	DFSDS040
S050xxx	DFSDS050
S06xxxx	DFSDS060
S07xxxx	DFSDS070
S3Gxxxx	DFSCS3G0
S3Pxxxx	DFSCS3P0
S3Qxxxx	DFSCS3Q0
S37xxxx	DFSS3741
S7Axxxx	DFSCS7A0
S7Bxxxx	DFSCS7B0
S7Cxxxx	DFSCS7C0
S7Dxxxx	DFSCS7D0
S7Gxxxx	DFSCS7G0
S7Ixxxx	DFSCS7I0
S7Lxxxx	DFSCS7L0
S7Pxxxx	DFSCS7P0
S7Txxxx	DFSCS7T0
S7Uxxxx	DFSCS7U0
S7770xx	DFSS7770

SAVID	Module
TBL60	DFSCVH60
TBL70	DFSCVH70
TCFETXR	DFSFMOD0
TER0	DFSESI30
TER0	DFSIESI0
TIMB	DFSCTIM0
TLAxxxx	DFSUTLA0
TLBxxxx	DFSUTLB0
TLCxxxx	DFSUTLC0
TLDxxxx	DFSUTLD0
TLExxxx	DFSUTLE0
TLFxxxx	DFSUTLF0
TLGxxxx	DFSUTLG0
TLHxxxx	DFSUTLH0
TLRxxxx	DFSUTLW0
TLSxxxx	DFSUTLW0
TLTxxxx	DFSUTLT0
TLUxxxx	DFSUTLU0
TLVxxxx	DFSUTLV0
TLXxxxx	DFSUTLX0
TLYxxxx	DFSUTLY0
TLZxxxx	DFSUTLZ0
TL3xxxx	DFSUTL30
TL7xxxx	DFSUTL70
TL8xxxx	DFSUTL80
TL9xxxx	DFSUTL90
TODMSG	DFSTVER0
TRACE	DFSSMSC0
TRACERTN	DFSDBAU0
TRACERTN	DFSSDLB0
TRANQNG	DFSUTR20
TRIALUSE	DFSRCHB0
TSBxxxx	DFSUTSB0
TSCxxxx	DFSUTSC0
T0Ixxxx	DFSUT0I0
T0Txxxx	DFSUT0T0
T03xxxx	DFSUT030
T04xxxx	DFSUT040
T07xxxx	DFSUT070
T1TABLE	DBFULTA0
T15xxxx	DFSUT150
T16xxxx	DFSUT160
T17xxxx	DFSUT170
T19xxxx	DFSUT190
T2ENTRY	DBFULTA0
T20xxxx	DFSUT200

SAVID	Module
T26xxxx	DFSUT260
T28xxxx	DFSUT280
T29xxxx	DFSUT290
UNHKCHLD	DFSDLDD0
UNHKTWIN	DFSDLDD0
UNPACK	DFSERA10
UPDPARPT	DFSDLDD0
UPDPTRHD	DFSDLDD0
UPDRAPTR	DFSDLDD0
UPRxxxx	DFSUPRT0
UTSO	DFSUTSO0
UTSO-LOCMSG	DFSUTSO0
UTSO-SORT	DFSUTSO0
VBHDSHR	DFSDVBH0
VBHHOTS	DFSDVBH0
VCB1xxxx	DFSCVCB0
VCB2xxxx	DFSCVCB0
VCCxxxx	DFSCVCC0
VCDxxxx	DFSCVCD0
VCExxxx	DFSCVCE0
VCFxxxx	DFSCVCF0
VCGxxxx	DFSCVCG0
VCIxxxx	DFSCVCI0
VCLxxxx	DFSCVCL0
VCNxxxx	DFSCVCN0
VCOxxxx	DFSCVCO0
VCPxxxx	DFSCVCP0
VCQxxxx	DFSCVCQ0
VCRxxxx	DFSCVCR0
VCTxxxx	DFSCVCT0
VCV	DFSCVCV0
VDAxxxx	DFSCVDA0
VDBxxxx	DFSCVDB0
VDCxxxx	DFSCVDC0
VDDxxxx	DFSCVDD0
VDExxxx	DFSCVDE0
VDIxxxx	DFSCVDI0
VD2xxxx	DFSCVCD0
VEA1	DFSCVEA0
VEA2	DFSCVEA0
VEA3	DFSCVEA0
VEBxxxx	DFSCVEB0
VECxxxx	DFSCVEC0
VEDxxxx	DFSCVED0
VEGxxxx	DFSCVEG0
VEHxxxx	DFSCVEH0

SAVID	Module
VEIxxxx	DFSCVEI0
VEJxxxx	DFSCVEJ0
VEMxxxx	DFSCVEM0
VEOCxxx	DFSCVEO0
VEPxxxx	DFSCVEP0
VEQxxxx	DFSCVEQ0
VESxxxx	DFSCVES0
VES1	DFSCVES0
VFAxxxx	DFSCVFA0
VFCxxxx	DFSCVFC0
VFDxxxx	DFSCVFD0
VFExxxx	DFSCVFE0
VFGxxxx	DFSCVFG0
VFHxxxx	DFSCVFH0
VFIxxxx	DFSCVFI0
VFJxxxx	DFSCVFJ0
VFMxxxx	DFSCVFM0
VFNxxxx	DFSCVFN0
VFPxxxx	DFSCVFP0
VFQxxxx	DFSCVFQ0
VFR0xxx	DFSCVFR0
VFR3xxx	DFSCVFR0
VFSxxxx	DFSCVFS0
VFXxxxx	DFSCVFX0
VFXxxxx	DFSCVFX0
VFYxxxx	DFSCVFY0
VFZxxxx	DFSCVFZ0
VF1xxxx	DFSCVF10
VF3xxxx	DFSCVF30
VF4xxxx	DFSCVF40
VF5xxxx	DFSCVF50
VGAxxxx	DFSCVGA0
VGBxxxx	DFSCVGB0
VGCxxxx	DFSCVGC0
VGDxxxx	DFSCVGD0
VGExxxx	DFSCVGE0
VGFxxxx	DFSCVGF0
VGGxxxx	DFSCVGG0
VGHxxxx	DFSCVGH0
VGIxxxx	DFSCVGI0
VGJxxxx	DFSCVGJ0
VGKxxxx	DFSCVGK0
VGLxxxx	DFSCVGL0
VGMxxxx	DFSCVGM0
VGNxxxx	DFSCVGN0
VGOxxxx	DFSCVGO0

SAVID	Module
VGO1	DFSCVGO0
VGO2	DFSCVGO0
VGPxxxx	DFSCVGP0
VHA0	DFSCVHA0
VHB0	DFSCVHB0
VHCxxxx	DFSCVHC0
VHH0	DFSCVHH0
VHP	DFSCVHP0
VHQ	DFSCVHQ0
VHR	DFSCVHR0
VHS	DFSCVHS0
VHT	DFSCVHT0
VHX	DFSCVHX0
VIRDRTST	DFSDLDD0
VIRPRTST	DFSDLDD0
VJBxxxx	DFSCVJB0
VJKxxxx	DFSCVJK0
VJLxxxx	DFSCVJL0
VJMxxxx	DFSCVJM0
VJOxxxx	DFSCVJO0
VJRxxxx	DFSCVJR0
VLOG	DFSCVLG0
VMSTAT	DFSUTR30
VRAxxxx	DFSCVRA0
VRBxxxx	DFSCVRB0
VRCxxxx	DFSCVRC0
VRC1	DFSCVRC0
VRC2	DFSCVRC0
VRC3	DFSCVRC0
VRC4	DFSCVRC0
VRC5	DFSCVRC0
VRFxxxx	DFSCVRF0
VRGxxx	DFSCVRG0
VRHxxxx	DFSCVRH0
VRJxxxx	DFSCVRJ0
VRKxxxx	DFSCVRK0
VRLxxxx	DFSCVRL0
VRMxxxx	DFSCVRM0
VRNxxxx	DFSCVRN0
VROxxxx	DFSCVRO0
VRO1	DFSCVRO0
VRO2	DFSCVRO0
VRO3	DFSCVRO0
VRPxxxx	DFSCVRP0
VRRxxxx	DFSCVRR0
VRR1	DFSCVRR0

SAVID	Module
VRR2	DFSCVRR0
VRR3	DFSCVRR0
VRSxxxx	DFSCVRS0
VRTxxxx	DFSCVRT0
VRYxxxx	DFSCVRY0
VRY1	DFSCVRY0
VRZxxxx	DFSCVRZ0
VRZ1	DFSCVRZ0
V36xxxx	DFS36010
XC50BKRT	DFSFXC50
XC50FSRT	DFSFXC50
XLGIDBRC	DFSXLGI0
XLICSTAE	DFSXLIC0
XMRESUME	DFSREP00
327xxxx	DFSC3270

Appendix E. Dependency Keywords

Dependency keywords can be used with the keyword string to select only those APARs that apply to a certain environment. These can be particularly useful when a search yields a large number of hits and you are almost certain that the program failure occurs only in a specific environment.

Keyword	Environment	Keyword	Environment
D/CICS	CICS	D/TRKREC	Track Recovery
D/CONVPROC	Conversational	D/TWX	Teletype
	Processing	D/UCF	Utility Control Facility
D/FP	Fast Path	D/VSAM	VSAM
D/GSAM	GSAM	D/VTAM	VTAM
D/HDAM	HDAM	D/1050	1050 Device Type
D/HIDAM	HIDAM	D/2260	2260 Device Type
D/HISAM	HISAM	D/2740	2740 Device Type
D/HSAM	HSAM	D/2741	2741 Device Type
D/IRLM	MS/VS Resource Lock Manager	D/2770	2770 Device Type
D/LU6	VTAM LU 6	D/2780	2780 Device Type
	(Intersystem Communication)	D/2980	2980 Device Type
D/MFS	Message Format Services	D/3270	3270 Large Screen
D/MSC	Multiple System Coupling	D/3270L	3270 Local
D/MVS	MVS	D/3270R	3270 Remote
D/None	No dependencies	D/3274	3274 Device Type
D/NTO	Network Terminal Option	D/3275	3275 Device Type
D/OSAM	OSAM	D/3276	3276 Device Type
D/SB	Sequential Buffering	D/3278	3278 Device Type
D/SECINDX	Secondary Index	D/3279	3279 Device Type
D/SHISAM	Simple HISAM	D/3284	3284 Device Type
D/SLU1	VTAM Type SLU 1	D/3286	3286 Device Type
D/SLU2	VTAM Type SLU 2	D/3287	3287 Device Type
D/SLU4	VTAM Type SLU 4	D/3350	3350 Device Type
D/SLU P	VTAM Type SLU P	D/3375	3375 Device Type
D/SYSGEN	PTFs that should be	D/3380	3380 Device Type
	applied prior to SYSGEN	D/3600	3600 Device Types
D/SYS3	System/3	D/3790	3790 Device Types
D/SYS7	System/7		

Appendix F. Module-to-Waiting-Resource List

This table lists most waiting conditions or resources associated with an IMS task.

Module Name	Waiting Condition or Resource
DBFCHKP0	Wait for DEDB close
	Wait for MSDB checkpoint
DBFCPID0	Wait for sync latch
DBFDBDL0	Wait for I/O
DBFDBDP0	Wait for I/O
DBFERST0	Wait for all Fast Path forward recoveries to complete
	Wait for XRF area preopen to complete
DBFFATC0	Wait for asynchronous request
	Wait for work to process
DBFFORI0	Wait for output thread
DBFHRTR0	Wait for output message
DBFINTE0	Wait for DEDB close
DBFMIOS0	Wait for VSAM placeholder
DBFPVTS0	Wait for private pool services
DBFRMRC0	Wait for IPAGE storage
DBFSYN00	Wait for /STOP REGION ABDUMP to complete
DBFVOCI0	Wait for VSO services
DBFVXCS0	Wait for XES services requests
DBFXCGL0	Wait for resource latch
DFSAOS10	Wait for OSAM I/O
	Wait for pending EOV
DFSAOS60	Wait for I/O
	Wait for an IOSB
	Wait for I/O and EOV synchronization
DFSAOS70	Wait for page fix operation
DFSAOS80	Wait for I/O
DFSASK00	Wait for copy function 4 to complete
	Wait for /STOP REGION ABDUMP to complete
	Wait for dependent region termination
DFSBML00	Wait for APSB latch
DFSCLM00	Wait for a latch
DFSCFRT0	Wait for a DCB to be freed
	Wait for a read operation to complete

DFSCMC00	Wait for I/O to complete
DFSCMC50	Wait for a page fix operation to complete
DFSCMD30	Wait for command processing to complete
DFSCMTI0	Wait for send message/AOI command requests
DFSCNS00	Wait for open/close data set requests
DFSCPCP0	Wait for a checkpoint operation to complete
	Wait for a purge operation to complete
DFSCPDM0	Wait for READ to inactive ACBLIB to complete
DFSCPY00	Wait for /STOP REGION ABDUMP
DFSCSS00	Wait for storage requests
DFSCST00	Wait for control task service requests
DFSCS7B0	Wait for a QCB
DFSCS7L0	Wait for a QCB to be loaded
DFSCVEH0	Wait for a 3270 printer
DFSCVEI0	Wait for a 3270 copy operation to complete
DFSCVEQ0	Wait for a 3270 copy operation to complete
DFSDBAU0	Wait for MPP to pseudoabend
DFSDBDR0	Wait for MPP to pseudoabend
DFSDBH10	Wait for PI enqueue OSAM format operation
	Wait for coupling facility structure or connection failure processing to complete
DFSDBH20	Wait for write I/O
	Wait for read I/O
	Wait for coupling facility structure or connection failure processing to complete
	Wait for coupling facility data transfer
DFSDBH30	Wait for OSAM write
	Wait for wait common
	Wait for coupling facility structure or connection failure processing to complete
	Wait for coupling facility data transfer
DFSDBH40	Wait for WTOR response
DFSDBLM0	Wait for RLM reconnect
DFSDBLR0	Wait for I/O
DFSDCFR0	Wait for in-flight OSAM and/or VSAM coupling facility requests to complete
	Wait for next XES event to be presented
DFSDDLE0	Wait for I/O
DFSDDLS0	Wait for WTOR response
DFSDHD00	Wait for data block serialization
DFSDLA00	Wait for DFSDBAU0

DFSDLA30	Wait for input (PWFI)
DFSDLOC0	Wait for enqueue of ACB/DCB
	Wait for VSAM I/O to complete
	Wait for log to free
	Wait for coupling facility data delete operation
DFSDLR00	Wait for PI enqueue
	Wait for I/O
DFSDMG10	Waiting for a message (GMSG DL/I call)
DFSDVBH0	Wait for PSTs in processing buffer invalidates
	Wait for coupling facility data delete operation
DFSDVSM0	Wait for VSAM request to be issued
	Wait for logical record to be enqueued/dequeued
	Wait for VSAM UPAD exit
	Wait for ISWITCH to complete
	Wait for log write
	Wait for log write ahead
	Wait for VSAM JRNAD exit
	Wait for coupling facility structure or connection failure processing to complete
DFSDYA00	Wait for dynamic allocation service requests
DFSESI30	Wait for additional AWE requests
DFSFCTT0	Wait for modify or terminate command
DFSFDLB0	Wait for primary log I/O
	Wait for secondary log I/O
	Wait for WADS I/O
DFSFDLG0	Wait for AWE to be placed on queues
DFSFDLL0	Wait for RDS OPEN
	Wait for log buffer
DFSFDLS0	Wait for log STIMER interval to elapse
	Wait for work to do
DFSFDMP0	Wait for WTOR response
DFSFDPL0	Wait for RDS I/O
	Wait for control function request
	Wait for log buffer
DFSFFET0	Waiting to enqueue on a resource
	Wait for a READ macro to complete
DFSFFRH0	Waiting to enqueue on a resource
DFSFLLG0	Wait for log buffer

	Wait for all log requestors if abnormal termination in progress
	Wait for log write ahead to complete
	Wait for FEOV/CLOSE to complete
	Wait for log latch
DFSFMOD0	Wait for attach task request
	Wait for control TCB to terminate
DFSFPMM0	Waiting to dequeue on a resource
DFSFSTM0	Wait for the DL/I subordinate address space to complete resource cleanup
DFSFXC30	Wait for input (WFI)
DFSFXC50	Wait for I/O error handling EEQE notifies to complete
DFSICA10	Wait for external subsystem processing
DFSICL20	Wait for DBRC/IRLM ISERWAIT
DFSICL40	Wait for DBRC/IRLM ISERWAIT
DFSICLW0	Wait for DBRC ISERWAIT
DFSICV30	Wait for DFSOCMT0 to acquire block mover latch
DFSIC420	Wait for physical logger to perform /STA or /STO OLDS processing
DFSIDPA0	Wait for the DL/I subordinate address space to return buffer handler statistics
DFSIDSP0	Wait for ECB initialization
DFSIESI0	Wait for additional AWE requests
DFSIMNT0	Wait for DB Monitor log open
	Wait for IMS Monitor work areas to be deleted
DFSIOPH0	Wait for status from physical logger
DFSIRST0	Wait for restart request
DFSISMN0	Wait for storage
DFSIWAIT	Batch WAIT and dispatch routine
DFSLATE0	Wait for latch
DFSLMGR0	Wait for DFSDBAU0 to awaken DFSLMGR0
DFSMDA00	Wait for latch
DFSMNTR0	Wait for the DL/I subordinate address space to return buffer handler statistics
DFSOCMT0	Wait for DFSICV31 to complete processing
DFSPIEX0	Wait for PI enqueue/dequeue lock
DFSPR000	Wait for control service to complete
DFSQBFM0	Wait for a Message queue buffer to become available
	Wait for a Message queue buffer to be written out
DFSQC010	Wait for Shutdown checkpoint to complete
DFSQC030	Wait for QMGR or QBSL latch
DFSQC060	Wait for a logical queue destination

DFSQC070	Wait for 4001 Checkpoint to complete
DFSQMGR0	Wait for XRF message queue merge to complete
DFSRCP00	Wait for inactive dependent regions
	Wait for database close
	Wait for PST dequeue
	Wait for message queues
	Wait for log write
	Wait for log latch
	Wait for log close
DFSRDS00	Wait for restart data set related requests
DFSRDSH0	Wait for WTOR response
DFSRLD00	Wait for system data sets to be opened
	Wait for OSAM I/O
	Wait for restart backouts to complete
DFSRLP00	Wait for all backouts to complete
	Wait for complete of start surveillance logic
	Wait for XRF database preopens to complete
	Wait for XRF session initiation to complete
DFSRMDD0	Wait for IPAGE storage
DFSRMDM0	Wait for IPAGE storage
DFSRMPD0	Wait for IPAGE storage
	Wait for READ to complete to active ACBLIB
DFSRMPS0	Wait for IPAGE storage
DFSRMS00	Wait for storage
DFSRRA00	Wait for abend completion
	Wait for operator reply on signon
DFSRRC10	Wait for ATTACH
	Wait for ABEND
DFSRST00	Wait for checkpoint to complete
	Wait for the DL/I subordinate address space to close databases
	Wait for NOTIFY to control task that restart is complete
	Wait for system data sets to be opened
DFSSBCR0	Wait for OSAM read I/O
DFSSBEV0	Wait to manipulate a sequential buffering SDSG subsystem chain
DFSSBIO0	Wait for OSAM read I/O
DFSSBMP0	Wait for write to log
	Wait for enqueue BMP on SUBQ

	Wait for takeover to complete
	Wait until all Fast Path locks have been reacquired and all required Fast Path DBRC reverifies have been sent
DFSSBTD0	Wait to manipulate a sequential buffering SDSG subsystem chain
DFSSDLB0	Wait for Fast Path to reverify databases after an IRLM failure
DFSSDL20	Wait to terminate the job step TCB for the DL/I subordinate address space (normal termination)
DFSSMIC0	Wait for LSO daughter TCB attach
DFSSMSC0	Wait for write to log
	Wait for enqueue PST
DFSSTAT0	Wait for the DL/I subordinate address space to return buffer handler statistics
DFSSUSX0	Waiting until IRLM can grant request
DFSTERM0	Wait for the DL/I subordinate address space to close databases
	Wait for the DL/I subordinate address space to terminate (normal shutdown)
	Wait for trace logging to complete
DFSTMAD0	Wait for online change to complete
	Wait after enqueuing PST on SUBQ
	Wait for notify (in XRF environment)
	Wait for write to log
DFSTMCD0	Wait for Connect/Disconnect requests (normal wait for work)
DFSUACB0	Wait for I/O
DFSUCMN0	Wait for SORT
DFSUCPA0	Wait for IMS utility (UCF)
DFSUCP60	Wait for IMS utility (UCF)
DFSUICC0	Wait for /STOP REGION ABDUMP to complete
DFSURG10	Wait for SORT
DFSURUL0	Wait for I/O
DFSUSE00	Wait for interest in a structure
DFSV4200	Wait for page fix
DFSXCIC0	Wait for the DBRC subordinate address space to connect to the control region
	Wait for the DL/I subordinate address space to connect to the control region
DFSXDL10	DL/I subordinate address space wait for DFSIINS0 to complete in the control region
	DL/I subordinate address space wait for DFSIIND0 to complete in the control region
DFSXIOB0	Wait for WTOR response
DSPRSV00	Wait for RECON data sets
DXRRL020	Wait for an IRLM operator command
DXRRL070	Wait for IRLM SRBs to complete
DXRRL080	Wait for IRLM storage request

Exception: IRLM waiting subtasks are normally waiting and are not associated with an IMS task. They are waiting to perform a task-related service for the Internal Resource Lock Manager.

Appendix G. Locating IMS Blocks and Work Areas Using Load List Elements

IMS loads IMS blocks and work areas using the IMS IMODULE facility. IMS generates a load list element from which you can obtain the unique name and location of each work area. Table 194 is a list of the areas that appear formatted as the load list in an IMS control region dump. Global areas are in the common storage area (CSA).

Table 194. Load List Areas

Load List Name	IMS Block/Work Area	Pool Type
DFSABSxx	Abend Diagnostic Area, xx=PST number	Global
DFSBFSPP	DL/I Buffer Handler Pool	Global
DFSBLK0x	SCD, x=same as nucleus suffix	Global
DFSBWLOG	BG Write Log Work Area	Local
DFSCBTHD	Control block table header that points to the storage pools defined in DFSCBT00	Global[1 on page 557]
DFSCBT10	Storage pool headers for the pools defined in DFSCBT00	Global[1 on page 557]
DFSDLWxx	Retrieve Work Area, xx=PST number	Global
DFSDMBRS	Resident DMBs	Global
DFSDSET	OLDS Data Set Entry Table	Local
DFSEOVOS	OSAM DCB Work Area	Global
DFS01FXL	Fixlist for OSAM I/O Driver	Local
DFSINTRS	Resident Intent Lists	Global
DFSIPB	Initialization Parameter Block	Local
DFSISIT	Ident Table and ISI Storage	Global
DFSLCD	Logger LCD	Global[2 on page 557]
DFSLCDST	IMS Monitor Logger LCD	Global
DFSLLOG	X'06' and X'42' Log Records	Local
DFSLOCP	Storage Management Local Pool	Local
DFSLOGxx	Log Work Area, xx=PST number	Global
DFSLXBC	Link Extension Blocks for MSC CTC	Global
DFSLXBM	Link Extension Blocks and I/O Buffers for MSC MTM links	Global
DFSMFDDH	MFS Pool Dynamic Directory Hash Table	Local[4 on page 557]
DFSMFDDP	MFS Pool Dynamic Directory Prime Area	Local[4 on page 557]
DFSMFDD0	MFS Pool Dynamic Directory Entry Area	Local[4 on page 557]
DFSMFPDS	MFS Pool PDS Directory Indexes	Local[4 on page 557]
DFSMFSTG	MFS Pool Staging Buffers	Local[4 on page 557]
DFSMTCLB	CLB (ECB) for DFSCMTIO	Global
DFSMTIOT	Monitor TIOT Table	Global
DFSMTMH	MSC Main Storage-to-Main Storage Queue Header	Local[3 on page 557]
DFSMTMW	MSC Main Storage-to-Main Storage Window	Local[3 on page 557]
DFSOBFPL	OSAM Buffer Pool	Global[2 on page 557]

Table 194. Load List Areas (continued)

Load List Name	IMS Block/Work Area	Pool Type
DFSOBFWA	OSAM Buffer Pool Work Area	Local
DFSOLRnn	OLDS Read DCB where nn must be numeric	Local
DFSOSDEB	OS/VS2 “Fake” OSAM DEB	Global
DFSPCWAP	Communications Work Pool	Local
DFSPDBWP	Database Work Pool	Global
DFSPDMB	DMB Pool	Global
DFSPFBP	MFS Pool	Local
DFSPFWA	Prefetch Work Area, ECB and Save Sets	Local
DFSPPSBW	PSB and PSB Work Pool	Global
DFSPQBUF	Queue Manager Buffers	Local
DFSPSBRS	Resident PSBs	Global
DFSPSTQE	Scheduler Sequence Queue	Global
DFSPSTxx	SAP Work Area, xx=PST number	Global
DFSPTPDB	Communications Pool	Local
DFSPWKAP	Working Storage General Pool	Global[2 on page 557]
DFSRSTEB	Restart ECB and Save Sets	Local
DFSRSTWA	Restart Work Area	Local
DFSSBBUF	Sequential buffering: SBUF	Local
DFSSBCA1	Sequential buffering: SCAR	Global
DFSSBDCB	Sequential buffering: SDCB	Local
DFSSBDSE	Sequential buffering: EDSG	Local
DFSSBDSG	Sequential buffering: SDSG	Local
DFSSBITA	Sequential buffering: ITASK storage for overlapped I/O	Global
DFSSBPSS	Sequential buffering: SBPSS	Global
DFSSBPST	Sequential buffering: SBPST	Local
DFSSBRAN	Sequential buffering: SRAN	Local
DFSSBSBU	Sequential buffering buffers	Local
DFSSBSCD	Sequential buffering: SBSCD	Global
DFSSBWO	Sequential buffering: DFSSBWO	Local
DFSSLX	SCD Latch Extension	Global
DFSSSCT	Subsystem Control Table	Local[3 on page 557]
DFSSTAEB	STAE Work Area	Local
DFSSTPEB	Stop Region ECB, Save Sets and Work Area	Local
DFSSTPWA	Stop Region Message Work Area	Local
DFSTRMWK	Modify/Terminate Task Save Sets, ECB and Work Area	Local
DFSTSAV	Temporary Save Sets	Local
DFSVRFXL	Fixlist for EXCPVR	Local
DFSXCWxx	Exclusive Control Enqueue/Dequeue Work Area, xx=01-99	Global[2 on page 557]
DFSZIBxx	ZIB/FAQE Pool, xx=01-99	Global

Table 194. Load List Areas (continued)

Load List Name	IMS Block/Work Area	Pool Type

Notes:

1. A large number of storage pools are defined in module DFSCBT00. The contents directory element (CDE) name for storage in a given control block table (CBT) pool is #xxxxyyy, where xxxx is the pool name, and yyy is a number from 001 to 999. See "Control Block Table (CBT) Pools" for a description of the CBT pools.
2. When you use the local storage option (LSO), all these areas are obtained from local storage. When you use Fast Path and LSO, DFSLCD, DFSDBUFF, and DFSXCWxx remain in global storage. When you select LSO = S, DFSLCD and DFSPWKAP remain in global storage.
3. IMS constructs these areas at abend time. They consist of copies of the subject areas preceded by one word containing the original address of the area.
4. IMS builds these areas in extended private storage.

Control Block Table (CBT) Pools

Table 195. CBT Pool Names and Descriptions

CBT Pool	Description
AHDR	Autologon LU headers
ADSC	Fast Path DEDB area data set control block
AESL	Fast Path DBRC parameter area
AWE	Work-to-do element for task communication
BCPT	Checkpoint ID table
BQEL	Used when a buffer is altered and released at sync point
BXQE	Storage manager queue elements
CBLK	LU 6.2 CPI communications driven control block
CCB	Conversational control block
CLLE	Common latch list element
CMWU	Save sets/ECB for ITASKs which do not require a PST
CSAG	Callable services anchor block (ECSA storage)
CSAL	Callable services anchor block (E-private storage)
DBPB	Database purge block
DBRC	DBRC work area
DDIR	Database directories
DDRE	DMB directory extension
DESC	LU 6.2 descriptor block
DG2W	Dispatcher work area section 2 (global storage)
DL2W	Dispatcher work area section 2 (local storage)
DPST	Dependent region PST: The following blocks are associated with the dependent region structure: `D1WA, DG2W, EPST, FSRB, GQMW, IDT, IOSB, IRLM, KLSD, LCRE, SAP, SLOG, STTR, XPST.`
D1WA	Dispatcher work area section 1
EPST	Fast Path PST extension
EQEL	Recoverable in doubt structure queue elements
EZS	External subsystem storage

Table 195. CBT Pool Names and Descriptions (continued)

CBT Pool	Description
FEIB	Front-end message switch interface block
FNCB	Used by Fast Path for global command notifies
FPCP	Used by Fast Path for local commands
FSRB	Fast Path wake up/sleep SRBs
GESE	Represents a defined external subsystem
GIOB	IOB for batch
GOWA	OSAM channel programs for batch
GQMW	Global queue manager work area
GS24	Global 24-bit savearea
GSAV	Global save area
IAFP	IMS advanced future print block
IDT	Block used to keep track of identified regions
IEQE	In-flight/in-doubt data buffers
IOSB	I/O supervisor block for OSAM
IRLM	Dependent region block, if IRLM is used
KLSD	LSO=X,Y block for each dependent region
LCLL	Local common latch list element (E-private storage)
LCRE	Local Recovery element (persists across restart)
LG24	Below the 16MB line dynamic SAP save sets
LGND	Block used to hold logon descriptor representations
LGWA	Log work area
LGWX	Log work area extension (private)
LPST	PSTs for IMS internal use in local storage
LQB	Local queue block (SPQBs and CNTs)
LQMW	Local queue manager work area
LS24	Local 24-bit savearea
LSAV	Dynamic SAP save sets
LUB	LU 6.2 LU block
L56X	Fast Path database control log record
MSGP	Message buffers in global storage
OSWA	OSAM channel program areas
PCIB	MFS Partition CIB
PDIR	Program directories
PF62	LU 6.2 message prefix block
PST	PSTs for IMS internal use in global storage
QAB	LU 6.2 queue anchor block
QMBA	Queue manager global buffer area
QSAV	Save sets with AWEs
RACW	RACF workarea
RCNT	Remote communication name table

Table 195. CBT Pool Names and Descriptions (continued)

CBT Pool	Description
RCTE	Fast Path routing codes
RECA	VTAM receive any buffers
RPST	Restart PST
RRE	Represents an active thread to an external subsystem
SAP	Save area prefix – Includes fixed and dynamic SAPs
SIDX	One for each identified external subsystem
SLOG	IMS Monitor parameter area block
SMB	Scheduler message blocks
SOPB	Sign-on parameter list block
SRBC	Common SRBs used for data sharing asynchronous NOTIFYs
STAT	Database Control (DBCTL) and Database Resource Adapter (DRA) statistics area
STTR	Retrieve trace area
SVPG	System service parameter list block (global-ECSA)
SVPL	System service parameter list block (local-private)
TCBT	TCB table
TIB	LU 6.2 transaction instance block
TTAB	Trace table (31-bit storage)
TT24	Trace table (24-bit storage)
USMU	Security block
USRD	Blocks used to represent user control block structure
VRPL	VSAM RPL with two save areas
VTCB	VTAM terminal control blocks
VWA	Volatile work area
XMCI	Cross memory ITASK block
XPST	Dependent region PST extension
X124	DLI pool below the 16MB line for MVS/ESA

Appendix H. Acronyms and Abbreviations Used in This Book

ACB	access method control block
AIB	application interface block
AMP	access method prefix block
APAR	authorized program analysis report
BMP	batch message processing
BSAM	Basic Sequential Access Method
CBT	control block table
CCB	conversational control block
CCTL	coordinator controller
CDE	contents directory element
CIB	communication interface block
CICS	Customer Information Control System
CLB	communication line block
CNT	communication name table
CQS	Common Queue Server
CRB	communication restart block
CTB	communication terminal block
CTRL	IMS control region
CTT	communication translate table
DBCTL	Database Control
DBRC	Database Recovery Control
DB	Database function
DCB	data control block
DC	data communication function
DDIR	data management block directory
DEDB	Data entry database
DMAC	data management area control block
DMB	data management block
DMCB	data management control block
DRA	Database Resource Adapter
DSG	data set group
DSP	IMS dispatcher
DSPWRK	IMS dispatcher work area
ECB	event control block
ECNT	extended communication name table
EEVT	external entry vector table

EEVTP	external entry vector table prefix
EPST	extended partition specification table
ES	extended security support
ESCD	extended system contents directory
ESETP	external subsystem entry table prefix
ESS	external subsystem
EWS	Early Warning System
EZS	external connection status element
FP	Fast Path
FTSC	Field Technical Support Center
GESE	global external subsystem entry
ID	identification
ILS	isolated log send
IMS	Information Management System
I/O	input/output
IOB	input/output block
IRLM	Internal Resource Lock Manager
ISC	Intersystem Communication
ISI	resource access security
ISL	IRLM identified subsystem list
ITASK	IMS task
IWALE	internal work area list elements
IXRF	IMS-related XRF complex
LCB	link control block
LCRE	local current recovery entry
LESE	local external subsystem entry
LLB	logical link block
LNB	logical link name block
LTERM	logical terminal
MFS	Message Format Service
MNOTE	macro note
MPP	message processing program
MRMB	randomizing module block
MRQ	Message Requeuer
MSC	Multiple Systems Coupling
MSDB	main storage database
MVS	Multiple Virtual System

NM	notify message
OSAM	Overflow Sequential Access Method
PCB	program communication block
PCIB	Partition CIB
PDIR	PSB directory
PSB	program specification block
PST	partition specification table
PSW	program status word
PTERM	physical terminal
PTF	program temporary fix
QCB	queue control block
QSAM	Queued Sequential Access Method
RCTE	routing code table entry
RHB	IRLM resource header block
RLB	IRLM request lock block
RLMCB	IRLM master control block
RPL	request parameter list
RRE	residual recovery element
RSR	Remote Site Recovery
SAP	save area prefix
SB	sequential buffering
SCD	system contents directory
SCP	system control program
SE	system engineer
SMB	scheduler message block
SMP	System Modification Program
SPA	scratch pad area
SQ	shared queues
SSCD	secondary system contents directory
SSF	Software Support Facility
SSIE	subsystem status index block
SSQ	schedule sequence queue
SUR	Database Surveyor utility feature
SYS	systems
TCB	task control block
TCT	transaction class table
TKO	takeover

TPPCB	telecommunication program PCB
TRK	tracking
UTIL	utility
VTCB	VTAM terminal control block
XRF	Extended Recovery Facility

Notices

This information was developed for products and services offered in the U.S.A. IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
J46A/G4
555 Bailey Avenue
San Jose, CA 95141-1003
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurement may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information is for planning purposes only. The information herein is subject to change before the products described become available.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Programming Interface Information

This book is intended to help system programmers diagnose IMS problems. This book documents information that is Diagnosis, Modification or Tuning Information provided by IMS.

Attention: Do not use this Diagnosis, Modification or Tuning Information as a programming interface.

Trademarks

The following terms are trademarks of the IBM Corporation in the United States or other countries or both:

BookManager	MVS/ESA
DATABASE 2	OS/390®
DB2	RACF
CICS	RETAIN
IBM	SAA
IMS	System/390
IMS/ESA	VTAM
MVS	

Other company, product, and service names, which may be denoted by a double asterisk (**), may be trademarks or service marks of others.

Product Names

In this book, the licensed program “DB2 for OS/390” is referred to as “DB2”.

Bibliography

This bibliography includes all the publications cited in this book, including the publications in the IMS library.

- *Authorized Assembler Programming Guide*, GC28-1645
- *IMS Message Requeuer Program Description/Operations Manual*, SH21-1089
- *MVS Data Areas*, LY28-1857 though LY28-1861
- *MVS/ESA Diagnosis: Procedures*, LY28-1844
- *MVS/ESA Diagnosis: Tools and Service Aids*, LY28-1845
- *MVS/ESA Planning: Operations*, GC28-1441
- *MVS/ESA System Commands*, GC28-1442
- *MVS/ESA System Codes*, GC28-1486
- *OS/390 MVS IPCS User's Guide*, GC28-1756
- *System Messages, Volume 1*, GC28-1480
- *System Messages, Volume 2*, GC28-1481
- *System Messages, Volume 3*, GC28-1482
- *System Messages, Volume 4*, GC28-1483
- *System Messages, Volume 5*, GC28-1484
- *System/390 Reference Summary*, GX20-1850
- *VTAM Messages and Codes*, SC31-6564

IMS Version 7 Library

SC26-9419	ADB	*IMS Version 7 Administration Guide: Database Manager*
SC26-9420	AS	*IMS Version 7 Administration Guide: System*
SC26-9421	ATM	*IMS Version 7 Administration Guide: Transaction Manager*
SC26-9422	APDB	*IMS Version 7 Application Programming: Database Manager*
SC26-9423	APDG	*IMS Version 7 Application Programming: Design Guide*
SC26-9424	APCICS	*IMS Version 7 Application Programming: EXEC DLI Commands for CICS and IMS*
SC26-9425	APTM	*IMS Version 7 Application Programming: Transaction Manager*
SC26-9427	CG	*IMS Version 7 Customization Guide*
SC26-9426	CQS	*IMS Version 7 Common Queue Server and Base Primitive Environment Guide and Reference*
SC26-9436	CR	*IMS Version 7 Command Reference*
SC26-9428	DBRC	*IMS Version 7 Database Recovery Control (DBRC) Guide and Reference*
LY37-3738	DGR	*IMS Version 7 Diagnosis Guide and Reference*
LY37-3739	FAST	*IMS Version 7 Failure Analysis Structure Tables (FAST) for Dump Analysis*
SC27-0832	JGR	*IMS Version 7 IMS Java Guide and Reference*
GC26-9429	IIV	*IMS Version 7 Installation Volume 1: Installation and Verification*
GC26-9430	ISDT	*IMS Version 7 Installation Volume 2: System Definition and Tailoring*
SC26-9432	MIG	*IMS Version 7 Master Index and Glossary*
GC26-9433	MC1	*IMS Version 7 Messages and Codes, Volume 1*
GC26-1120	MC2	*IMS Version 7 Messages and Codes, Volume 2*
SC26-9434	OTMA	*IMS Version 7 Open Transaction Manager Access Guide*
SC26-9435	OG	*IMS Version 7 Operations Guide*
GC26-9437	RPG	*IMS Version 7 Release Planning Guide*
SC26-9438	SOP	*IMS Version 7 Sample Operating Procedures*
SC26-9440	URDBTM	*IMS Version 7 Utilities Reference: Database and Transaction Manager*
SC26-9441	URS	*IMS Version 7 Utilities Reference: System*

Supplementary Publications

GC26-9431	LPS	*IMS Version 7 Licensed Program Specifications*
SC26-9439	SOC	*IMS Version 7 Summary of Operator Commands*

Publication Collections

LK3T-3526	CD	IMS Version 7 Licensed Product Kit: Online Library
LK3T-7144	CD	IMS Favorites

		Publication Collections
LBOF5294	Hardcopy and CD	Licensed Bill of Forms (LBOF): IMS Version 7 Hardcopy and Online Library
SBOF5297	Hardcopy	Unlicensed Bill of Forms (SBOF): IMS Version 7 Unlicensed Hardcopy Library
SK2T-0730	CD	IBM Online Library: Transaction Processing and Data
SK2T-6700	CD	IBM Online Library OS/390

Index

Special characters

Numerics

A

B

C

D

J

K

L

T

U

V

W

X

Z

Program Number: 5655-B01

LY37-3738-05

Spine information:

IBM IMS Diagnosis Guide and Reference Version 7