

IMS



# Diagnosis Guide and Reference

*Version 9*



IMS



# Diagnosis Guide and Reference

*Version 9*

**Note**

Before using this information and the product it supports, be sure to read the general information under “Notices” on page 613.

**First Edition (October 2004)**

This edition applies to Version 9 of IMS (product number 5655-J38) and to all subsequent releases and modifications until otherwise indicated in new editions.

This is a licensed document that contains restricted materials of International Business Machines Corporation.

© **Copyright International Business Machines Corporation 1974, 2004. All rights reserved.**

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

# Contents

<b>Figures</b> . . . . .	vii
<b>Tables</b> . . . . .	xi
<b>About This Book</b> . . . . .	xvii
Summary of Contents . . . . .	xvii
Prerequisite Knowledge. . . . .	xviii
Product Names. . . . .	xviii
How to Send Your Comments. . . . .	xix
<b>Summary of Changes</b> . . . . .	xxi
Changes to This Book for IMS Version 9. . . . .	xxi
Library Changes for IMS Version 9. . . . .	xxii
<hr/>	
<b>Part 1. Identifying System Problems</b> . . . . .	1
<b>Chapter 1. Setting Up Your System</b> . . . . .	3
Setup Recommendations for z/OS. . . . .	3
Setup Recommendations for IMS . . . . .	5
CQS Trace Setup Recommendations. . . . .	7
Installing the IMS Dump Formatter . . . . .	8
Setting Up the External Trace Environment . . . . .	8
Setting Up CQS, OM, RM, and SCI Tracing. . . . .	10
Management of Standard Documentation. . . . .	11
Manual Intervention for Dump Creation . . . . .	12
<b>Chapter 2. Collecting Data about Problems</b> . . . . .	17
Collecting Data about General Problems . . . . .	17
Collecting Data about Specific Problems . . . . .	18
<b>Chapter 3. Searching Problem Reporting Databases</b> . . . . .	29
Developing Search Arguments. . . . .	29
Creating a Search Argument . . . . .	30
<b>Chapter 4. Selecting the Keywords</b> . . . . .	31
Component Identification Keyword Procedure . . . . .	31
Type-of-Failure Keyword . . . . .	31
<b>Chapter 5. Procedures and Techniques</b> . . . . .	59
Searching the Database . . . . .	59
Searching for APARs Closed within a Specific Time Period . . . . .	60
Preparing an APAR. . . . .	61
<hr/>	
<b>Part 2. Data Areas and Record Formats</b> . . . . .	63
<b>Chapter 6. Data Areas and Record Formats</b> . . . . .	65
Getting More Information on Modules, Control Blocks, and Record Formats . . . . .	65
Table of Control Block Definitions . . . . .	67
Control Block Interrelationship Diagrams . . . . .	74
DL/I Record Formats. . . . .	113
<hr/>	
<b>Part 3. Diagnostic Aids</b> . . . . .	121

<b>Chapter 7. SYS—System Service Aids</b>	127
Log Records	127
File Select and Formatting Print Utility	152
I Log Merge Utility	154
Formatting IMS Dumps Offline	154
Edited Command Buffer Format	177
Interactive Dump Formatter	178
Formatting IMS Dumps Online	183
SNAP Call Facility	189
I /DIAGNOSE Command SNAP Function	190
Common Trace Table Interface	190
<b>Chapter 8. DB—Database Service Aids</b>	253
The Job Control Block (JCB) Trace	253
Data Language/I Test Program—DFSDDLTO	255
COMPARE Statement SNAPs	255
SNAPs on Exceptional Conditions	256
DL/I Call Image Capture	257
DL/I Analysis	258
Locating Database-Related Traces	262
DL/I Trace	263
Retrieve Trace	301
I Online Recovery Manager Trace	306
Program Isolation-Related Problem Analysis	311
Log Analysis (Database Related)	312
Sequential Buffering Service Aids	316
GSAM Control Block Dump—DFSZD510	319
<b>Chapter 9. DC—Data Communication Service Aids</b>	325
Terminal Communication Task Trace	325
DC Trace	327
Diagnosing Problems in the Queue Control Facility/Message Requeuer	341
Diagnosing Message Routing Problems	348
I IMS Transaction Trace	358
Receive-Any Buffer Analysis	361
Finding the Active Save Set	363
IMS-VTAM Interface	363
IBM 3270 Error Recovery Analysis	363
Message Format Service Normal BTAM Path	364
Diagnosing Message Format Service Problems	369
Message Format Service Module Traces	371
I Tracing Errors in Module DFSCNXA0	372
IDC0 Trace Table Entries	379
APPC/IMS Diagnostic Aids	382
OTMA Diagnostic Aids	400
Diagnosing Errors Diagnosing Fast Path Problemsrelated to Print Data Set Options: IMS Spool API Support	405
<b>Chapter 10. IRLM Service Aids</b>	411
I IRLM Lock Trace Analysis Using KBLA	411
IRLM Dumps	412
SYS1.LOGREC	412
z/OS Component Trace	412
<b>Chapter 11. FP—Fast Path Service Aids</b>	415
Diagnosing Fast Path Problems	415

DEDB Control Interval (CI) Problem Assistance Aids . . . . .	419
Locating Fast Path Control Blocks and Tables . . . . .	422
Fast Path External Trace . . . . .	424
<b>Chapter 12. MSC—Multiple Systems Coupling Service Aids . . . . .</b>	<b>433</b>
Multiple Systems Coupling Communication Task Trace . . . . .	433
Multiple Systems Coupling Device-Dependent Module . . . . .	433
Multiple Systems Coupling Traces . . . . .	435
Diagnosing Link Problems . . . . .	435
MSS1 and MSS2 Records. . . . .	440
Channel-to-Channel Access Method Trace Stack (LXB Trace). . . . .	441
<b>Chapter 13. DBRC—Database Recovery Control Service Aids . . . . .</b>	<b>447</b>
Diagnosing from a RECON List . . . . .	447
RECON Record Types . . . . .	447
DBRC Internal Trace . . . . .	450
DBRC Internal Unformatted Trace Example . . . . .	462
DBRC External Trace . . . . .	469
DBRC API Return and Reason Codes . . . . .	472
<b>Chapter 14. DRA—Database Resource Adapter Service Aids . . . . .</b>	<b>477</b>
DRA Dumps . . . . .	477
Analyzing DRA Problems . . . . .	478
<b>Chapter 15. RSR—Remote Site Recovery Service Aids . . . . .</b>	<b>481</b>
Determining Last Non-MSM Message Recorded . . . . .	481
Determining Last MSM Message Recorded . . . . .	483
Fast Path Tracker Trace Entries. . . . .	483
X'D4': Database Tracker Trace Entries (D4) . . . . .	489
Buffer Handler Trace Entries at Database Tracker . . . . .	492
Log Router Trace Data . . . . .	492
X'4930': Database Tracker FSE Error Log Record Format . . . . .	519
<b>Chapter 16. CQS Diagnosis. . . . .</b>	<b>521</b>
Diagnosing a CQS Related Problem . . . . .	521
CQS Structure Rebuild Problems . . . . .	525
CQS Trace records . . . . .	526
CQS Log Records. . . . .	529
Printing CQS Log Records . . . . .	531
Copying CQS Log Records for Diagnostics . . . . .	532
<b>Chapter 17. CSL Diagnosis . . . . .</b>	<b>535</b>
CSL Trace Records . . . . .	535
RM Trace Record Example . . . . .	537
<hr/>	
<b>Part 4. Appendixes . . . . .</b>	<b>539</b>
<b>Appendix A. IMS Keyword Dictionary . . . . .</b>	<b>541</b>
<b>Appendix B. Dependency Keywords . . . . .</b>	<b>545</b>
<b>Appendix C. AIBREASN Codes for Queue Control Facility (QCF)/Message Requeuer (MRQ)</b>	
<b>Errors . . . . .</b>	<b>547</b>
AIBREASN Codes Set by DFSQMRQ0 . . . . .	547
AIB Return Codes Set by DFSQMRQ0 . . . . .	553

**Appendix D. Locating IMS Blocks and Work Areas Using Load List Elements . . . . . 581**  
Load List Areas. . . . . 581  
Control Block Table (CBT) Pools . . . . . 583

**Appendix E. Acronyms and Abbreviations Used in This Section . . . . . 587**

**Appendix F. Fast Path Trace Entries . . . . . 591**

**Notices . . . . . 613**  
Programming Interface Information . . . . . 614  
Trademarks . . . . . 615

**Bibliography . . . . . 617**  
IMS Version 9 Library . . . . . 617  
Supplementary Publications . . . . . 618  
Publication Collections . . . . . 618  
Accessibility Titles Cited in This Library . . . . . 618

**Index . . . . . 619**



## Figures

1. IMS Control Block Linkage for a Static DB/DC Environment . . . . .	66
2. Online System Contents Directory (SCD) . . . . .	75
3. DFSPRPX0–Parameter Blocks. . . . .	81
4. DL/I OSAM Buffer Pool . . . . .	82
5. Sequential Buffering Control Blocks . . . . .	83
6. Buffer Handler Pool (VSAM). . . . .	84
7. OSAM DECB with IOB in Use . . . . .	85
8. OSAM IOB Pool Showing Available IOBs . . . . .	86
9. Storage Management Control Block Relationships Created for the MAIN Pool . . . . .	87
10. Storage Management Control Block Relationships for Preallocated Storage Blocks . . . . .	88
11. Storage Management Control Block Relationships (DFSPPOOL Pools) . . . . .	90
12. Storage Management Control Block Relationships (DFSCBT00 Pools) . . . . .	91
13. Database Manager Control Blocks for a Representative Database. . . . .	92
14. Database Control Blocks . . . . .	94
15. Diagram of a Data Management Block (DMB) . . . . .	96
16. Overview of Fast Path Control Blocks . . . . .	97
17. Relationships Between Buffer Control Blocks for Fast Path Databases . . . . .	98
18. GSAM Control Block Overview. . . . .	99
19. GSAM Control Blocks . . . . .	100
20. DL/I Control Block Relationships. . . . .	101
21. IMS Transaction Manager Control Blocks . . . . .	102
22. Intersystem Communication Control Block Structure . . . . .	102
23. VTCB Load Module . . . . .	103
24. Multiple Systems Coupling (MSC) Control Block Overview . . . . .	105
25. Multiple Systems Coupling (MSC) Main Storage-to-Main Storage Control Block Overview . . . . .	106
26. z/OS Storage Map Showing IMS-to-IRLM Interrelationships. . . . .	107
27. IRLM Overall Control Block Structure . . . . .	108
28. IRLM Storage Manager Pools . . . . .	109
29. IRLM Lock Request Examples . . . . .	109
30. Control Block Overview of Database Recovery Control (DBRC) . . . . .	110
31. Organization and Basic Linkages: DOF (Device Output Format) and MOD (Message Output Descriptor). . . . .	111
32. Organization and Basic Linkages: DIF (Device Input Format) and MID (Message Input Descriptor). . . . .	112
33. HSAM and SHSAM Segment Format . . . . .	113
34. Delete Byte (Flag) Format . . . . .	113
35. Block Format for HSAM and SHSAM . . . . .	114
36. HISAM Segment Format . . . . .	114
37. SHISAM Segment Format . . . . .	114
38. LRECL Format . . . . .	115
39. VSAM Block Formats . . . . .	115
40. HDAM, HIDAM, PHDAM, or PHIDAM Segment Format . . . . .	115
41. Mapping the Prefix of a Segment . . . . .	116
42. OSAM and VSAM ESDS Block Format . . . . .	117
43. LRECL Format On Storage Device and in Buffer Pool . . . . .	118
44. VSAM LRECL Format As Returned by Buffer Handler . . . . .	118
45. VSAM Block Format on Device and in Buffer Pool . . . . .	118
46. LRECL Format on Device and in Buffer Pool . . . . .	118
47. LRECL as Returned by Buffer Handler . . . . .	119
48. VSAM Block Format on Device and in Buffer Pool . . . . .	119
49. Segment Data Format . . . . .	119
50. HISAM, HDAM, HIDAM, PHDAM, and PHIDAM Segment Format . . . . .	120
51. HDAM, HIDAM, PHDAM, and PHIDAM . . . . .	120

52. Log Record Layout . . . . .	150
53. Log Record Prefix Area Layout . . . . .	150
54. Log Subrecord and Data Area Layout. . . . .	151
55. Log Sequence Field Layout . . . . .	151
56. Unformatted Output Using DFSERA10 . . . . .	153
57. Formatted Output Using DFSERA10 with Option Statement, Exit=DFSERA30 . . . . .	153
58. Edited Command Layout . . . . .	177
59. IMS Dump Formatting Primary Menu Panel . . . . .	179
60. IMS Dump Formatting Initialization/Content Panel - Inactive . . . . .	179
61. IMS Dump Formatting Initialization/Content Panel - Active . . . . .	180
62. IMS High-Level Dump Formatting Panel . . . . .	181
63. IMS Low-Level Dump Formatting Selection Panel . . . . .	181
64. IMS Analysis Selection Panel . . . . .	182
65. IMS Enhanced Dump Formatting Menu . . . . .	183
66. Sample Filtering Panel . . . . .	183
67. Sample Filtering Criteria. . . . .	183
68. How to Locate Trace Tables . . . . .	193
69. General Trace Record Format . . . . .	194
70. Example of a Dispatcher Trace . . . . .	211
71. External Subsystem (ESS) Trace Record Format . . . . .	212
72. Example of an External Subsystem Trace (SST). . . . .	225
I 73. Scheduler Trace Record Format for Function Code X'41'. . . . .	238
I 74. Scheduler Trace Record Format for Function Code X'42'. . . . .	238
I 75. Scheduler Trace Record Format for Function Code X'43'. . . . .	239
I 76. Scheduler Trace Record Format for Function Code X'44'. . . . .	239
I 77. Scheduler Trace Record Format for Function Code X'45'. . . . .	239
I 78. Scheduler Trace Record Format for Function Code X'46'. . . . .	239
I 79. Scheduler Trace Record Format for Function Code X'47'. . . . .	240
I 80. Scheduler Trace Record Format for Function Code X'48'. . . . .	240
I 81. Example of a Scheduler Trace . . . . .	241
82. Example of a Latch Trace . . . . .	246
83. Low Level Trace Record Format. . . . .	248
84. Medium Level Trace Record Format - X'21' . . . . .	248
85. Medium Level Trace Record Format - X'20' . . . . .	248
86. Medium Level Trace Record Format - X'22' . . . . .	249
87. Low Level Trace Record Format - X'08', X'15', X'1B' . . . . .	249
88. Low Level Trace Record Format - X'0A' . . . . .	250
89. Low Level Trace Record Format - X'16' . . . . .	250
90. Example of a Job Control Block (JCB) Dump . . . . .	254
91. How to Locate the Database Traces . . . . .	263
92. X'0C' Trace Entry . . . . .	266
93. X'31', X'32', X'34', X'B1', and X'B2' Trace Entries . . . . .	267
94. X'60' and X'61' Trace Entries . . . . .	268
95. X'62' Trace Entry . . . . .	268
I 96. X'69' Trace Entry . . . . .	269
I 97. X'6A Trace Entry . . . . .	270
I 98. X'6B Trace Entry . . . . .	270
I 99. X'6C' Trace Entry . . . . .	271
I 100. X'6F' Trace Entry . . . . .	272
101. X'80', X'81', X'82' Trace Entry . . . . .	272
102. X'AA' Trace Entry . . . . .	273
I 103. X'AB' Trace Entry . . . . .	273
104. X'AC' Trace Entry . . . . .	275
105. X'C4' Trace Entry . . . . .	276
I 106. X'C6' Trace Entry . . . . .	276
107. X'C7' Trace Entry (When Not Using the IRLM) . . . . .	277

108. X'C7' Trace Entry (When Using the IRLM)	277
109. X'C8' Trace Entry	278
110. X'C9' Trace Entry	279
111. X'CA' Trace Entry	280
112. X'CA'—X'08' Trace Entry	281
113. X'CA' Trace Entry for Fast Path Calls	281
114. X'CB' Trace Entry	282
115. X'CC' Trace Entry	282
116. X'CF' Trace Entry	285
117. X'D0' Trace Entry	286
118. X'D1' Trace Entry	287
119. X'D9' Trace Entry - Words 0 through 2	288
120. X'D9' Trace Entry - Words Specific to the Validation or Creation of the OLR Output Data Set and the Deletion of the Inactive Data Set	288
121. X'9D' Trace Entry - Words Specific to the Fence Value Before an OLR IPOST/IWAIT	291
122. X'9D' Trace Entry - Next UOR Determination	293
123. X'9D' Trace Entry - Words Specific to OLR Command Processing	293
124. X'9D' Trace Entry - Words Specific to OLR Start	294
125. X'9D' Trace Entry - Words Specific to Start of UOR	294
126. X'9D' Trace Entry - Words Specific to UOR Wait for Timer	294
127. X'DA' Trace Entry	295
128. X'DB' through X'FA' Trace Entries	296
129. Example of a DL/I Trace	300
130. Example of a Retrieve Trace	305
131. General Areas of Database (DB) Analysis	313
132. Formatted GSAM Control Block Dump	320
133. Unformatted GSAM Control Block Dump	321
134. Example of a Terminal Communication Task Trace Entry	327
135. Data Communication (DC) Trace Records	340
136. Query Control Facility Interface to IMS	342
137. QCF Prefix Mapped by DFSMRQPF	344
138. Sample JCL for Printing SCRAPLOG Records	345
139. Sample JCL for Printing 6701-MRQE Records	346
140. Sample Log Record Showing Successfully Requeued Message	348
141. IMS Transaction Trace Records	360
142. IPCS Dump Formatter EDA/TM Option	362
143. Example of Save Area Set	363
144. Message Format Service (MFS) Normal BTAM Path	365
145. Example of CIBSTRAC Trace	371
146. Example of CIBTRACE Trace	372
147. Example of an LU Manager Trace	389
148. IRLM Lock Trace Analysis	411
149. Example of Message DFS2712I	416
150. Example of a Save Area Set	417
151. EQE list in CI 1	420
152. RAP CI	420
153. First DOVF CI	420
154. Other DOVF CIs	420
155. First IOVF CI	421
156. Other IOVF CIs	421
157. SDEP CI	421
158. Printout of the LXB Trace Stack	444
159. DBRC Trace Header Record	453
160. DBRC Trace Processing Flow	454
161. One-Line Trace Entry Produced When Module A Calls Module B	454
162. One-Line Trace Entry Produced When Module B Returns to Module A	455

163. DSPSTACK Trace Entry . . . . .	455
164. BGNCABN0 Trace Entry . . . . .	456
165. DSPCABN0 Trace Entry . . . . .	456
166. BGNRETRY Trace Entry . . . . .	456
167. DSPCRTR0 Trace Entry . . . . .	457
168. CRTR0XIT Trace Entry . . . . .	457
169. DSPURI00 Entry Trace Entry . . . . .	459
170. GETFEED Trace Entry for One RECON . . . . .	460
171. DSPURI00 Exit Trace Entry . . . . .	462
I 172. Example of Internal Trace Table Entries . . . . .	463
173. Format of Trace Records . . . . .	469
174. DBRC External Trace Output for DBRC Router Processing . . . . .	470
175. DBRC External Trace Output for RECON I/O Error Processing . . . . .	471

## Tables

1. Licensed Program Full Names and Short Names . . . . .	xviii
2. IMS Component Identification Numbers . . . . .	31
3. Key Fields in SAP Analysis . . . . .	43
4. Key Data from an Abnormal Save Area Set . . . . .	45
5. Preparing an APAR . . . . .	61
6. Table of Control Block Definitions . . . . .	67
7. Description of Control Block Interrelationship Diagrams . . . . .	74
8. IMS Log Records Used to Analyze IMS Problems . . . . .	127
9. X'2900' Log Record Layout . . . . .	142
10. X'2910' Log Record Layout . . . . .	142
11. X'2920' Log Record Layout . . . . .	143
12. X'2930' Log Record Layout . . . . .	144
13. X'2930' Log Record Layout — Repeated Data Set Fields . . . . .	144
14. X'2940' Log Record Layout . . . . .	146
15. X'2950' Log Record Layout . . . . .	147
16. X'2970' Log Record Layout . . . . .	148
17. X'2990' Log Record Layout . . . . .	149
18. Log Record Prefix Area Format for X'67' . . . . .	150
19. Log Record Prefix Area Format for X'67FA' Records . . . . .	151
20. Log Subrecord Area Format . . . . .	151
21. Log Data Area Format . . . . .	151
22. Log Sequence Field Format . . . . .	152
23. FMTIMS Parameters for General Problems . . . . .	156
24. FMTIMS Parameters Based on CALLER= and TCB= Fields . . . . .	157
25. Formatted Areas Under the FMTIMS Options DB and DB,MIN . . . . .	162
26. Data Communication Areas Formatted by DC and DC,MIN . . . . .	163
27. DEDB Control Block Areas Formatted by DEDB and DEDB,MIN . . . . .	163
28. Areas Formatted by DISPATCH and DISPATCH,MIN . . . . .	164
29. Areas Formatted by EMH and EMH,MIN . . . . .	164
30. Areas Formatted by LOG and LOG,MIN . . . . .	164
31. Main Storage Databases Formatted by MSDB and MSDB,MIN . . . . .	165
32. Areas Formatted by QM and QM,MIN . . . . .	165
33. Sections Formatted by SB and SB,MIN . . . . .	167
34. Areas Formatted by SCD and SCD,MIN . . . . .	168
35. Trace Tables in the Common Trace Interface . . . . .	190
36. Trace Function Codes . . . . .	194
37. Common Service Layer Trace Function and Subfunction Codes . . . . .	198
38. Dispatcher Trace Record Format . . . . .	203
39. System Post Codes . . . . .	212
40. Module ID and Subfunction Table . . . . .	213
41. Resource Recovery Service Calls Associated with the Subfunction Codes . . . . .	226
42. Resource Recovery Services Function Routines Associated with DFSRRSI Function Routine Codes . . . . .	227
43. TRACE ID = X'5F03' (Get Trace Record) . . . . .	242
44. TRACE ID = X'5F04' (Get Trace Record) . . . . .	242
45. TRACE ID = X'5F05' (Release Trace Record) . . . . .	242
46. Format of a Latch Trace Entry . . . . .	242
47. Format of the Fast Path X'9C' Trace Entry . . . . .	251
48. Format of the Fast Path X'9D' Trace Entry . . . . .	252
49. DL/I User Call Encoded Functions . . . . .	254
50. Data Set Information . . . . .	274
51. Caller Information . . . . .	274
52. VSAM Request Option . . . . .	274

	53. VSAM Request Option . . . . .	275
	54. Special Lock or Unlock Call . . . . .	277
	55. PSTLRPRM Chart (Bytes 0 through 3) . . . . .	283
	56. I/O Toleration Function Code . . . . .	285
	57. I/O Toleration Flag 1 . . . . .	285
	58. I/O Toleration Flag 2 . . . . .	286
	59. Module and Subcode ID for X'D9': Validation or Creation of the OLR Output Data Set and the Deletion of the Inactive Data Set . . . . .	289
	60. Module and Subcode ID for X'D9': Fence Value Before an OLR IPOST/WAIT . . . . .	291
	61. Module and Subcode ID for X'D9': OLR Command Processing . . . . .	293
	62. JRNAD and UPAD Codes for X'DA' Trace Entry . . . . .	295
	63. Buffer Handler Function Codes Chart . . . . .	298
	64. Buffer Handler Return Codes Chart . . . . .	298
	65. Space Management and Buffer Handler Module Trace IDs . . . . .	299
	66. The Subroutines of the Retrieve Module (DFSDLR00). . . . .	302
	67. Trace Record 3702 - Create Data Set Routine Invoke DYA . . . . .	306
	68. Trace Record 3702 - Create Data Set Routine Invoke DYA . . . . .	307
	69. Trace Record 3702 - Create Data Set Routine Invoke DYA . . . . .	308
	70. Trace Record 3702 - Create Data Set Routine Invoke DYA . . . . .	308
	71. Trace Record 3702 - Create Data Set Routine Invoke DYA . . . . .	309
	72. Database Change Log Record DSECT . . . . .	313
	73. Example Processing Flow for a Terminal Communication Task Trace Entry . . . . .	327
	74. DC Trace Records . . . . .	331
	75. Map of Formatted CRTU Log Record . . . . .	336
	76. VTCB Posting in DFSVTPO0 . . . . .	337
	77. Key Fields in DFSMRQPF . . . . .	344
	78. Key Fields in Message (offset 0140 = offset 00 into message). . . . .	344
	79. Control Blocks and Data Areas Logged at Time of Error for 6701-MRQE Records . . . . .	346
	80. DFS070 Module Identifier Table . . . . .	349
	81. DFS070 Reason (RSN) Codes Table . . . . .	350
	82. DFS081 Module Identifier Table . . . . .	355
	83. DFS081 Reason (RSN) Codes Table . . . . .	356
	84. Location Codes for DFSCNXA0 Error Messages. . . . .	373
	85. Codes Related to ISC Processing . . . . .	374
	86. Codes Related to ISC BINDRACE Processing . . . . .	375
	87. Codes Related to MSC Errors . . . . .	376
	88. Codes Related to MSC SCIP Errors . . . . .	376
	89. Codes Related to Dynamic Logon Errors . . . . .	376
	90. Codes Related to Existing ISC Session Errors . . . . .	377
	91. Codes Related to User-Logon-Exit Routine Processing . . . . .	377
	92. Codes Related to Logon Errors . . . . .	377
	93. Codes Related to Logon Descriptor Processing . . . . .	378
	94. Codes Related to Logging-on Device Characteristics . . . . .	378
	95. Qualifier Codes Related to ETO Parsing Errors . . . . .	379
	96. Qualifier Codes Related to VTCB-Creation Errors . . . . .	379
	97. Qualifier Codes Related to Screen-Attribute Errors . . . . .	379
	98. Codes that Identify Error Messages Issued by DFSCNXA0 . . . . .	379
	99. LU Manager Trace Record Format . . . . .	382
	100. LU 6.2 Module-to-Code Cross-Reference Table . . . . .	390
	101. APPC/MVS Verb-to-Code Cross-Reference Table . . . . .	391
	102. OTMA Trace Record Format . . . . .	400
	103. OTMA Module-to-Code Cross-Reference Table . . . . .	403
	104. z/OS XCF Verb-to-Code Cross-Reference Table . . . . .	404
	105. Locating Information About the Offline Dump Formatter (ODF) . . . . .	415
	106. Fast Path Control Blocks and Work Areas that Appear in IMS Dumps . . . . .	423
	107. Control Block Structure of DBFCNT0 . . . . .	423

108. Multiple Systems Coupling Communication Task Trace . . . . .	434
109. Significant Fields in MSS1 and MSS2 Records . . . . .	440
110. RECON Record Types . . . . .	448
111. Calls to the Trace Routine in DSPURI00. . . . .	457
112. Translated RPLREQ Printable Codes . . . . .	461
113. Return and Reason Codes for TYPE=BACKOUT Query Requests . . . . .	473
114. Return and Reason Codes for TYPE=DB Query Requests . . . . .	473
115. Return and Reason Codes for TYPE=xxxxGROUP Query Requests . . . . .	474
116. Return and Reason Codes for TYPE=LOG Query Requests . . . . .	474
117. Return and Reason Codes for TYPE=OLDS Query Requests . . . . .	474
118. Return and Reason Codes for TYPE=RECON Query Requests . . . . .	475
119. Return and Reason Codes for TYPE=SUBSYS Query Requests . . . . .	475
120. Determining the Type of Dump the DRA Created . . . . .	477
121. Recovery Token Format . . . . .	478
122. Trace Record 9E01 - DBFDT210 Redo Record Processor Module Entry . . . . .	483
123. Trace Record 9F22 - DBFDT300 Fast Path/Fast Path TCB AWE Queue Server Module Entry . . . . .	484
124. Trace Record 9F22 - DBFDT300 Fast Path/Fast Path TCB AWE Queue Server Module Entry . . . . .	484
125. Trace Record 9F41 - DBFDT180 Area Status Change Module Entry . . . . .	484
126. Trace Record 9F44 - DBFROFR0 OFR Module Entry . . . . .	485
127. Trace Record 9F44 - DBFROFR0 OFR Module Entry . . . . .	485
128. Trace Record 9F44 - DBFROFR0 OFR Module Entry . . . . .	485
129. Trace Record 9F50 - DBFDT350 IPOST. . . . .	485
130. Trace Record 9F51 - DBFDT350 IWAIT . . . . .	486
131. Trace Record 9F52 - DBFDT350 GETEMAC . . . . .	486
132. Trace Record 9F53 - DBFDT350 GETERQE . . . . .	487
133. Trace Record 9F54 - DBFDT350 EMAC2 . . . . .	487
134. Trace Record 9F55 - DBFDT350 ERQE2 . . . . .	487
135. Trace Record 9F70 - DBFDT400 IPOST. . . . .	488
136. Trace Record 9F71 - DBFDT400 IWAIT . . . . .	488
137. Trace Record 9F72 - DBFDT400 EMAC . . . . .	488
138. Trace Record 9F73 - DBFDT400 Read . . . . .	488
139. Trace Record 9F74 - DBFDT400 Write . . . . .	489
140. Database Tracking Trace Entries for X'D4' Trace Entry . . . . .	489
141. Trace Record 3701 - Data Set Services Control Routine Entry . . . . .	492
142. Trace Record 3702 - Create Data Set Routine Invoke DYA . . . . .	493
143. Trace Record 3703 - Create Data Set Routine Exit . . . . .	493
144. Trace Record 3704 - Allocate Data Set Routine Exit . . . . .	494
145. Trace Record 3705 - Open Data Set Routine Exit . . . . .	494
146. Trace Record 3707 - Deallocate/Delete Data Set Routine Exit . . . . .	494
147. Trace Record 3709 - End of Merge . . . . .	495
148. Trace Record 370E - Received Last Buffer of the Active Stream . . . . .	495
149. Trace Record 370F - Routed Log Records from Buffer to Trackers . . . . .	496
150. Trace Record 3710 - Active Stream Tracker RSR04_PTKO . . . . .	496
151. Trace Record 3712 - Active Stream Tracker RSR04SUB . . . . .	497
152. Trace Record 3731 - Stream Archiver Controller Entry . . . . .	497
153. Trace Record 3732 - Stream Archiver Controller Exit . . . . .	498
154. Trace Record 3733 - Stream Archiver WRITE Invocation. . . . .	499
155. Trace Record 3734 - Stream Archiver Switch Data Set . . . . .	499
156. Trace Record 3736 - Stream Archiver Log Truncation Start Exit . . . . .	500
157. Trace Record 3737 - Log Router Log Truncation Exit . . . . .	500
158. Trace Record 3738 - Log Router Log Read Controller Exit . . . . .	501
159. Trace Record 373A - Log Router Log Reader First Read Request . . . . .	501
160. Trace Record 373B - Log Router Log Reader Buffer Return . . . . .	502
161. Trace Record 373C - Log Router Log Reader Reread Data Set Request. . . . .	503
162. Trace Record 373D - Log Router Log Reader Exit . . . . .	503
163. Trace Record 373E - Log Router Start Log Reader Entry . . . . .	504

164.	Trace Record 3740 - DFSLRCAS Create Active Stream New Stream . . . . .	504
165.	Trace Record 3741 - DFSLRCAS Create Active Stream Allocate Conversation . . . . .	505
166.	Trace Record 3742 - DFSLRCAS Create Active Stream Set Position . . . . .	505
167.	Trace Record 374F - DFSLRASC Active Stream Control Entry . . . . .	505
168.	Trace Record 3750 - Initiate Online Forward Recovery (OFR) . . . . .	505
169.	Trace Record 3751 - Create the OFR ITASK . . . . .	506
170.	Trace Record 3752 - OFR Processor Request . . . . .	506
171.	Trace Record 3753 - OFR Processor Exit . . . . .	506
172.	Trace Record 3754 - Log Descriptors Obtained from DBRC . . . . .	507
173.	Trace Record 3756 - Log Descriptors Obtained from DBRC . . . . .	507
174.	Trace Record 3757 - Log Descriptors Obtained from DBRC . . . . .	507
175.	Trace Record 3758 - Start Points List Error detected . . . . .	507
176.	Trace Record 3760 - DFSLRARC Auto Archive Controller Entry . . . . .	508
177.	Trace Record 3760 - DFSLRARC Auto Archive Controller Entry . . . . .	508
178.	Trace Record 3761 - DFSLRARC Auto Archive Controller Exit. . . . .	509
179.	Trace Record 3762 - DFSLRARP Auto Archive Processor Entry . . . . .	509
180.	Trace Record 3762 - DFSLRARP Auto Archive Processor Entry . . . . .	509
181.	Trace Record 3762 - DFSLRARP Auto Archive Processor Entry . . . . .	510
182.	Trace Record 3762 - DFSLRARP Auto Archive Processor Entry . . . . .	511
183.	Trace Record 3763 - DFSLRARC Get LDS List from DBRC . . . . .	512
184.	Trace Record 3764 - DFSLRARP After Create Log Reader . . . . .	512
185.	Trace Record 3765 - DFSLRARP Enqueue Buffer to Write . . . . .	513
186.	Trace Record 3770 - Isolated Log Transport Control Routine Entry . . . . .	513
187.	Trace Record 3771 - Isolated Log Transport Control Routine Exit . . . . .	514
188.	Trace Record 3772 - Isolated Log Transport Send Routine Entry. . . . .	514
189.	Trace Record 3773 - Isolated Log Transport Schedule Control Message . . . . .	514
190.	Trace Record 3774 - Isolated Log Transport Gap Fill . . . . .	514
191.	Trace Record 3775 - Isolated Log Transport Query Response. . . . .	514
192.	Trace Record 3776 - Isolated Log Transport DS Abort. . . . .	515
193.	Trace Record 3777 - Isolated Log Transport Receive DS . . . . .	515
194.	Trace Record 3778 - Isolated Log Transport Send OK . . . . .	515
195.	Trace Record 3779 - Isolated Log Transport DS Received . . . . .	515
196.	Trace Record 377A - Isolated Log Transport DS Abort . . . . .	516
197.	Trace Record 3780 - Milestone Request Entry . . . . .	516
198.	Trace Record 3781 - Milestone Complete . . . . .	516
199.	Trace Record 3782 - Unplan Takeover Process Phase 1 Entry . . . . .	517
200.	Trace Record 3783 - Unplan Takeover Process Phase 2 Entry . . . . .	517
201.	Trace Record 3784 - Log Router Master ITASK Request. . . . .	517
202.	Trace Record 3785 - Log Router Master ITASK Request Done . . . . .	517
203.	Trace Record 3786 - Log Router Master ITASK Exit . . . . .	518
204.	Trace Record 3787 - Log Router End DataBase Tracking . . . . .	518
205.	Trace Record 3788 - Create Active Stream Begin Takeover. . . . .	518
206.	X'4930' Log Record Layout . . . . .	519
I 207.	Trace Tables Containing CQS Trace Records . . . . .	527
I 208.	CQS Trace Codes and Mapping Macros. . . . .	527
I 209.	CQS Mapping Macros and Request Trace Records. . . . .	528
210.	CQS Log Records . . . . .	530
I 211.	Trace Tables for OM Trace Records . . . . .	535
I 212.	Trace Tables for RM Trace Records . . . . .	535
I 213.	Trace Tables for SCI Trace Records . . . . .	535
I 214.	CSL Address Space Trace Code Mapping Macros . . . . .	536
I 215.	CSL Address Space Trace Record Mapping Macros . . . . .	536
I 216.	CSL Request Return, Reason, and Completion Codes Mapping Macros . . . . .	537
217.	IMS Keyword Dictionary. . . . .	541
218.	AIBREASN Codes Set by DFSQMRQ0 . . . . .	547
I 219.	DFSQQRQY Return Codes . . . . .	577



220. Load List Areas . . . . .	581
221. CBT Pool Names and Descriptions. . . . .	583
I 222. Fast Path Trace Entries . . . . .	591



---

## About This Book

This information is available as part of the DB2® Information Management Software Information Center for z/OS® Solutions. To view the information within the DB2 Information Management Software Information Center for z/OS Solutions, go to <http://publib.boulder.ibm.com/infocenter/dzichelp>. This information is also 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](http://www.ibm.com/software/data/ims/library.html). To view or download the PDF of this information on the Web, you must enter a valid IMS customer number in the Web form that appears after you click the PDF icon for this book on the IMS Version 9 Library page.

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

---

## 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 63, 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 121, 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 541, contains information that you might need while following the procedures in Chapter 4, “Selecting the Keywords,” on page 31 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 617.

With IMS™ Version 9, you can reorganize HALDB partitions online, either by using the integrated *HALDB Online Reorganization* function or by using an external product. In this information, the term HALDB Online Reorganization refers to the integrated HALDB Online Reorganization function that is part of IMS Version 9, unless otherwise indicated.

IMS Version 9 provides an integrated IMS Connect function, which offers a functional replacement for the IMS Connect tool (program number 5655-K52). In this information, the term *IMS Connect* refers to the integrated IMS Connect function that is part of IMS Version 9, unless otherwise indicated.

---

## 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
  - z/OS diagnostic practices
  - Telecommunications
  - System Network Architecture (SNA)
- 

## Product Names

In this information, the licensed programs shown in Table 1 are referred to by their short names.

*Table 1. Licensed Program Full Names and Short Names*

<b>Licensed program full name</b>	<b>Licensed program short name</b>
IBM DB2 Universal Database for z/OS	DB2 UDB for z/OS
IBM DB2 Universal Database	DB2 Universal Database
IBM CICS Transaction Server for z/OS	CICS
IBM CICS Transaction Server for OS/390	CICS
IBM z/OS	z/OS
IBM WebSphere Application Server for z/OS and OS/390	WebSphere® Application Server for z/OS
IBM WebSphere MQ for z/OS	WebSphere MQ
WebSphere Studio Application Developer Integration Edition	WebSphere Studio
IBM IMS DataPropagator for z/OS	IMS DataPropagator™
IBM IMS Database Recovery Facility for z/OS	IMS Database Recovery Facility
IBM IMS Connect for z/OS	IMS Connect
IBM IMS Connector for Java	IMS Connector for Java™
IBM Tivoli NetView for z/OS	Tivoli NetView for z/OS
IBM Application Recovery Tool for IMS and DB2	Application Recovery Tool
IBM IMS Advanced ACB Generator	IMS Advanced ACB Generator
IBM IMS Batch Backout Manager	IMS Batch Backout Manager
IBM IMS Batch Terminal Simulator	IMS Batch Terminal Simulator
IBM IMS Buffer Pool Analyzer	IMS Buffer Pool Analyzer
IBM IMS Command Control Facility for z/OS	IMS Command Control Facility
IBM IMS Database Control Suite	IMS Database Control Suite
IBM IMS Database Repair Facility	IMS Database Repair Facility
IBM IMS DEDB Fast Recovery	IMS DEDB Fast Recovery
IBM IMS Extended Terminal Option Support	IMS ETO Support
IBM IMS Fast Path Basic Tools	IMS Fast Path Basic Tools
IBM IMS Fast Path Online Tools	IMS Fast Path Online Tools
IBM IMS Hardware Data Compression-Extended	IMS Hardware Data Compression-Extended
IBM IMS High Availability Large Database (HALDB) Conversion Aid for z/OS	IBM IMS HALDB Conversion Aid

Table 1. Licensed Program Full Names and Short Names (continued)

Licensed program full name	Licensed program short name
IBM IMS High Performance Change Accumulation Utility for z/OS	IMS High Performance Change Accumulation Utility
IBM IMS High Performance Load for z/OS	IMS HP Load
IBM IMS High Performance Pointer Checker for OS/390	IMS HP Pointer Checker
IBM IMS High Performance Prefix Resolution for z/OS	IMS HP Prefix Resolution
IBM Enterprise COBOL for z/OS and OS/390	Enterprise COBOL
IBM Enterprise PL/I for z/OS and OS/390	Enterprise PL/I
IBM High Level Assembler for MVS & VM & VSE	High Level Assembler

---

## 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 or any other IMS information, you can do one of the following:

- Go to the IMS Library page at [www.ibm.com/software/data/ims/library.html](http://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](mailto:imspubs@us.ibm.com). Be sure to include the title, the part number of the title, the version of IMS, and, if applicable, the specific location of the text you are commenting on (for example, a page number in the PDF or a heading in the Information Center).



---

## Summary of Changes

---

### Changes to This Book for IMS Version 9

New information on the following enhancements is included:

- /DIAGNOSE Command for Serviceability: “/DIAGNOSE Command SNAP Function” on page 190.
- DBRC Enhancements: Table 110 on page 448.
- FP Serviceability/Usability:
  - “Fast Path Trace” on page 251.
  - Table 36 on page 194.
- HALDB Online Reorganization Support:
  - Table 8 on page 127.
  - “Format of X'29' Log Record” on page 141.
- OTMA Serviceability and Usability Enhancements: “OTMA Diagnostic Aids” on page 400.

New chapters and appendixes added:

- Chapter 17, “CSL Diagnosis,” on page 535
- Appendix F, “Fast Path Trace Entries,” on page 591

Significant changes to Chapter 1, “Setting Up Your System,” on page 3, including:

- “Setup Recommendations for z/OS” on page 3
- “Setup Recommendations for IMS” on page 5
- “Management of Standard Documentation” on page 11

Significant changes to Chapter 2, “Collecting Data about Problems,” on page 17, including:

- “Diagnosing a DBRC Related Problem” on page 21
- “Diagnosing CQS-Related Problems” on page 24
- “Diagnosing ESAF Interface Problems” on page 25
- “Diagnosing Database Problems” on page 25
- “Diagnosing RRS Problems” on page 26
- “Diagnosing MSC-Related Problems” on page 27

Significant changes to Chapter 6, “Data Areas and Record Formats,” on page 65, including:

- “DL/I Record Formats” on page 113

Significant changes to Chapter 7, “SYS—System Service Aids,” on page 127, including:

- “Common Trace Table Interface” on page 190
- “Online Recovery Manager Trace” on page 306
- “Common Service Layer Trace” on page 197

Significant changes to Chapter 8, “DB—Database Service Aids,” on page 253, including:

- “DL/I Trace” on page 263

Significant changes to Chapter 16, “CQS Diagnosis,” on page 521, including:

- “CQS Structure Rebuild Problems” on page 525

In addition to the changes mentioned above, this book has undergone a major upgrade since IMS Version 8, with all chapters being reviewed, upgraded, or changed accordingly.

## Library Changes for IMS Version 9

Changes to the IMS Library for IMS Version 9 include the addition of one title, a change of one title, organizational changes, and a major terminology change. Changes are indicated by a vertical bar (|) to the left of the changed text.

### New and Revised Titles

The following list details the major changes to the IMS Version 9 library:

- *IMS Version 9: IMS Connect Guide and Reference*

The library includes new information: *IMS Version 9: IMS Connect Guide and Reference*. This information is available in softcopy format only, as part of the DB2 Information Management Software Information Center for z/OS Solutions, and in PDF and BookManager formats.

IMS Version 9 provides an integrated IMS Connect function, which offers a functional replacement for the IMS Connect tool (program number 5655-K52). In this information, the term *IMS Connect* refers to the integrated IMS Connect function that is part of IMS Version 9, unless otherwise indicated.

- The information formerly titled *IMS Version 8: IMS Java User's Guide* is now titled *IMS Version 9: IMS Java Guide and Reference*. This information is available in softcopy format only, as part of the DB2 Information Management Software Information Center for z/OS Solutions, and in PDF and BookManager formats.
- To complement the IMS Version 9 library, a new book, *An Introduction to IMS* by Dean H. Meltz, Rick Long, Mark Harrington, Robert Hain, and Geoff Nicholls (ISBN # 0-13-185671-5), is available starting February 2005 from IBM Press. Go to the IMS Web site at [www.ibm.com/ims](http://www.ibm.com/ims) for details.

### Organizational Changes

Organization changes to the IMS Version 9 library include changes to:

- *IMS Version 9: IMS Java Guide and Reference*
- *IMS Version 9: Messages and Codes, Volume 1*
- *IMS Version 9: Utilities Reference: System*

The chapter titled "DLIModel Utility" has moved from *IMS Version 9: IMS Java Guide and Reference* to *IMS Version 9: Utilities Reference: System*.

The DLIModel utility messages that were in *IMS Version 9: IMS Java Guide and Reference* have moved to *IMS Version 9: Messages and Codes, Volume 1*.

### Terminology Changes

IMS Version 9 introduces new terminology for IMS commands:

#### type-1 command

A command, generally preceded by a leading slash character, that can be entered from any valid IMS command source. In IMS Version 8, these commands were called *classic* commands.

#### type-2 command

A command that is entered only through the OM API. Type-2 commands are more flexible than type-1 commands and can have a broader scope. In IMS Version 8, these commands were called *IMSplex* commands or *enhanced* commands.

### Accessibility Enhancements

Accessibility features help a user who has a physical disability, such as restricted mobility or limited vision, to use software products. The major accessibility features in z/OS products, including IMS, enable users to:

- Use assistive technologies such as screen readers and screen magnifier software



- Operate specific or equivalent features using only the keyboard
- Customize display attributes such as color, contrast, and font size

### **User Assistive Technologies**

Assistive technology products, such as screen readers, function with the IMS user interfaces. Consult the documentation of the assistive technology products for specific information when you use assistive technology to access these interfaces.

### **Accessible Information**

Online information for IMS Version 9 is available in BookManager format, which is an accessible format. All BookManager functions can be accessed by using a keyboard or keyboard shortcut keys. BookManager also allows you to use screen readers and other assistive technologies. The BookManager READ/MVS product is included with the z/OS base product, and the BookManager Softcopy Reader (for workstations) is available on the IMS Licensed Product Kit (CD), which you can download from the Web at [www.ibm.com](http://www.ibm.com).

### **Keyboard Navigation of the User Interface**

Users can access IMS user interfaces using TSO/E or ISPF. Refer to the *z/OS V1R1.0 TSO/E Primer*, the *z/OS V1R5.0 TSO/E User's Guide*, and the *z/OS V1R5.0 ISPF User's Guide, Volume 1*. These guides describe how to navigate each interface, including the use of keyboard shortcuts or function keys (PF keys). Each guide includes the default settings for the PF keys and explains how to modify their functions.



# Part 1. Identifying System Problems

<b>Chapter 1. Setting Up Your System</b>	3
Setup Recommendations for z/OS	3
System Trace Table	3
Common Storage Tracker	4
CHNGDUMP MAXSPACE	4
Automatic Dump Data Set Allocation	4
Ensuring That the Sizes of SYS1.DUMPxx Data Sets Are Correct	5
Setup Recommendations for IMS	5
FMTO Option	5
SYSDUMP DD	6
Table Traces	6
Interactive Dump Formatter	7
External Trace Environment	7
CQS Trace Setup Recommendations	7
Trace Environment - Conservative	7
Trace Environment - More Aggressive	7
Installing the IMS Dump Formatter	8
Setting Up the External Trace Environment	8
Control the Volume of Traces	8
Activate Fast Path Traces	9
Write Trace Tables Externally	9
Create Output Data Sets with Correct Attributes	10
Setting Up CQS, OM, RM, and SCI Tracing	10
Management of Standard Documentation	11
z/OS Console (Syslog) Preservation	11
JES JOBLOG Preservation	11
IMS Master Console Log Preservation	12
SYS1.LOGREC Preservation	12
Dump Preservation	12
IMS OLDS and SLDS Preservation	12
Manual Intervention for Dump Creation	12
Deciding When to Dump	13
IMS Dump Techniques	13
IEADMCxx, z/OS SYS1.PARMLIB	14
IMS Sysplex Dump Considerations	14
<b>Chapter 2. Collecting Data about Problems</b>	17
Collecting Data about General Problems	17
Collecting Data about Specific Problems	18
Diagnosing a Control Region Wait or Hang	18
Diagnosing a Control or DL/I Region Loop	19
Diagnosing an IMS Dependent Region Wait or Loop	20
Diagnosing a DB2 ESS Interface Problem	20
Diagnosing a DBRC Related Problem	21
Diagnosing a DBCTL Related Problem	21
Diagnosing a DC Related Problem	22
Diagnosing an APPC Related DC Problem	23
Diagnosing CQS-Related Problems	24
Diagnosing CSL-Related Problems	24
Diagnosing ESAF Interface Problems	25
Diagnosing Database Problems	25
Diagnosing RRS Problems	26
Diagnosing MSC-Related Problems	27

**Chapter 3. Searching Problem Reporting Databases** . . . . . 29

Developing Search Arguments. . . . . 29

Creating a Search Argument . . . . . 30

**Chapter 4. Selecting the Keywords** . . . . . 31

Component Identification Keyword Procedure . . . . . 31

Type-of-Failure Keyword . . . . . 31

    ABENDxxx Procedure . . . . . 32

    ABENDUxxxx Procedure . . . . . 33

    DOC Procedure . . . . . 35

    PERFM Procedure . . . . . 36

    MSG Procedure . . . . . 37

    INCORROUT Procedure . . . . . 37

    WAIT/LOOP Procedure . . . . . 40

**Chapter 5. Procedures and Techniques** . . . . . 59

Searching the Database . . . . . 59

Searching for APARs Closed within a Specific Time Period . . . . . 60

Preparing an APAR. . . . . 61

---

## Chapter 1. Setting Up Your System

IMS can process large amounts of work efficiently; it is a very complex product. As a result of this complexity, 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 will need to be sure your system is set up correctly. To ensure that you have gathered all of the correct data to diagnose a problem, set up your system according to the recommendations in the following:

- “Setup Recommendations for z/OS”
- “Setup Recommendations for IMS” on page 5
- “CQS Trace Setup Recommendations” on page 7
- “Installing the IMS Dump Formatter” on page 8
- “Setting Up the External Trace Environment” on page 8
- “Setting the z/OS System Trace Table Size” on page 6
- “Setting the z/OS Master Trace Table Size” on page 6
- “Management of Standard Documentation” on page 11
- “Manual Intervention for Dump Creation” on page 12

---

### Setup Recommendations for z/OS

This topic gives specific recommendations on how to optimally set up your z/OS system:

- “System Trace Table”
- “Common Storage Tracker” on page 4
- “CHNGDUMP MAXSPACE” on page 4
- “Automatic Dump Data Set Allocation” on page 4
- “Ensuring That the Sizes of SYS1.DUMPxx Data Sets Are Correct” on page 5

### System Trace Table

Set z/OS system trace table size to 999 KB:

- The default size is only 64 KB.
- You can specify the z/OS command `TRACE ST,999K` in the z/OS `COMMNDxx` member of the `SYS1.PARMLIB` data set. See the *z/OS MVS System Commands* manual for more information.
- Advantages:
  - The z/OS system trace table is extremely valuable for a large variety of problem types.
- Considerations:
  - The system trace table is page-fixed storage.
  - You must ensure that there are enough real page frames for this specification.

Set z/OS Master Trace Table size to 500 KB:

- Default size is only 24 KB, which allows approximately 336 messages. A 500K specification allows approximately 7000 messages.

- | • You can specify TRACE MT,500K in the SCHEDxx member of the SYS1.PARMLIB data set. See the *z/OS MVS Diagnosis: Tools and Service Aids*, *z/OS MVS Initialization and Tuning Guide*, and *z/OS MVS System Commands* manuals for complete details.
- | • Advantages:
  - | – The master trace maintains a table of the most recently issued operator messages.
  - | – The master trace allows view of external events at the time of failure.
- | • Considerations:
  - | – Ensure that the master trace table is large enough to span most error time frames.
  - | – The master trace uses Subpool 229 Key 0 High Private Pageable Storage of the master scheduler address space.

## | **Common Storage Tracker**

| To track ownership of the Common Service Area (CSA) and the Extended Common Service Area (ECSA), turn on the z/OS common storage tracking function.

- | • Use the DIAGxx member of the SYS1.PARMLIB data set to contain the request. Specify DIAG=xx in the IPL system parameters or use the SET DIAG=xx operator command.
  - | – For example, in the DIAGxx member:
 

```
VSM TRACK CSA(ON)
```
- | See the *z/OS MVS Diagnosis: Tools and Service Aids*, *z/OS MVS Initialization and Tuning Guide*, and *z/OS MVS System Commands* manuals for complete details.
- | • Advantages:
  - | – Supervisor call (SVC) dumps (or RMF reports) provide CSA/ECSA ownership information with job name, time, and requesting module information.
- | • Considerations:
  - | – Performance can be degraded and extended system queue area (ESQA) is used proportionally to the CSA workload.

## | **CHNGDUMP MAXSPACE**

| Ensure that an adequate CHNGDUMP MAXSPACE value is specified to hold the internal supervisor call (SVC) dump.

- | • Use the COMMNDxx member of the SYS1.PARMLIB data set to issue the appropriate CHNGDUMP command during IPL.
  - | – For example: CD SET,SDUMP,MAXSPACE=1000M
    - | - Default size is 500 MB
    - | - 2500 MB is standard for large multi-address space SVC Dumps.
  - | – See the *z/OS MVS Diagnosis: Tools and Service Aids*, *z/OS MVS Initialization and Tuning Guide*, and *z/OS MVS System Commands* manuals for complete details.
- | • Advantages:
  - | – Higher likelihood that SVC dumps are captured in their entirety without worry of partial dump.
- | • Considerations:
  - | – Ensure that local page data sets are large enough to contain their normal peak load, plus additional SVC dumps.
  - | – See the *z/OS MVS Initialization and Tuning Guide* for more information.

## | **Automatic Dump Data Set Allocation**

| Ensure that automatic dump data set allocation is in place.

- | • Use the COMMNDxx member of the SYS1.PARMLIB data set to issue the appropriate DUMPDS commands to set up dump data set allocations:

- | – DUMPDS NAME=, DUMPDS ADD, and DUMPDS ALLOC=ACTIVE
- | – See the *z/OS MVS Diagnosis: Tools and Service Aids* and *z/OS MVS System Commands* manuals for complete details.
- | • Advantages:
  - | – SVC dumps are allocated to the correct size without worry of partial dump.
- | • Considerations:
  - | – Ensure that the assigned storage class has enough space for the SVC dump storage requirements.

## | Ensuring That the Sizes of SYS1.DUMPxx Data Sets Are Correct

| The SYS1.DUMPxx data sets should be large enough to contain up to 7 IMS regions in one dump data set. IMS attempts to dump the CTL, DL/I, DBRC, IRLM, and possibly one dependent region, into the SYS1.DUMP data set. IMS also attempts to dump the CQS or SCI regions, or both, if they are being used. For some large installations, the required size can be more than 500 cylinders of 3390 DASD. The mixture of IMS specifications, z/OS specifications, and IMS processing requirements produce different usage of storage, 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 z/OS DUMP command to obtain an IMS dump for estimation purposes:

```
| DUMP COMM=(dump title)  
| R id JOBNAME=(j1,j2,j3,j4,j5,j6,j7),  
| 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 job name
- | *j2* is the IMS DL/I region job name
- | *j3* is the Large IMS dependent region job name
- | *j4* is the IRLM region job name (If IRLM DB Locking used)
- | *j5* is the DBRC region job name
- | *j6* is the CQS region name (if IMS is using CQS for shared queues or shared expedited message handling (EMH))
- | *j7* is the SCI region (if IMS is using structured call interface (SCI))

| The SYS1.DUMPxx data sets can be dynamically allocated.

- | • Take the dump of these regions during a period of high workload, if possible.

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

---

## | Setup Recommendations for IMS

| The following topics provide specific recommendations on how to optimally set up your IMS system:

- | • “FMTO Option”
- | • “SYSMDUMP DD” on page 6
- | • “Table Traces” on page 6
- | • “Interactive Dump Formatter” on page 7
- | • “External Trace Environment” on page 7

### | FMTO Option

| Specify the FMTO=D IMS control region EXEC parameter value.

- | • This parameter produces a system dump (SDUMP) for terminating and non-terminating errors, specifically, DB2 and dynamic-allocation abends. Non-terminating errors include:
  - | – IMS dynamic allocation failures.
  - | – Some external subsystem attach facility (ESAF) failures.

| A SYSMDUMP, SYSABEND, or SYSUDUMP is produced only if SDUMP fails.

## | **SYSDUMP DD**

- | • Specify the SYSDUMP DD statement in JCL of the following IMS regions:
  - | – IMS CTL (control)
  - | – IMS DLI (data language interface)/SAS (separate address space)
  - | – IMS DBRC (Database Recovery Control)
- | • The SYSDUMP specification is used by IMS if SDUMP processing fails.
- | • You should specify the following dump options in the SYS1.PARMLIB(IEADMR00) member to ensure that adequate areas of z/OS storage are dumped to diagnose the problem under most circumstances:
 

```
| SDATA=(CSA,LSQA,RGN,SQA,SUM,SWA,TRT)
```
- | • Specify the SYSUDUMP DD statement in JCL of IMS Dependent Regions. The SYSUDUMP specification is used by IMS dependent regions for failure events.
- | • You should specify the following dump options in the z/OS SYS1.PARMLIB(IEADMP00) member to ensure that adequate areas of z/OS storage are dumped:
 

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

## | **Table Traces**

- | • Set the IMS Dispatcher, Scheduler, DLI, and Lock traces on. Perform one of the following:
  - | – The DLI and LOCK traces are set on by default when IMS initializes.
  - | – To set the DISP and SCHED traces on, specify the following options in the DFSVSMxx member of the IMS.PROCLIB data set:
 

```
| DISP=ON, SCHED=ON
```
  - | – Use the IMS /TRA SET ON TABLE *nnnn* command, where *nnnn* is alternately = DISP, SCHED, DLI, or LOCK.
- | • You should turn on the LATCH trace only in non-production environments.
 

| The LATCH trace can carry a large amount of overhead, so it is not recommended as a default in a production environment.

| **Recommendation:** Use the IMS LATCH trace for all test systems. Your system might experience measurable performance reduction if the LATCH trace is active in production. To set the LATCH trace on, specify LATCH=ON for the LATCH trace in the DFSVSMxx member of the IMS.PROCLIB data set. See the section “Member DFSVSMxx” in *IMS Version 9: Installation Volume 2: System Definition and Tailoring*.

## | **Setting the z/OS System Trace Table Size**

| The z/OS system trace is useful for many types of z/OS 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 z/OS command TRACE ST,999K in the z/OS COMMNDxx member of the SYS1.PARMLIB data set so that the trace table size is in effect during IPL. If you do not specify a trace table size, the default size is 64 KB. 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.

## | **Setting the z/OS Master Trace Table Size**

| The z/OS master trace table contains a buffer of messages from the z/OS master console. These messages are saved in the SDUMP data set and can be viewed using IPCS to aid in problem diagnosis.



| Specify the z/OS command TRACE MT,100K in the z/OS SCHEDxx member of the SYS1.PARMLIB data set  
| so that the trace table size is in effect during IPL. If you do not specify a trace table size, the default size  
| is 64 KB.

## | **Interactive Dump Formatter**

| Install the IMS Interactive Dump Formatter:

- | • The Interactive Dump Formatter provides IPCS (Interactive Problem Control System) menu driven dump analysis.
- | • The Interactive Dump Formatter is highly effective for RSV (Remote Screen Viewing) users.

| For more information about using the interactive dump formatter, see “Interactive Dump Formatter” on page 178.

## | **External Trace Environment**

- | • IMS external tracing allows for IMS trace table output to be placed on IMS trace data sets rather than on the IMS OLDS (online data set) when:
  - | – The DISP=OUT option is used in the DFSVSMxx member of the PROCLIB data set.
  - | – The LOG option is used with the IMS TRACE commands.
- | • Using external trace can increase IMS system throughput.
- | • External trace data sets are allocated in the following order:
  - | 1. DASD JCL: DFSTRA01 and DFSTRA02 DD statement.
  - | 2. DASD MDA: DFSTRA01 and DFSTRA02 Dynamic Allocation Members.
  - | 3. TAPE MDA: DFSTRA0T Dynamic Allocation Member.
  - | 4. IMS OLDS: If none of the above are found.

| For more information, see “Setting Up the External Trace Environment” on page 8.

---

## | **CQS Trace Setup Recommendations**

| This topic gives specific recommendations on how to optimally set up your CQS (Common Queue Server) system:

- | • “Trace Environment - Conservative”
- | • “Trace Environment - More Aggressive”

### | **Trace Environment - Conservative**

| Specify the CQS execution parameter BPECFG=*nnnnnnnn*:

- | • Specify the following trace entries within the BPECFG=*nnnnnnnn* Proclib member:

```
| --DEFINITIONS FOR BPE SYSTEM TRACES  
| TRCLEV=(AWE,LOW,BPE) /* AWE SERVER TRACE *  
| TRCLEV=(CBS,LOW,BPE) /* CONTROL BLK SRVCS TRACE *  
| TRCLEV=(DISP,LOW,BPE) /* DISPATCHER TRACE *  
| TRCLEV=(LATC,LOW,BPE) /* LATCH TRACE *  
| TRCLEV=(SSRV,LOW,BPE) /* GEN SYS SERVICES TRACE *  
| TRCLEV=(STG,LOW,BPE) /* STORAGE TRACE *  
| TRCLEV=(USRX,LOW,BPE) /* USER EXIT TRACE *  
| --DEFINITIONS FOR CQS TRACES  
| TRCLEV=(CQS,LOW,CQS) /* CQS GENERAL TRACE *  
| TRCLEV=(STR,LOW,CQS) /* CQS STRUCTURE TRACE *  
| TRCLEV=(INTF,LOW,CQS) /* CQS INTERFACE TRACE *
```

### | **Trace Environment - More Aggressive**

| Specify the CQS execution parameter BPECFG=*nnnnnnnn*:

- | • Specify the following trace entries within the BPECFG=*nnnnnnnn* proclib member:

```

|  --DEFINITIONS FOR BPE SYSTEM TRACES
|  TRCLEV=(AWE,HIGH,BPE,PAGES=24)/*AWE SERVER TRACE */
|  TRCLEV=(CBS,MEDIUM,BPE,PAGES=12)/*CONTROL BLK SRVCS TRACE */
|  TRCLEV=(DISP,HIGH,BPE,PAGES=36)/*DISPATCHER TRACE */
|  TRCLEV=(LATC,HIGH,BPE,PAGES=72)/*LATCH TRACE */
|  TRCLEV=(SSRV,HIGH,BPE,PAGES=6)/*GEN SYS SERVICES TRACE */
|  TRCLEV=(STG,LOW,BPE,PAGES=12)/*STORAGE TRACE */
|  TRCLEV=(USRX,MEDIUM,BPE,PAGES=12)/*USER EXIT TRACE */
|  --DEFINITIONS FOR CQS TRACES */
|  TRCLEV=(CQS,HIGH,CQS,PAGES=12)/*CQS GENERAL TRACE */
|  TRCLEV=(STR,MEDIUM,CQS,PAGES=24)/*CQS STRUCTURE TRACE */
|  TRCLEV=(INTF,HIGH,CQS,PAGES=24)/*CQS INTERFACE TRACE*/

```

---

## Installing the IMS Dump Formatter

Install the IMS interactive dump formatter. For more information on installing the IMS Dump Formatter, see the section titled “IMS Dumping and Dump Formatting Options” in *IMS Version 9: Installation Volume 2: System Definition and Tailoring*.

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 (Interactive Systems Productivity Facility) dialogue to allow you to view a specific control block.

See “Interactive Dump Formatter” on page 178 for more information about using the interactive dump formatter.

---

## Setting Up the External Trace Environment

You can request external tracing by starting traces with the OUT option, or by entering the `/TRACE SET ON TABLE xxxxx OPTION LOG` command to start the trace with the LOG option.

You can start certain traces at IMS initialization 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.

See section “Member `DFSVSMxx`” in *IMS Version 9: Installation Volume 2: System Definition and Tailoring* for more information.

The following topics provide additional information:

- “Control the Volume of Traces”
- “Activate Fast Path Traces” on page 9
- “Write Trace Tables Externally” on page 9
- “Create Output Data Sets with Correct Attributes” on page 10
- 

## 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 9: Command Reference*. For details about the `OPTIONS` statement in the `DFSVSAMP` or `DFSVSMxx` data set, see *IMS Version 9: 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.

**Note:** The DL/I and LOCK traces are set on as a default at IMS initialization.

## 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 9: 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 in memory trace tables to an external device, tape data set, or to the OLDS (online data set).

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:

- I **DSORG** PS (Physical 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:

- I **DSORG** PS (Physical 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
```

---

## Setting Up CQS, OM, RM, and SCI Tracing

The PROCLIB member that you specify using the `BPECFG=` parameter in the CQS (common queue server), OM (Operations Manager), RM (Resource Manager), and SCI (Structured Call Interface) execution parameters defines configuration parameters to BPE. The `TRCLEV=` parameter is used in the BPE configuration PROCLIB member 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, CQS, OM, RM, and SCI support. These trace tables are internal in memory tables only. Trace records are not written to any external data sets.

- I Specify the following trace entries within the `BPECFG=nnnnnnnn` Proclib member:

```
| --DEFINITIONS FOR BPE, CQS, OM, RM AND SCI SYSTEM TRACES
| TRCLEV=(*,HIGH,BPE) /*DEFAULT ALL BPE TRACES TO HIGH*/
| TRCLEV=(*,HIGH,CQS) /*DEFAULT ALL CQS TRACES TO HIGH*/
| TRCLEV=(*,HIGH,OM) /*DEFAULT ALL OM TRACES TO HIGH*/
| TRCLEV=(*,HIGH,RM) /*DEFAULT ALL RM TRACES TO HIGH*/
| TRCLEV=(*,HIGH,SCI) /*DEFAULT ALL SCI TRACES TO HIGH*/
```

| **Related Reading:** For more information see the section titled "BPE Definition and Tailoring" in the *IMS Version 9: Base Primitive Environment Guide and Reference* and "CQS Trace Setup Recommendations" on page 7.

---

## | Management of Standard Documentation

| This topic discusses how to preserve documentation that can be helpful near the time of error. Consider implementing normal operating procedures for the following tasks:

- | • "z/OS Console (Syslog) Preservation"
- | • "JES JOBLOG Preservation"
- | • "IMS Master Console Log Preservation" on page 12
- | • "SYS1.LOGREC Preservation" on page 12
- | • "Dump Preservation" on page 12
- | • "IMS OLDS and SLDS Preservation" on page 12

## | z/OS Console (Syslog) Preservation

| The z/OS Console should be saved to view relevant system messages:

- | • The ideal time frame:
  - | – Back to the last IMS restart
  - | – z/OS Console from the prior clean execution (for comparison)
- | • The moderate time frame:
  - | – 24 hours of z/OS Console messages
- | • The minimum time frame:
  - | – Two IMS system checkpoint intervals

## | JES JOBLOG Preservation

| The JES JOBLOG should be saved to view relevant Job related messages:

- | • Save the JES JOBLOGs for:
  - | – The IMS control region
  - | – The IMS DLI/SAS region
  - | – The IMS DBRC region
  - | – Any suspicious IMS Dependent Regions
  - | – The CQS regions
  - | – The OM region
  - | – The RM region
  - | – The SCI regions
- | • The ideal time frame:
  - | – JES JOBLOG from the current error execution
  - | – JES JOBLOG from the prior clean execution (for comparison)
- | • The moderate time frame:
  - | – 24 hours of JES JOBLOG
- | • The minimum time frame:

- | – Two IMS system checkpoint intervals, or two hours, whichever is greater

## | **IMS Master Console Log Preservation**

| The IMS Master Console Log should be saved to view relevant IMS messages:

- | • The ideal time frame:
  - | – IMS Master Console Log from the current error execution
  - | – IMS Master Console Log from the prior clean execution (for comparison)
- | • The moderate time frame:
  - | – 24 hours of IMS Master Console
- | • The minimum time frame:
  - | – Two IMS system checkpoint intervals or two hours, whichever is greater

## | **SYS1.LOGREC Preservation**

| The SYS1.LOGREC should be saved to view system failures logged internally:

- | • The ideal time frame:
  - | – Back to the last IMS restart
- | • The moderate time frame:
  - | – 48 hours of SYS1.LOGREC data
- | • The minimum time frame:
  - | – Current SYS1.LOGREC data set

## | **Dump Preservation**

| All associated IMS dumps should be retained:

- | • SYS1.DUMP data sets should be examined:
  - | – Multiple dumps might be created.
  - | – Keep all dumps at time of failure, regardless of the subsystem.
- | • SYSMDUMP for the IMS Control, DLI/SAS, and DBRC regions need to be examined in case of primary SYS1.DUMP failures.
  - | – Save these data sets, if a dump was produced.
- | • SYSUDUMP should be saved for IMS dependent regions.

## | **IMS OLDS and SLDS Preservation**

| The IMS OLDS and SLDS should be saved in case IMS log analysis is required:

- | • The ideal time frame:
  - | – From the time of the last IMS restart
  - | – Prior execution
- | • The moderate time frame:
  - | – 24 hours of IMS log records
- | • The minimum time frame:
  - | – Active IMS OLDS

---

## | **Manual Intervention for Dump Creation**

| IMS produces SDUMPs for some internal errors without manual intervention. However, IMS Wait/Loop or partial loss-of-function conditions require manual intervention to produce an SVC dump. IMS hangs can be caused by interaction with many address spaces, including those shown in the list below:

- | • IMS control region

- | • IMS DLI/SAS region
- | • DBRC region
- | • IRLM region
- | • CQS
- | • Operations Manager
- | • Resource Manager
- | • Structure Call Interface
- | • Troublesome IMS dependent regions
- | • CCTL regions
- | • ODBA
- | • IXGLOGRC
- | • RRS
- | • APPC
- | • VTAM
- | • WLM
- | • TCPIP
- | • WebSphere
- | • ESAF - DB2, MQSeries, others
- | • Other regions
- | • Other IMSplex members with all their related regions

| Dumps are discussed in the following sections:

- | • “Deciding When to Dump”
- | • “IMS Dump Techniques”
- | • “IEADMCxx, z/OS SYS1.PARMLIB” on page 14
- | • “IMS Sysplex Dump Considerations” on page 14

## | **Deciding When to Dump**

- | • Because of the complex interactions between these address spaces, it is difficult to determine exactly where the source of the problem lies without a dump of the associated address spaces.
- | • Omission of any interrelated address space adds to the possibility that the dump might not be sufficient to solve the problem.
- | • The time that is required to produce the dump must be weighed against the possibility that there might not be sufficient data to solve the problem, adding to the possibility that the problem could recur.

## | **IMS Dump Techniques**

| IMS SVC dumps can be requested using various techniques:

- | • z/OS SYS1.PARMLIB IEADMCxx
  - | – DUMP command parmlib member available with OS/390 V2R6.0 and later
- | • z/OS SYS1.PARMLIB IEASLPxx
  - | – SLIP command parmlib member
- | • z/OS DUMP command
- | • Customized JCL can be built and submitted

## IEADMCxx, z/OS SYS1.PARMLIB

The following are characteristics of the IEADMCxx member of the z/OS SYS1.PARMLIB data set:

- DUMP command parmlib member (for OS/390 operating systems V2R6.0 and later).
- Can be used to customize IMS dumps prior to error event.
- Simple operator interface.
- Create SYS1.PARMLIB members called IEADMCxx for each customized dump command. See *z/OS MVS Initialization and Tuning Guide* and *z/OS MVS System Commands* for more detailed information.

### IEADMCxx Example for IMS

Create a SYS1.PARMLIB member called IEADMC11 containing the following DUMP parameters:

```
JOBNAME=(j1,j2,j3,j4),SDATA=(CSA,PSA,RGN,SQA,SUM,TRT,GRSQ)
```

Where:

*j1* IMS Control region job name.

*j2* IMS DL/I region job name.

*j3* DBRC region job name.

*j4* IRLM region job name.

Create a second SYS1.PARMLIB member called IEADMC12 containing the following DUMP parameters:

```
JOBNAME=(j5,j6,j7),SDATA=(CSA,PSA,RGN,SQA,SUM,TRT)
```

Where:

*j5* IMS CCTL region 1.

*j6* IMS CCTL region 2.

*j7* IMS CCTL region 3.

= J7 = j6 = IMS CCTL Region 2.

### IEADMCxx DUMP Activation

To request a dump from the IEADMC11 and IEADMC12 parmlib members, enter the following z/OS command:

```
DUMP TITLE=(DUMP OF IMS and CCTL Regions ),PARMLIB=(I1,I2)
```

Two dump data sets are created on the z/OS image from which the dump command was entered.

## IMS Sysplex Dump Considerations

The following are considerations for IMS sysplex dumps:

- IMS sysplex implementations need to consider the possibility that a hang or problem on one IMSplex member might be due to a problem originating from another member.
- Problems such as IMS Wait/Loops or partial loss-of-function conditions which require manual intervention to produce an SVC dump, should include SVC dumps from other members of the IMSplex.
- Ensure that a dump is taken for all necessary address spaces on each system.

### Sysplex IEADMCxx Example

Create a SYS1.PARMLIB member called IEADMC11 containing the following DUMP parameters:

```
JOBNAME=(j1,j2,j3,j4),SDATA=(CSA,PSA,RGN,SQA,SUM,TRT,GRSQ),  
REMOTE=(SYSLIST=(*'j1','j2','j3','j4'),SDATA))
```

Where:



- | *j1*      IMS Control region job name.
- | *j2*      IMS DLI region job name.
- | *j3*      DBRC region job name.
- | *j4*      IRLM region job name.

| Create a second SYS1.PARMLIB member called IEADMCI2 containing the following DUMP parameters:

- | JOBNAME=(*j5*,*j6*,*j7*),SDATA=(CSA,PSA,RGN,SQA,SUM,TRT,XESDATA),
- | REMOTE=(SYSLIST=(\*'j5','j6','j7'),SDATA))

| Where:

- | *j6*      CCTL region 1.
- | *j7*      CCTL region 3.
- | *j8*      CCTL region 2.

| **Note:** The XESDATA and REMOTE parameters are for use in sysplex environments.

### | **Sysplex IEADMCxx DUMP Activation**

| To request a dump from the IEADMCI1 and IEADMCI2 parmlib members, enter the following z/OS command:

- | DUMP TITLE=(IMS/CCTL SYSPLEX Dumps),PARMLIB=(I1,I2)

| Two dump data sets are created on each z/OS image in the sysplex matching the REMOTE parameter specifications for the JOBNAMEs.



---

## Chapter 2. Collecting Data about Problems

When you report a problem to the IBM Support Center, it is very important that you collect the problem information to help document 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 document 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 to search the symptom database with.
4. Perform the search. You might have to refine your search with more data from the problem.
5. If you cannot find a known problem with the same symptoms, report the problem to IBM.

### **In this section:**

- “Collecting Data about General Problems”
- “Collecting Data about Specific Problems” on page 18

---

## Collecting Data about General Problems

Depending on the complexity of the problem, you might 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 contain all necessary error messages.
- **LOGREC data set**  
Save the LOGREC data set from IMS start up time. z/OS 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.
- **z/OS log data sets produced**  
Save the current z/OS log data sets for the failing CQS jobstream. The z/OS 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. These types of problems are discussed in the following:

- “Diagnosing a Control Region Wait or Hang”
- “Diagnosing a Control or DL/I Region Loop” on page 19
- “Diagnosing an IMS Dependent Region Wait or Loop” on page 20
- “Diagnosing a DB2 ESS Interface Problem” on page 20
- “Diagnosing a DBRC Related Problem” on page 21
- “Diagnosing a DBCTL Related Problem” on page 21
- “Diagnosing a DC Related Problem” on page 22
- “Diagnosing an APPC Related DC Problem” on page 23
- “Diagnosing CQS-Related Problems” on page 24
- | • “Diagnosing CSL-Related Problems” on page 24
- | • “Diagnosing ESAF Interface Problems” on page 25
- | • “Diagnosing Database Problems” on page 25
- | • “Diagnosing RRS Problems” on page 26
- | • “Diagnosing MSC-Related Problems” on page 27

## 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 is the z/OS SVC dump.

**Recommendation:** Do not use the z/OS 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 a z/OS 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 job name
- j2* is the IMS DL/I region job name
- j3* is the suspicious IMS dependent region job name, if any
- j4* is the suspicious CCTL (CICS) region name, if any
- j5* is the IRLM region job name (if IRLM DB locking is used)
- j6* is the DBRC region job name

| Also, consider dumping related regions:

- | • IMS Control region
- | • IMS DLI/SAS region
- | • DBRC region
- | • IRLM region
- | • CQS
- | • Operations Manager
- | • Resource Manager

- | • Structure Call Interface
- | • Troublesome IMS dependent regions
- | • CCTL regions
- | • ODBA
- | • IXGLOGRC
- | • RRS
- | • APPC
- | • VTAM
- | • WLM
- | • TCPIP
- | • WebSphere
- | • ESAF - DB2, MQSeries, others
- | • Other Regions
- | • Other IMSplex members with all their related regions

Most likely, a dump of the IMS CTL, DL/I, 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 z/OS SVC dump will help differentiate normal IMS processing from the problem.

## Diagnosing a Control or DL/I 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 in a separate /TRA command.

2. Set the z/OS system trace table size to 999K and turn on branch tracing with this command:

```
TRACE ST,999K,BR=ON
```

3. Obtain two z/OS SVC dumps of the CTL, DL/I, suspicious dependent region, or CCTL, DBRC, and IRLM regions. Taking a second z/OS SVC dump will help differentiate normal IMS processing from the problem. Obtain a z/OS 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 job name
- j2* is the IMS DL/I region job name
- j3* is the suspicious IMS dependent region job name, if any
- j4* is the suspicious CCTL (CICS) region name, if any
- j5* is the IRLM region job name (if IRLM DB locking is used)
- j6* is the DBRC region job name

- | 4. Reset the z/OS system trace table to its original settings.

- I **Note:** IMSplex partner dumps are probably not required for loop problems, unless they are also looping.

## 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 z/OS 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 z/OS SVC dumps of the CTL, DL/I, suspicious dependent region, or CCTL, DBRC, and IRLM regions. If the problem is a loop, obtain two z/OS SVC dumps of the CTL, DL/I, suspicious dependent region, or CCTL, DBRC, and IRLM regions. Obtaining a second z/OS SVC dump will help differentiate normal IMS processing from the problem. Obtain a z/OS 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 job name
- j2* is the IMS DL/I region job name
- j3* is the suspicious IMS dependent region job name, if any
- j4* is the IRLM region job name (if IRLM DB locking is used)
- j5* is the DBRC region job name

4. Reset the z/OS system trace table to its original settings.

- I **Note:** IMSplex partner dumps are probably not required for loop problems, unless they are also looping.

## 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 a z/OS 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 UDB for OS/390 and z/OS 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 Recovery 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 Recovery Control Utility (DSPURX00) LIST.SUBSYS ALL command to obtain a subsystem listing.
- If recreates are possible, obtain them before and after copies of the RECONS.
- Use the D GRS,CONTENTION command on each system sharing the RECON to determine if the RECON is held at the exclusion of other waiters. If so, dump the owning address space:  
DUMP COMM=(dump title)  
R nn,JOBNAME=(j1),SDATA=(CSA,PSA,RGN,SQA,SUM,TRT,GRSQ),END

## Diagnosing a DBCTL Related Problem

DBCTL related problems can be centered in either the CCTL region or in one of the IMS regions (CTL, DL/I, 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 AP 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 z/OS 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 job name
- j2* is the IMS DL/I region job name
- j3* is the suspicious IMS dependent region job name, if any
- j4* is the suspicious CCTL (CICS) region name, if any

*j5* is the IRLM region job name (if IRLM DB locking is used)

*j6* is the DBRC region job name

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 is captured in the IMS TYPE6701 log record. Save the IMS online log data set input to the IMS utility programs DFSERA10 and DFSERA30 or the Knowledge-Based Log Analysis (KBLA) program.

```
/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)
```

**Note:** GTF must be active with the USR option to capture these trace entries.

4. Obtain a z/OS 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 job name

*j2* is the IMS DL/I region job name

*j3* is the suspicious IMS dependent region job name, if any

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

*j5* is the IRLM region job name (if IRLM DB locking is used)

*j6* is the DBRC region job name

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 IMS 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 XXXXXXXX INPUT
TRACE SET ON LUNAME XXXXXXXX OUTPUT
```

where XXXXXXXX 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=1uname
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 z/OS APPC component trace with this command:

```
| TRACE CT,ON,200M,COMP=SYSAPPC
```

5. 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.

6. Reply to the z/OS outstanding reply with the following response:

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

7. When the problem has been recreated, stop the component trace with this command:

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

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

- For one-line entries:

```
| CTRACE COMP SYSAPPC SHORT
```

- Summary of each entry:

```
| CTRACE COMP SYSAPPC FULL
```

8. Obtain a z/OS 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 job name

*j2* is the IMS DL/I region job name

*j3* is the suspicious IMS dependent region job name, if any

- j4* is the suspicious CCTL (CICS) region name, if any
- j5* is the IRLM region job name (if IRLM DB locking is used)
- j6* is the DBRC region job name

9. 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 job name
- j2* is the APPC scheduler job name
- j3* is the VTAM job name

## Diagnosing CQS-Related Problems

CQS problems can take on various appearances and, like the IMS control region, they can manifest themselves in the form of WAITs, HANGs, LOOPs, or some other kind of internal error that results in an SDUMP being taken. These dumps are then found in the SYS1.DUMP data sets. CQS can also produce LOGREC data set entries for these types of errors.

If an isolated event type within CQS encounters an error, then the IBM Support Center might request additional CQS-trace level settings for the various trace types. See “CQS Trace Setup Recommendations” on page 7 and “Setting Up CQS, OM, RM, and SCI Tracing” on page 10 for more information about the traces.

For a CQS WAIT problem, one or more inflight dumps might be required. Multiple dumps might need to be taken if the problem is a LOOP. 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.

**Related Reading:** See Chapter 16, “CQS Diagnosis,” on page 521 for diagnostic information to help analyze the CQS problem itself. Also, see this section for more information on specific CQS problems and steps to follow so that appropriate documentation can be gathered to capture relevant diagnostic information which can enable the IBM Support Center to perform problem determination.

## Diagnosing CSL-Related Problems

The Common Service Layer address spaces, Operations Manager, Structured Call Interface, and Resource Manager produce SDUMPs for internal errors. The CSL dumps can be found in the SYS1.DUMP data sets.

You might need to collect one or more of the following types of information to diagnose CSL related problems:

### 1. SYSLOG

To determine the sequence of events, collect the SYSLOG from every logical partition (LPAR) where a CSL member resides. CSL address spaces issue messages that begin with “CSL”:

- OM messages - CSLOxxxx
- RM messages - CSLRxxxx
- SCI messages - CSLSxxxx
- CSL common messages - CSLZxxxx

### 2. QUERY IMSPLEX SHOW(ALL) command output

| Issue the QUERY IMSPLEX command to display the members of the IMSplex and their status.  
 | If there are problems accessing OM or RM services, verify that at least one OM or RM is active in the  
 | IMSplex and that an active SCI resides on every LPAR where a CSL address space resides.

### | 3. Obtain z/OS SVC dumps

| Obtain a z/OS SVC dump of the CSL address spaces that appear to have a problem, are waiting, or  
 | are looping. CSL dumps contain the CSL traces, which can be very useful for diagnosing CSL related  
 | problems. Dump all of the CSL address spaces that appear to have a problem with the following series  
 | of commands:

```
| DUMP COMM=(dump title)
| R id JOBNAME=(om1,rm1,sci1)
| SDATA=(CSA,PSA,RGN,SQA,SUM,TRT),END
```

| In the previous example:

| *om1* An OM address space.

| *rm1* An RM address space.

| *sci1* An SCI address space.

| For some CSL problems, the IBM Support Center might request additional trace level settings for the  
 | various trace types. See “Setting Up CQS, OM, RM, and SCI Tracing” on page 10 for information  
 | about trace descriptions.

| **Related Reading:** See Chapter 17, “CSL Diagnosis,” on page 535 for additional details on diagnosing  
 | CSL related problems.

## | Diagnosing ESAF Interface Problems

| This topic discusses problems involving the ESAF (External Subsystem Attach Facility). To begin  
 | troubleshooting:

- | • Turn on the subsystem trace to the external trace data set:
  - | – IMS command: /TRA SET ON TABLE SUBS OLDS
- | • Take an SVC dump of the related regions before and after the recreate attempt.
  - | – Ensure that pertinent regions for the affected subsystem (dbtmstr, db2dbm1, mqseries) are included  
 | with the other related IMS regions.

| **Note:** IMSplex partner dumps are probably not required for ESAF problems, unless they are also  
 | experiencing the problem.

## | Diagnosing Database Problems

| This topic discusses database problems. For database problems, obtain the following information:

- | • The damaged database data set.
- | • The database image copy of the damaged database in a state prior to damage.
- | • The image copy of logically related databases.
- | • The IMS OLDS from all data-sharing IMS subsystems.
  - | – Save from the last good database image copy of damaged database.
- | • If possible, and not already set, use the following IMS commands and save the output:

```
| /TRA SET ON TABLE DLI OPTION LOG
```

| and

```
| /TRA SET ON TABLE LOCK OPTION LOG
```

- | • The SYSOUT from the Pointer Checker jobs for the damaged database.
- | • The SYSOUT from batch jobs that accessed the damaged database.
- | • The LIST.RECON and LIST.HISTORY DBD from the damaged database.

- | • The SMF 60, 62, and 64 records from all data-sharing systems back to the last good image copy of damaged database.
- | • For VSAM data sets:
  - | – Issue IDCAMS LISTC for the damaged VSAM data set.
  - | – Issue IDCAMS DIAGNOSE and IDCAMS EXAMINE for the damaged VSAM KSDS data sets.

## | Diagnosing RRS Problems

| If you use RRS (Recovery Resource Services) with your IMS system, diagnostics can be enhanced by the following steps:

- | • Take an SVC dump of the standard IMS regions using one of the methods discussed earlier: CTL, DL/I, DBRC, suspicious dependent regions, IRLM, and so on.
  - | – In addition, include the z/OS RRS address space and the z/OS logger address space (IXGLOGR).
    - | - Consider setting the following SLIP trap to supplement standard IMS/RRS ABENDU0711 diagnostics:
 

```
| SLIP SET,C=U0711,JOBLIST=(ctljname,rrsjname,IXGLOGR),
| SDATA=(CSA,PSA,RGN,SQA,SUM,TRT,GRSQ,LPA,ALLNUC),
| ID=nnnn,DSPNAME=('RRS'.*),END
| ---
```

| In the above example:

<b>ctljname</b>	IMS control region job name
<b>rrsjname</b>	RRS region job name
<b>nnnn</b>	Name used to recognize this SLIP

- | • Turn on the RRS component trace.
  - | – Place the following statements in the CTIRRSxx PARMLIB member:
 

```
| TRACEOPTS
| ON
| BUFSIZE(8M)
| OPTIONS('EVENTS(ALL)')
```
  - | – Place the following statement in the z/OS COMMNDxx SYS1.PARMLIB member:
 

```
| TRACE CT,ON,COMP=SYSRRS,500M,PARAM=CTIRRSxx
```

| **Note:** This statement allows the trace to be active at IPL.

- | – Use the D TRACE,COMP=SYSRRS command to view the current trace setting.
- | – RRS component trace is present in the RRS address space.
- | – Format the trace by using IPCS CTRACE COMP(SYSRRS) FULL command.
- | • Save the IMS OLDS
  - | – IMS 67D0 log records are produced for some ABENDU0711s.
    - | - Print these records by using the IMS utility programs DFSERA10 and DFSERA30.
  - | – Other RRS related records that are produced:
    - | - TYPE4098 - Checkpoint for RRS/MVS log name.
    - | - TYPE5615 - IMS restarted with RRS.
    - | - TYPE5616 - Start of protected UOW.
  - | – Issue two or three IMS DISPLAY UOR ALL commands to show status about the IMS UOR for protected resources on the RRS/MVS recovery platform.
    - | - The RRS-URID provided by RRS and the IMS recovery token are displayed.
- | • If the problem is recreatable, then:
  - | – Turn on the RRS component trace:

```
| TRACE CT,ON,500M,COMP=SYSRRS  
| nn,OPTIONS=(EVENTS(ALL)),END
```

- When the problem has been recreated, stop the component trace:

```
| TRACE CT,OFF,COMP=SYSRRS
```

- RRS component trace is present in the RRS address space.
- Format the trace by using the IPCS CTRACE COMP(SYSRRS) FULL command.
- Issue two or three IMS DISPLAY UOR ALL commands to show status about the IMS UOR for protected resources on the RRS/MVS recovery platform.
  - The RRS-URID provided by RRS and the IMS recovery token are displayed.

## Diagnosing MSC-Related Problems

This topic discusses IMS Multiple Systems Coupling (MSC) related problems. If you use IMS MSC and experience a related problem, do the following tasks:

- Take an SVC dump of the coupled IMS regions (minimally, the CTL regions, but the problem might reside in any IMS—related region). In addition, ensure that the VTAM address space is also included. Do this as close to the time of the problem as possible, prior to attempting to fix the problem.
- Save the IMS OLDS for both coupled systems from the time of the message creation.
- Issue the DISPLAY LINK ALL and DISPLAY LINK ALL MODE commands to show the status and queue counts for the logical link and physical link information for the partner.
- If the problem is recreatable:
  - Turn on the VTAM Internal Trace for both coupled systems:

```
| F NET,TRACE,TYPE=VTAM,OPT=(API,PIU,MSG),DSPSIZE=5,SIZE=999
```
  - Using the options shown above, the VIT (VTAM internal trace) is created in a VTAM data space. After the problem has been recreated, the dump parameters should also include the VTAM data space:

```
| DSPNAME=('NET'.ISTITDS1)
```
  - You should turn on the MSC LINK trace for both coupled systems:

```
| TRACE SET ON LINK link# LEVEL 3 MODULE ALL
```

    - This trace creates IMS TYPE67 records that contain TM control blocks for each message at key points.
- KBLA (Knowledge-Based Log Analysis) provides MSC link performance analysis through an ISPF driven interface. KBLA uses IMS MSC log records to calculate the overall performance of each link that is subject to data traffic in the system.

There are two different report types that are generated:

### Summary

A report that contains the average response time, in milliseconds (msec), of the total number of send and receive data values for each link trace.

**Detail** A report that contains the individual response times, in milliseconds (msec), for every send and receive data value for each MSC link that has been traced.

The formatting of this record is only available if the IMS input log contains X'6701' records that are generated via the /TRACE SET ON LINK *link#* command. For more information, see the “MSC Link Performance Formatting Utility (DFSKMSC0)” section in the *IMS Version 9: Utilities Reference: System*.



---

## 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 541. Each search argument example in the procedures shows a free-form example followed by an SDB example.

### In this section:

- “Developing Search Arguments”
- “Creating a Search Argument” on page 30

---

## 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 B, “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 31 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 59, 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 61.

You might also want to refer to these sections:

- Appendix A, “IMS Keyword Dictionary,” on page 541 provides guidance on translating free-form keywords into structured database (SDB) format.
- Appendix B, “Dependency Keywords,” on page 545 lists words used as search techniques to narrow search arguments.



## Chapter 4. Selecting the Keywords

This section 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.

### **In this section:**

- “Component Identification Keyword Procedure”
- “Type-of-Failure Keyword”

---

### 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 2.

*Table 2. IMS Component Identification Numbers*

5655J3800	IMS Services Database Manager Transaction Manager Extended Terminal Option (ETO) Recovery-level Tracking Database-level Tracking
569516401	Internal Resource Lock Manager (IRLM) 2.1 or 2.2

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 within 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 the DFSADSCT module found in the IMS.ADFSSMPL library.

---

### 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.

<b>ABENDxxx</b>	Use this procedure when the system terminates abnormally with a system abend completion code. An abend produces an SVCDUMP, SYSABEND dump, or SYSUDUMP.  See “ABENDxxx Procedure” on page 32 for more information.
<b>ABENDUxxxx</b>	Use this procedure when an IMS application program terminates abnormally with an abend completion code. An abend produces a SYSABEND dump, SVCDUMP, or SYSUDUMP.  See “ABENDUxxxx Procedure” on page 33 for more information.
<b>DOC</b>	Use this procedure if a deficiency is found in documentation through omission or inaccuracy.  See “DOC Procedure” on page 35 for more information.
<b>PERFM</b>	Use this procedure if performance is other than what is expected.  See “PERFM Procedure” on page 36 for more information.

<b>MSGx</b>	Use this procedure if a problem involves an IMS message. See “MSG Procedure” on page 37 for more information.
<b>INCORROUT</b>	Use this procedure when output is missing or incorrect. See “INCORROUT Procedure” on page 37 for more information.
<b>WAIT/LOOP</b>	Use the WAIT/LOOP procedure when there is no response from IMS functions. See “WAIT/LOOP Procedure” on page 40 for more information.

## 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 33.

After you have developed a search argument, refer to Chapter 5, “Procedures and Techniques,” on page 59 for detailed information on how to use the search argument.

### Keyword: ABENDxxx

Compare the completion code and PSW address in both the z/OS-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.

## 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:

```
5655J3800 ABEND0C4 DFSFXC30
```

For a structured database search, use this search argument:

```
PIDS/5655J3800 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 32 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 9: Messages and Codes, Volume 1* or *IMS Version 9: Messages and Codes, Volume 2*. Then, if further diagnostic information (such as return codes) that you can use to build the search argument is needed, refer to the *IMS Version 9: Failure Analysis Structure Tables (FAST) for Dump Analysis*. 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 59 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 z/OS-formatted section. From the PSW address, scan backward through the eye-catchers until you find a module identifier.

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 9: 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 9: 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 `MSGxxxxxx` with the actual message identifier (for example, the keyword for message `DFS053I` is `MSGDFS053I`).
- Return codes
 

Replace the `xx` part of keyword `RCxx` 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:

```
5655J3800 ABENDU3046
```

For a structured database search, use this search argument:

```
PIDS/5655J3800 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” 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.ADFSSMPL` (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 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

To report problems for a specific IBM IMS manual, use one of the following methods:

- Go to the IMS home page at [www.ibm.com/ims](http://www.ibm.com/ims). There you will find an online feedback page where you can enter and submit comments.
- Send your comment by e-mail to [imspubs@us.ibm.com](mailto: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).

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.

APARs are not generally accepted for documentation errors. However, APARs that correct a programming error can result in documentation changes. You can search for changes to information using the procedure detailed in "Keyword: Order-number."

### Keyword: Order-number

Use this keyword to search for all changes to a specific manual. The format for the order-number is *ppnnnnnee*, where *pp* is the alphabetic prefix, *nnnnnn* is the 6-digit base publication number, and *ee* is the edition number. For example, the order number for *IMS Version 9: Messages and Codes, Volume 1* is GC26-9433-00. Replace *ppnnnnnee* 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 9: Messages and Codes, Volume 1*:

```
5655J3800 GC269433**
```

For a structured database search, use this search argument:

```
PIDS/5655J3800 PUBS/GC269433**
```

You can add more keywords to narrow the search. For example, if you cannot find message DFS3007 in *IMS Version 9: 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 9: 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 59 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:

```
5655J3800 PERFM | PERFORMANCE FAST | PATH | FASTPATH
```

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

```
PIDS/5655J3800 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 31. 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 9: Messages and Codes, Volume 1* and *IMS Version 9: Messages and Codes, Volume 2* describe IMS messages. If, after analyzing the message, you feel the message should not have been issued or describes an error condition, use the MSGxxxxxxx keyword.

After you have developed a search argument, refer to Chapter 5, “Procedures and Techniques,” on page 59 for detailed information on how to use the search argument.

### Keyword: MSGxxxxxxx

Replace the xxxxxxxx part of keyword MSGxxxxxxx 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:

```
5655J3800 MSGDFS3401I
```

For a structured database search, use this search argument:

```
PIDS/5655J3800 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 59 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:

```
5655J3800 INCORROUT DIS | CMDDIS
```

For a structured database search, use this search argument:

```
PIDS/5655J3800 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.”)

### 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:

```
5655J3800 INCORROUT DFSUTR20 DEADLOCK | DMB
```

For a structured database search, use this search argument:

```
PIDS/5655J3800 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 39.
  - 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 39.
  - 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 on page 39.
  - 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 39.
  - 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 40 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 53.
  - 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 A11 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 an 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 43. 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 z/OS 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 41 now.
  - If yes, the problem might be caused by:
    - A data communication 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 43.
  - 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 DL/I 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, DL/I, 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 z/OS 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 43. If SAP analysis does not yield any unusual flows, go to “Receive-Any Buffer Analysis” on page 361.

#### 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 43 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 z/OS formatted dump. Use the IPCS SUMMARY FORMAT ASID(X' \_\_ ') command for the ASID/TCB found in step 6a on page 41. 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 xxxxxxxx is the 8-character RB address, including leading zeros.

**Exception:** Using the last RB in the TCBs 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'00020033'. 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'0002000C'.

**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 z/OS 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 41. You can ignore entries between any SVC and associated SVCR because they reflect necessary z/OS 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 41. 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 for contact with the owning component's IBM Support Center representatives.
  - Use the CECB field obtained in step 6a on page 41 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 3 defines the key fields in SAP analysis.

Table 3. 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.  <b>Exception:</b> 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 DFSQMR0-11/13/94
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 50. 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 50. 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 on page 44 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 \*\*\*\* ACTIVE \*\*\*\* (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 6a in the "WAIT/LOOP Procedure" on page 40. 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 6a because this save area set should probably be ignored. Otherwise, continue.

Record the following key fields from the abnormal save area sets flagged as \*\*\*\* WAITING \*\*\*\*:

- 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 3 on page 43 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 DL/I region dump by using IPCS BROWSE mode for the DL/I 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 4 shows key data from an abnormal save area set.

Table 4. 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. Assemble the module that issued the wait. Use the offset obtained in step 5 on page 44 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, using IPCS BROWSE mode, scan backwards from the EPA address for the actual module name.

7. Repeat steps 5 on page 44 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 8-character module name for this keyword.

**Keyword: WAIT Reason**

The IWAIT REASON column in your worksheet indicates the reason or resource, or both, 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:

```
5655J3800 WAIT LOG | LATCH | W ADS | DFSFLLG0
```

For a structured database search, use this search argument:

```
PIDS/5655J3800 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'C8A' 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 43.

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:

- $SSQ\# \times X'18' - X'8' =$  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 TABSSQnF, 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 49.
  - 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 50.
    - 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 51.
    - 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'C8A') 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 Analysis” on page 46.

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 43. 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 46.

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” on page 40, steps 6c and 6d.

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” on page 40, steps 6c and 6d.

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'C68' (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.  $TCT = 4(\text{first PSTCLASS value} - 1) + X'70'$ .

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'C68' (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'C8A'). 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 3 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'40X'10X'08X'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 51 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 49.
- 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 49.
- 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 or subqueue 5, or both.  
 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 51.
  - b. Otherwise, go to step 5 on page 51.

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” on page 46, step 4 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 9: 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'C4' (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'C8A') (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 48.

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 z/OS system diagnostic procedures for loops.

Using the RB found in step 6c on page 42, 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 z/OS system trace to examine entries for the ASID and TCB indicated in the Dispatch Work Area at step 5 on page 41. The PSW address in the system trace entries indicates the modules involved in the loop.

- | Locate the 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.

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.

**Note:** 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 z/OS 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:

```
5655J3800 LOOP DFSCFE00 | DFSCFEI0 | DISPLAY | NODE
```

For a structured database search, use this search argument:

```
PIDS/5655J3800 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, 5655J3800.

## System Wait

Use standard z/OS 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 topic and in the next topic, “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 z/OS DUMP command to dump the VTAM address space and then modify IMS down with a dump.
- Enter the z/OS 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 information.
- 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'C00') = X'34' and then Set SCDSTOP1(X'C02') = X'80'
  - | - If not PURGE, then:
    - | • If DUMPQ, set SCDCKCTL(X'C00') = X'1C'
    - | • If FREEZE, set SCDCKCTL(X'C00') = X'14'
    - | – Reset POLL the lines and then (not applicable to DBCTL)
    - | – Set SCDSTOP1(X'C02') = X'C0' (for DBCTL, set AWE to TRM1)
- | – DFSICLX0
- | – DFSICIO0
- | – DFSIPCP0
  - | - If SCDCFLG1(X'AC7') = X'08', then
    - | • Set SCDCQFLG(X'AC8') = X'04' and
    - | • Set SCDCNXW4(X'ACF') = 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'AC4') = X'80'
    - | • Set AWE to TRM1
- | – DFSCST00
- | – DFSTRM00

#### For PURGE

- | – AWE = TRM1, First phase of termination
- | – If SCDIDCNT+1(X'BC8') is not equal to X'000000' and SCDCKCTL(X'C00') = X'20' (PURGE):
  - | - Set SCDSTOP1(X'C02') = X'10'
  - | - Set SCDSTOP1(X'C02') = X'02'
- | – If SCDFTFLG(X'290') = X'20' (Fast Path active), DBFTERM0 posts the Fast Path regions for SHUTDOWN
- | – DFSTRM00

#### For DUMPQ or FREEZE

- | – If SCDIDCNT+1(X'BC8') is not equal to X'000000' and SCDCKCTL(X'C00') is not equal to X'20' (Not PURGE)
  - | - Set SCDSTOP1(X'C02') = X'04'
  - | - Set SCDSTOP1(X'C02') = X'02'
- | – If SCDFTFLG(X'290') = 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, and SCDREGCT(X'C8A') is not equal to X'0000' (ACTIVE REGIONS), then post the PSTs waiting in the scheduler.
- | – If SCDSHFL1(X'3A4') = X'80' (IRLM in system) or SCDIDCNT+1(X'BC8'), or both, 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'BC8') = X'000000' (REGIONS ENDED), set SCDSTOP1(X'C02') = X'01'.

|   **For PURGE only**

- |           – If SCDCKCTL(X'C00') = X'20' (PURGE)
- |           – Set SCDSTOP1(X'C02') = 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'290') = X'20' (Fast Path active), DBFTERM1 closes the areas.
- |           – If SCDFTFLG(X'290') is not equal to X'20' or when Fast Path areas are closed then:
  - |               – If SCDSMMS1(X'033') = X'02' (DLI SAS), then:
    - |                   • 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'C02') = X'04'.

|   – DFSCPCP0

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

|   – DFSIPCP0 (DFSIPCP2)

- |           – 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.

|   – DFSCPCP0

- |           – If output is pending (RC = 4)
  - |               • Set SCDCPCTL(X'AC4') = X'08'
  - |               • Set SCDSTOP1(X'C02') = X'40'
  - |               • IWAIT for DC to finish.
- |           – If no output or when output finishes
  - |               • Set off SCDCPCTL(X'AC4') = X'08' (reset the bit)
  - |               • Set SCDSTOP1+1(X'C02') = 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'718') = X'80'
  - |               • Set SCDCPCTL(X'AC4') = 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, then set SCDSTOP1+1(X'C02') = X'01'
- | - DFSRCRT0
- | - DFSRCP00
  - | - Send "DFS994I \*CHKPT yyddd/hhmmss\*ctype" (first part of DFS994I message)
  - | - Set AWE = "TRM2"
  - | - Set off SCDCKCTL(X'C00') = X'04' (reset the bit)
- | - DFSTRM00
  - | Set SCDTRMFL(X'430') = 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'C32') = X'80' (Fast Path errors):
    - | • Print error message
    - | • Set off SCDRFPIN(X'C32') = X'80' (reset the bit)
    - | • Close queue data sets (not applicable to DBCTL)
    - | • IWAIT for closing
    - | • Set off SCDSTOP1(X'C02') = X'08' (reset the bit)
- | - DFSTERM0
  - | - Terminate DASD log
  - | - Set off SCDRECTL(X'146') = 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'D1C') is not equal to X'40' (posted) :
    - | • IWAIT for the DL/I region to end
    - | • Set AWE = "TRM3"
    - | • Set SCDTRMFL(X'430') = X'20'
    - | • Send "DFS994I IMS SHUTDOWN COMPLETED" (second part of DFS994I message)
- | - DFSTRM00
- | - DFSCST00

## IRLM Procedure

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

- “Deadlock Involving Non-IRLM Resources”
- “Deadlock Involving Only IRLM Resources”
- “Lock Request Not Granted Because Holder Did Not Release Lock”
- “IRLM Latch Unavailable” on page 58

### **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 section details procedures and techniques for the following tasks:
- “Searching the Database” provides information about searching the IBM Software Support Facility (SSF) to find out whether a problem like yours is already known to IBM.
  - “Searching for APARs Closed within a Specific Time Period” on page 60 provides information about searching RETAIN for APARs closed within a specific time period.
  - “Preparing an APAR” on page 61 provides information about preparing an APAR.

### 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 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 is 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 B, “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 9 these are:

<b>AR900</b>	IMS Services
<b>AR901</b>	Database Manager
<b>AR902</b>	Transaction Manager
<b>AR903</b>	ETO
<b>AR904</b>	Recovery Level Tracker
<b>AR905</b>	Database Level Tracker
<b>AR906</b>	Database recovery service
<b>AR907</b>	IMS Connect

**R101** To search all entries for Internal Resource Lock Manager (IRLM) 2.1

**Note:** To search only the APAR entries, use AR101 for IRLM 2.1.

| **R220** To search all entries for Internal Resource Lock Manager (IRLM) 2.2

| **Note:** To search only the APAR entries, use AR220 for IRLM 2.2.

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

<b>LVLS/900</b>	IMS Services
<b>LVLS/901</b>	Database Manager
<b>LVLS/902</b>	Transaction Manager
<b>LVLS/903</b>	ETO
<b>LVLS/904</b>	Recovery-level Tracking
<b>LVLS/905</b>	Database-level Tracking
<b>LVLS/906</b>	Database recovery service
<b>LVLS/907</b>	IMS Connect
<b>LVLS/101</b>	Internal Resource Lock Manager 2.1

| An example is: **565J3800 AR901** for the Database Manager

| For a structured database search, an example is: **PIDS/5655J3800 LVLS/901**

**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

| The following searches refer to the use of RETAIN and are therefore directed at IBM support personnel. RETAIN can be searched 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 10/02 and 04/04, use this search argument:

| P;CL02/10-04/4. 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 9 APARs, use this search argument:

| P;CL02/10-04/4. HIPER 5655J3800

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

| P;CL02/10-04/4. HIPER PIDS/5655J3800

## 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 5. Preparing an APAR

Procedure	What to Do
Reporting a problem	<p>To report a problem, contact your IBM Support Center. Be prepared to supply such information as:</p> <ul style="list-style-type: none"> <li>• Customer number</li> <li>• Release level</li> <li>• Current maintenance level (from PTF list)</li> <li>• The keyword string or strings used to search the IBM software support database</li> </ul>
Gathering APAR documentation	<p>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.</p> <ul style="list-style-type: none"> <li>• JCL listings</li> <li>• 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</li> <li>• Link-edit map</li> <li>• z/OS console printout. A partial console is generally in the offline formatted dump.</li> <li>• Master terminal printout</li> <li>• Local/remote terminal printout</li> <li>• IMS log data sets</li> <li>• IMSGEN listing</li> <li>• DBD listing</li> <li>• PSB listing</li> <li>• ACB generation output</li> <li>• Log trace</li> <li>• Consolidated trace output</li> <li>• Transmittal notes explaining any unusual events leading up to the problem symptoms</li> <li>• SNAPs produced before and after the failing call by DFSDDLTO</li> <li>• Type X'67FF' SNAP log records</li> <li>• Type X'6705' SNAP log records</li> <li>• DBRC—RECON data set</li> <li>• LPA map</li> <li>• LOGREC (especially software diagnostic records)</li> </ul>
Submitting APAR documentation	<p>When submitting material for an APAR to IBM, carefully pack and clearly label all materials sent to IBM with the following information:</p> <ol style="list-style-type: none"> <li>1. The APAR number assigned by IBM</li> <li>2. A list of data sets on the tape, including JCL, if any</li> <li>3. A description of how the tape was made, including: <ul style="list-style-type: none"> <li>• The exact JCL listing or the list of commands used</li> <li>• The recording mode and density</li> <li>• Tape labeling</li> <li>• The record format and block size used for each data set</li> </ul> </li> </ol>





---

## Part 2. Data Areas and Record Formats

<b>Chapter 6. Data Areas and Record Formats</b> . . . . .	65
Getting More Information on Modules, Control Blocks, and Record Formats . . . . .	65
Table of Control Block Definitions . . . . .	67
Control Block Interrelationship Diagrams . . . . .	74
DL/I Record Formats . . . . .	113
HSAM and SHSAM Database . . . . .	113
HISAM and SHISAM Database . . . . .	114
HDAM, HIDAM, PHDAM, or PHIDAM Database . . . . .	115
OSAM and VSAM ESDS Block Format . . . . .	117
I VSAM LRECL for a Primary Index . . . . .	117
Secondary Index or PSINDEX Database (VSAM Only) . . . . .	118
Variable-Length Segments . . . . .	119



---

## Chapter 6. Data Areas and Record Formats

This section 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 section includes:

- “Getting More Information on Modules, Control Blocks, and Record Formats”
- “Table of Control Block Definitions” on page 67
- “Control Block Interrelationship Diagrams” on page 74
- “DL/I Record Formats” on page 113

---

### 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, DFSUACB0, and the Write-PSBs-and-DMBs-to-ACBLIB module, DFSUAMB0.

Figure 1 on page 66 gives an overview of the linkage of the major control blocks used for diagnosis.

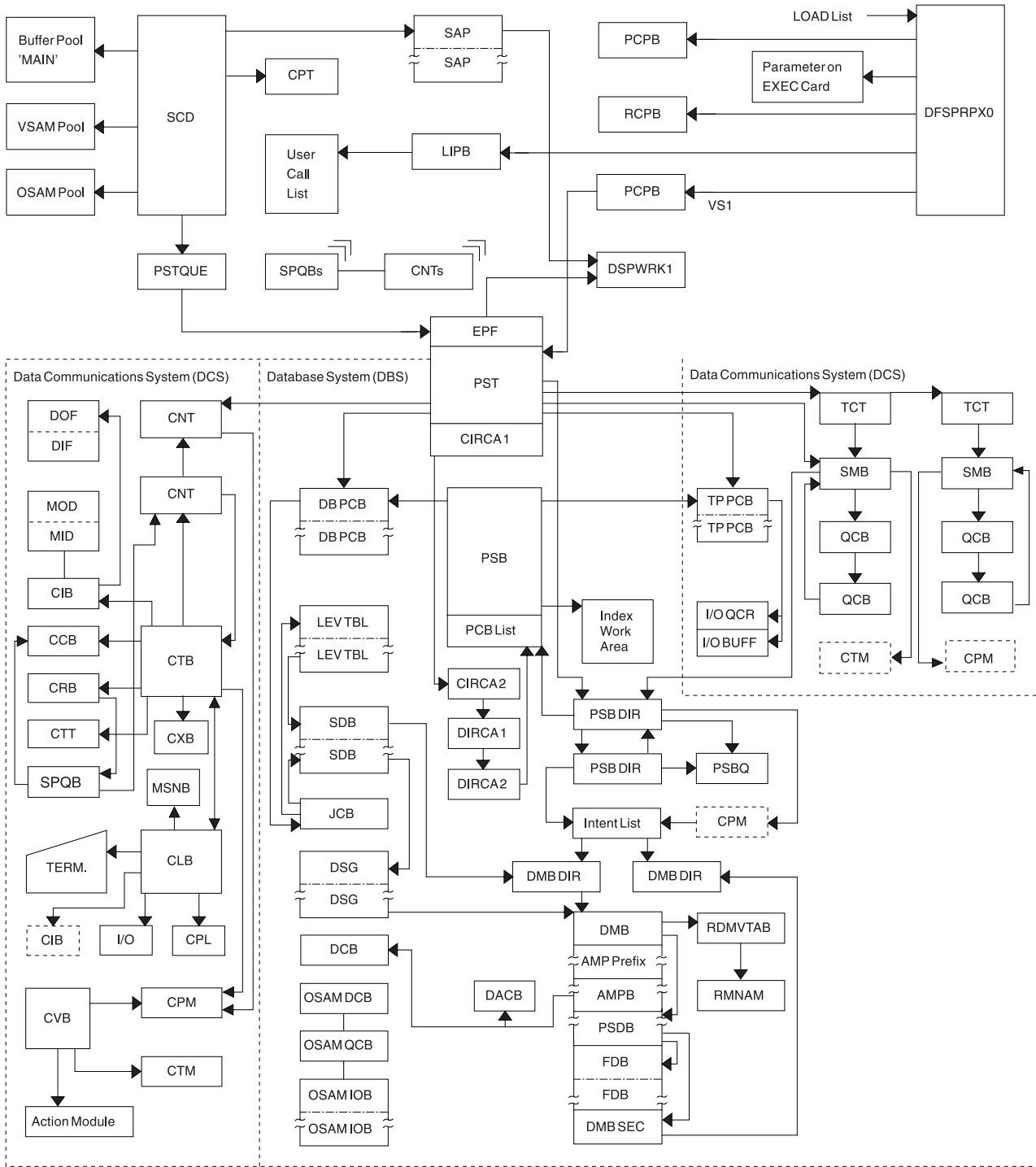


Figure 1. IMS Control Block Linkage for a Static DB/DC Environment

## Table of Control Block Definitions

Table 6 lists:

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

Table 6. 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	IDL DMBASE=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 DFSPPOOL 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 DFSPPOOL 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 CCBASE=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	DFSCLE	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.
CSV	DFSCSVT	Callable Services Vector Table. Used by IMS callable services modules.

Table 6. Table of Control Block Definitions (continued)

Control Block Acronym	Mapping Macro	Description
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).
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	IDCBOSD	Data communication block. Contains data pertinent to the current use of a data set.
DCB-EXT	DFSDCBEX	OSAM extension to the 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	DFSDPBFH	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 DSGBASE=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	z/OS 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)

Table 6. Table of Control Block Definitions (continued)

<b>Control Block Acronym</b>	<b>Mapping Macro</b>	<b>Description</b>
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.
EPST	DBFEPT	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 DFSISMNO 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 6. 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 IDSC 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	DXRRISL	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.
I LEV	IDLI LEVBASE=0	Level table. Consists of two parts: previous call and current call that is filled in by the call analyzer.
I LIPARMS	PARMBLK	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.
PCIB	ICLI PCIBASE=0	Partition communication interface block.
I PCPARMS	PARMBLK	Program control parameter block.
PCT	DFSPCT	Partition chaining table.
I PDAE	DFSPSEIB	Partition Definition Area Prefix. Partition Definition Area Entry.



Table 6. Table of Control Block Definitions (continued)

<b>Control Block Acronym</b>	<b>Mapping Macro</b>	<b>Description</b>
PDIR	IDLI PDRBASE=0	Program specification block directory. Contains entries for every program known to IMS.
PDL	DFSPDL	DRA dump parameter list.
PECA	DFSPSEIB	Partition Exit Communication Area.
PNT	DFSPNT	Partition Name Table.
POOLHDR	DFSSPBLK	Storage pool header used by the DFSPPOOL 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	PARMBLK	Parameter Anchor Block.
QCB	IAPS SMBBASE=0	Queue control block.
QEL	IAPS SMBBASE=0	Queue Element.
QMBA	DFSQMGR	Queue Manager Buffer Area.
RCPARMS	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	DXRRPL	IRLM request parameter list. This is the parameter list for all functional requests for the resource lock manager.
RLQD	DXRRQD	IRLM query mapping macro. Maps IRLM control blocks/structures returned to the IMS invoker of QUERY.
RPL	IDLIVSAM	Request parameter list. Contains parameters passed to VSAM from IMS and the status returned to IMS from VSAM.

Table 6. Table of Control Block Definitions (continued)

Control Block Acronym	Mapping Macro	Description
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 SDCB control blocks and to serialize the sequential buffer SDCB and SDSG control block subsystem chains. The SBHEs are part of the SBSCD.
SBPARMS	DFSSBPAP	Sequential buffering extension to PXPAPMS.
SBPSS	DFSSBPSS	Small section of the SBPST that needs to be in CSA.
SBPST	DFSSBPST	Sequential buffering extension to the PST.
SBSCD	DFSSBSCD	Sequential buffering extension to the SCD. This extension contains the SBHE hash entries.
SBUE	DFSSBBUF	Sequential buffering buffer extension. There is one SBUE for each SBUF.
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 SBPAPM control statement in the //DFSCTL file.
SCA1	DFSSBCAR	Control block containing the uninterpreted data of one SBPAPM 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.

Table 6. Table of Control Block Definitions (continued)

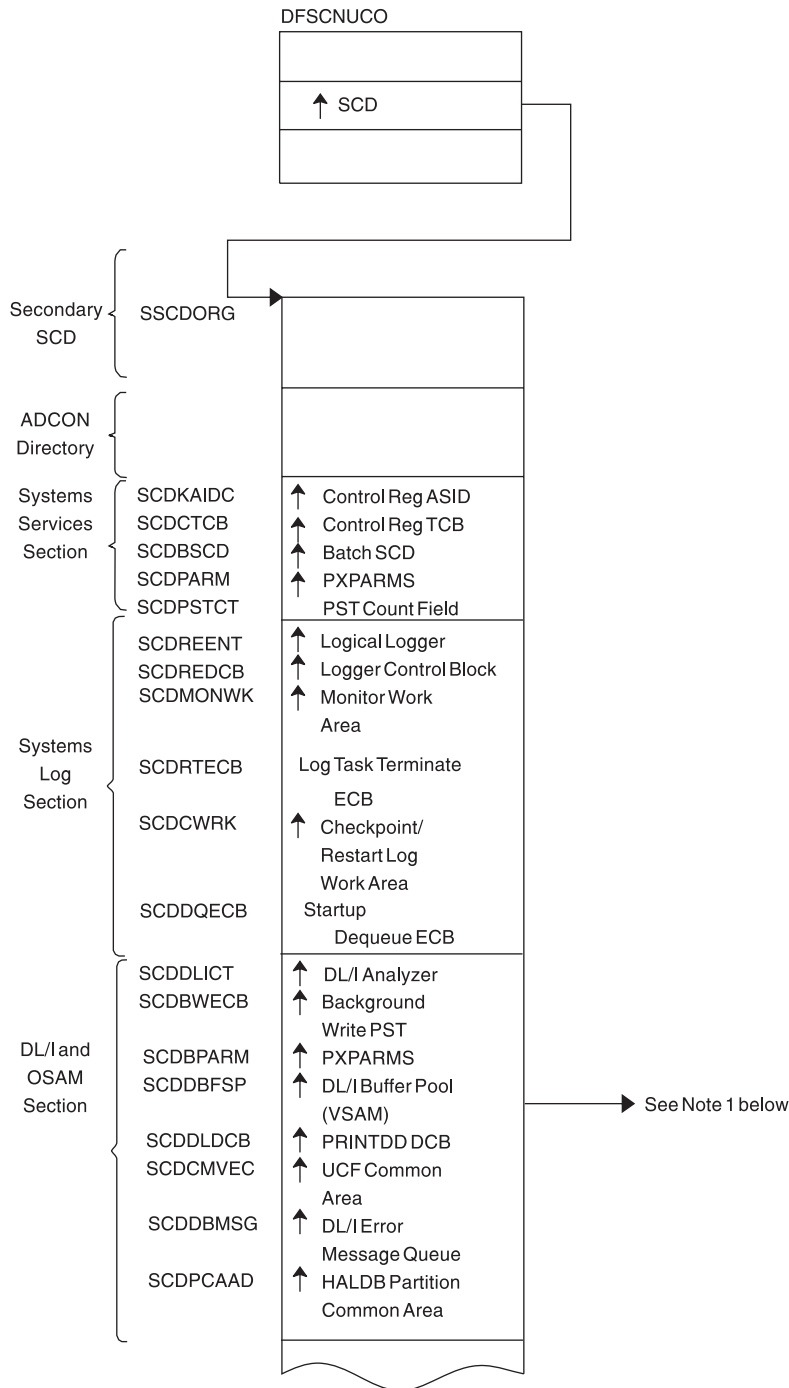
<b>Control Block Acronym</b>	<b>Mapping Macro</b>	<b>Description</b>
SPQB	ICLI SPQBASE=0	Subpool queue block. The SPQB represents the dynamic user for an ETO terminal and represents a set of static queues (CNTs) for a static ISC parallel session terminal.
SQPST	ISQPST	PST queue. Associated with the scheduler sequence queue.
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 z/OS 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.
TPPCB	IDLI TPCBASE=0	Program communication block. There is one for each logical database being referenced by the application program.
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 DFSISMNO Storage Manager to keep track of a buffer obtained using ICREATE.

## Control Block Interrelationship Diagrams

This topic contains diagrams that show the interrelationships of major control blocks in an IMS environment. Descriptions of the figures in this topic are listed in Table 7.

*Table 7. Description of Control Block Interrelationship Diagrams*

<b>Figure</b>	<b>Description</b>
Figure 2 on page 75	Online system contents directory (SCD)
Figure 3 on page 81	DFSPRPX0 parameter blocks
Figure 4 on page 82	OSAM buffer pool
Figure 5 on page 83	Sequential buffering control blocks
Figure 6 on page 84	VSAM buffer handler pool
Figure 7 on page 85	OSAM DECB with IOB in use
Figure 8 on page 86	OSAM IOB pool showing available IOBs
Figure 9 on page 87	Storage management control block relationships created by the ICREATE facility
Figure 10 on page 88	Storage management control block relationships for preallocated storage blocks
Figure 11 on page 90	Storage management control block relationships for DFSPPOOL pools
Figure 12 on page 91	Storage management control block relationships for DFSCBT00 pools
Figure 13 on page 92	
Figure 14 on page 94	Database control blocks
Figure 15 on page 96	Diagram of a data management block (DMB)
Figure 16 on page 97	Fast Path control block overview
Figure 17 on page 98	Relationships between buffer control blocks for Fast Path databases
Figure 18 on page 99	GSAM control block overview
Figure 19 on page 100	GSAM control blocks
Figure 20 on page 101	Relationships between DL/I control blocks
Figure 21 on page 102	IMS Transaction Manager control blocks
Figure 22 on page 102	Intersystem communication control block structure
Figure 23 on page 103	VTCB load module
Figure 24 on page 105	Multiple systems coupling (MSC) control block overview
Figure 25 on page 106	Multiple systems coupling (MSC) main storage-to-main storage control block overview
Figure 26 on page 107	z/OS storage map of interrelationships of IMS to IRLM
Figure 27 on page 108	IRLM overall control block structure
Figure 28 on page 109	IRLM storage manager pools
Figure 29 on page 109	IRLM lock request examples
Figure 30 on page 110	Control block overview of database recovery control (DBRC)
Figure 31 on page 111	Organization and basic linkages for DOF (device output format) and MOD (message output descriptor)
Figure 32 on page 112	Organization and basic linkages for DIF (device input format) and MID (message input descriptor)

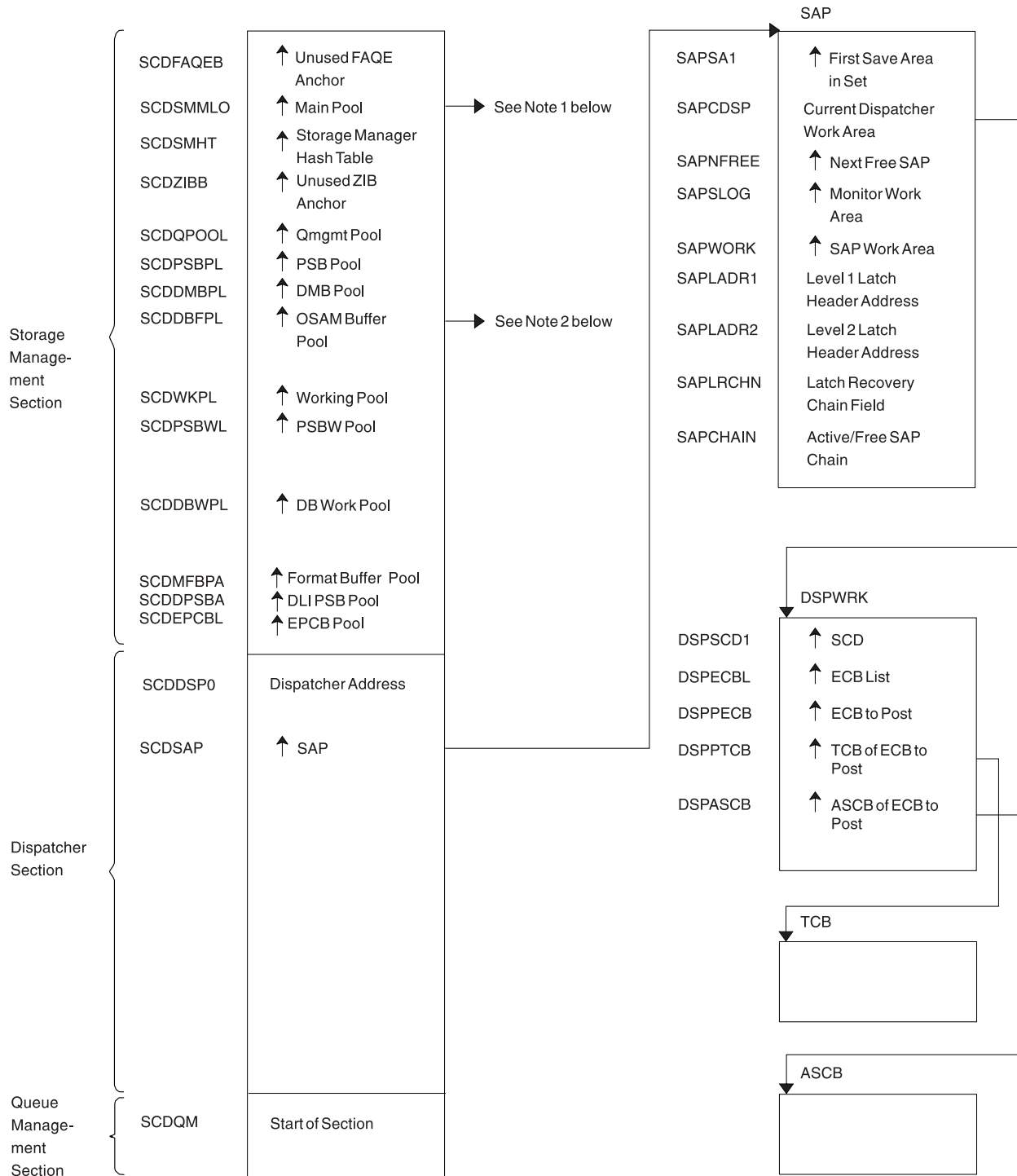


- Note 1: See Figure 4 on page 82. and Figure 6 on page 84.

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

Sequential Buffering Section	SCDSBPTR	▲ SB SCD
Data Sharing Section	SCDIRPM SCDRDSH0	▲ IRLM Parms ▲ DFSRDSH0 (ASYNCRoutine) Data Sharing
	SCDPCCC0	▲ DFSPCCC0 (IRLM/DBRC Handler)
Common Services Section	SCDQHRS SCDCIR00	▲ Queue Header Table Address ▲ Create ITASK Module
	SCDFMOD0	▲ Entry Point of Attach ITASK
STAE/ESTAE Section	SCDXSTA0	A(ESTAE)
Latch/Lock Section	SCDLRSAP	▲ Latch Recovery ITASK SAP
	SCDLMGRA	▲ Latch Manager Address
Formatted Dump Section	SCDDSDWA	▲ SDWAat Dump Time
Timer Services Section	SCDCKVAL	Clock Value
	SCDTIMEP	▲ Timer Services Module (DFSFTIM0)
Trace Services Section	SCDTRBLK SCDPITME	▲ Trace Control Block PITRACE Buffer
External Subsystem Section	SCDESETP	▲ ESET Prefix
Dynamic Control Block Builder Section	SCDCBTA	▲ Control Block Extension Address
	SCDBC00	▲ Address of Control Block Build

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



- Note 1: See Figure 9 on page 87.
- Note 2: See Figure 4 on page 82.

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

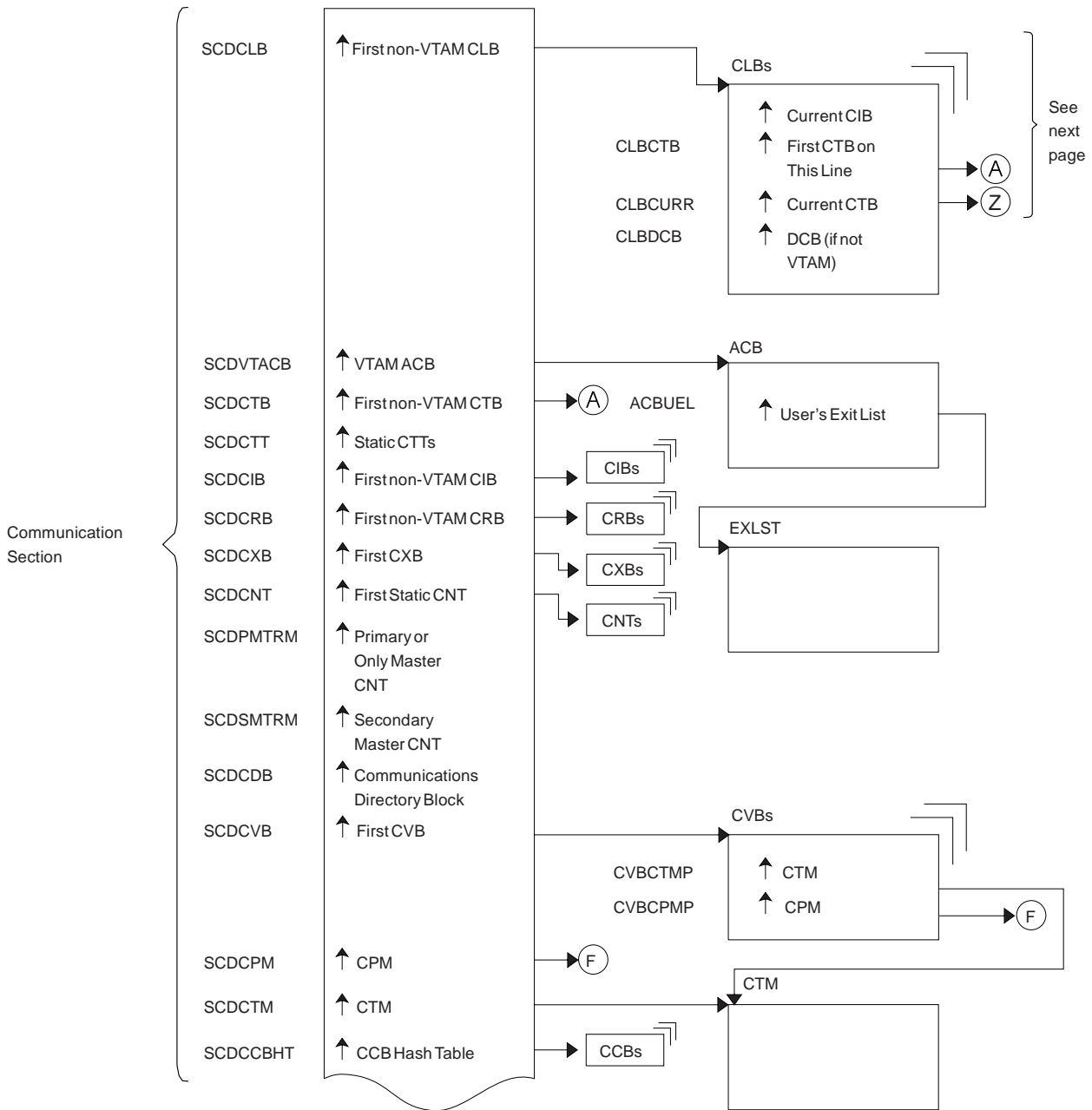


Figure 2. Online System Contents Directory (SCD) (Part 4 of 6)



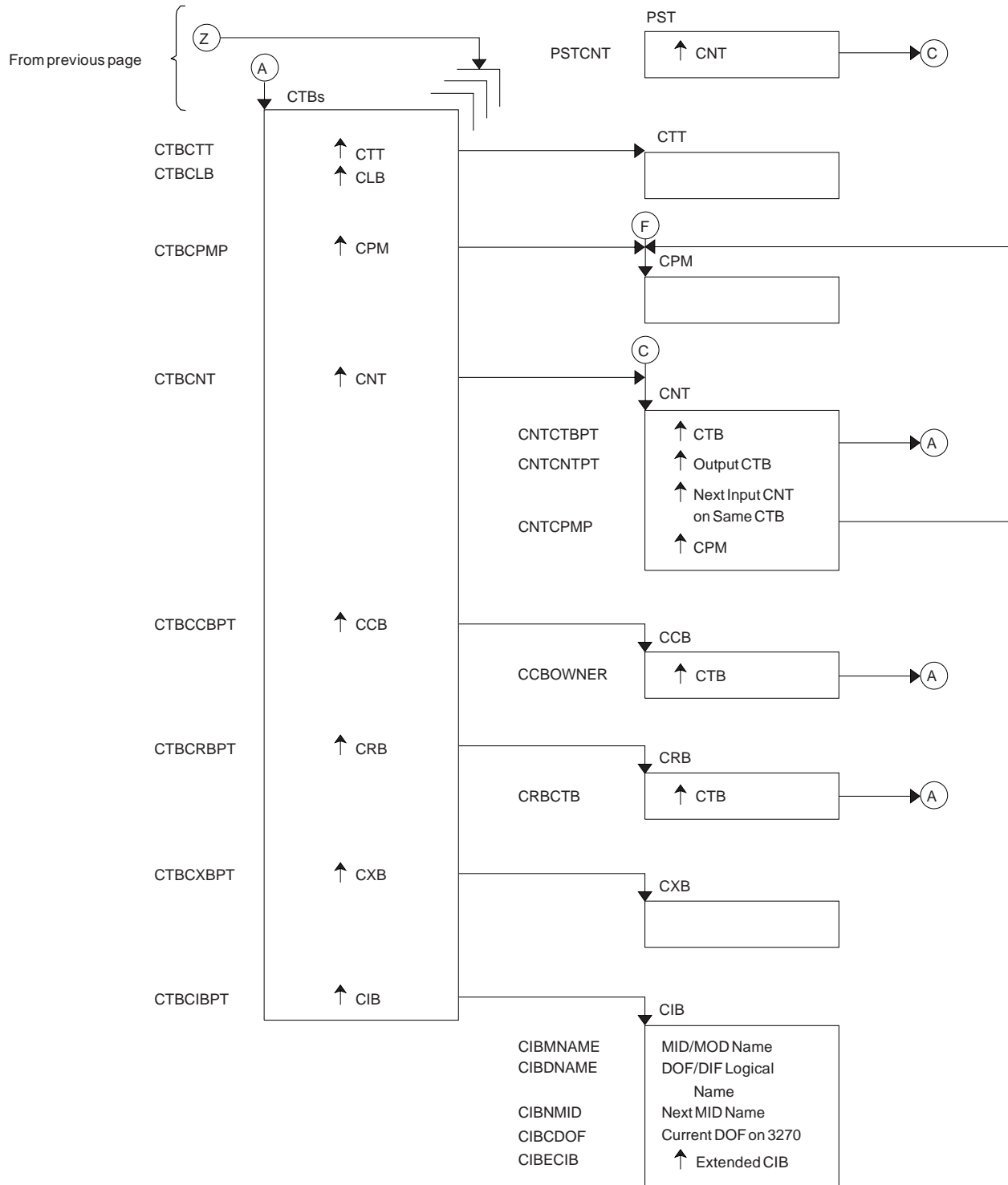


Figure 2. Online System Contents Directory (SCD) (Part 5 of 6)

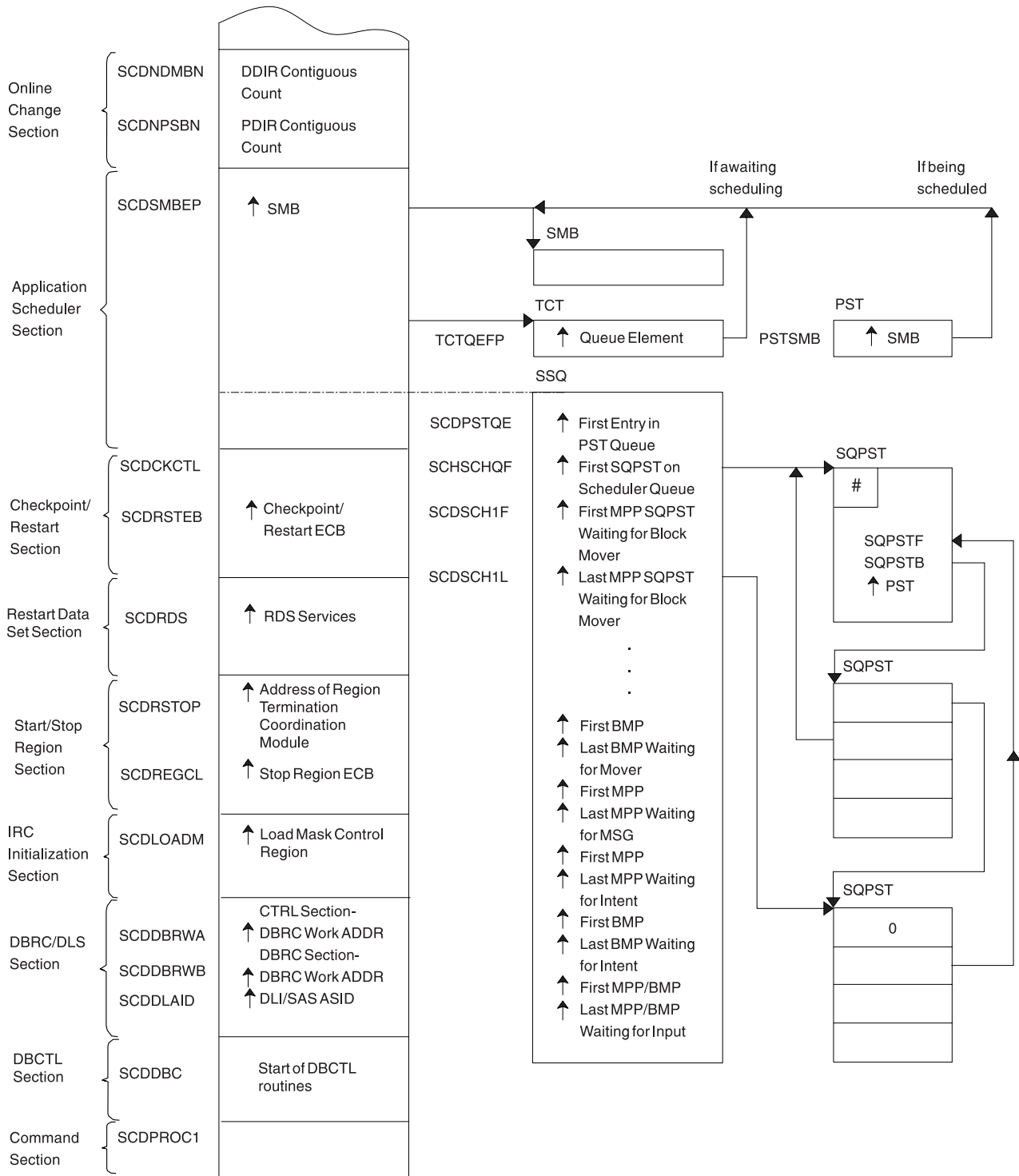
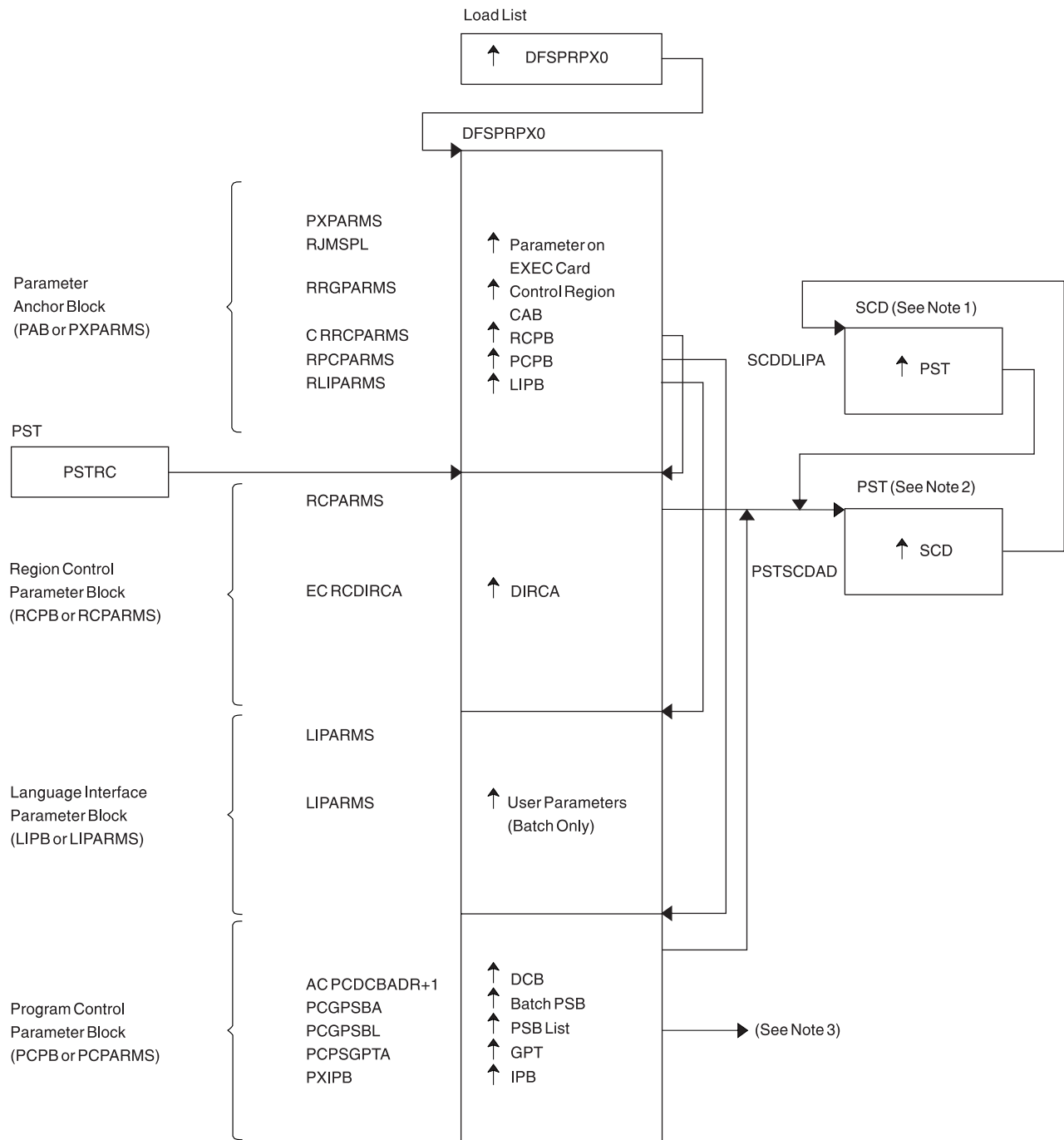


Figure 2. Online System Contents Directory (SCD) (Part 6 of 6)



- Note 1: See Figure 2 on page 75
- Note 2: See Figure 14 on page 94
- Note 3: See Figure 18 on page 99

Figure 3. DFSPRPX0–Parameter Blocks

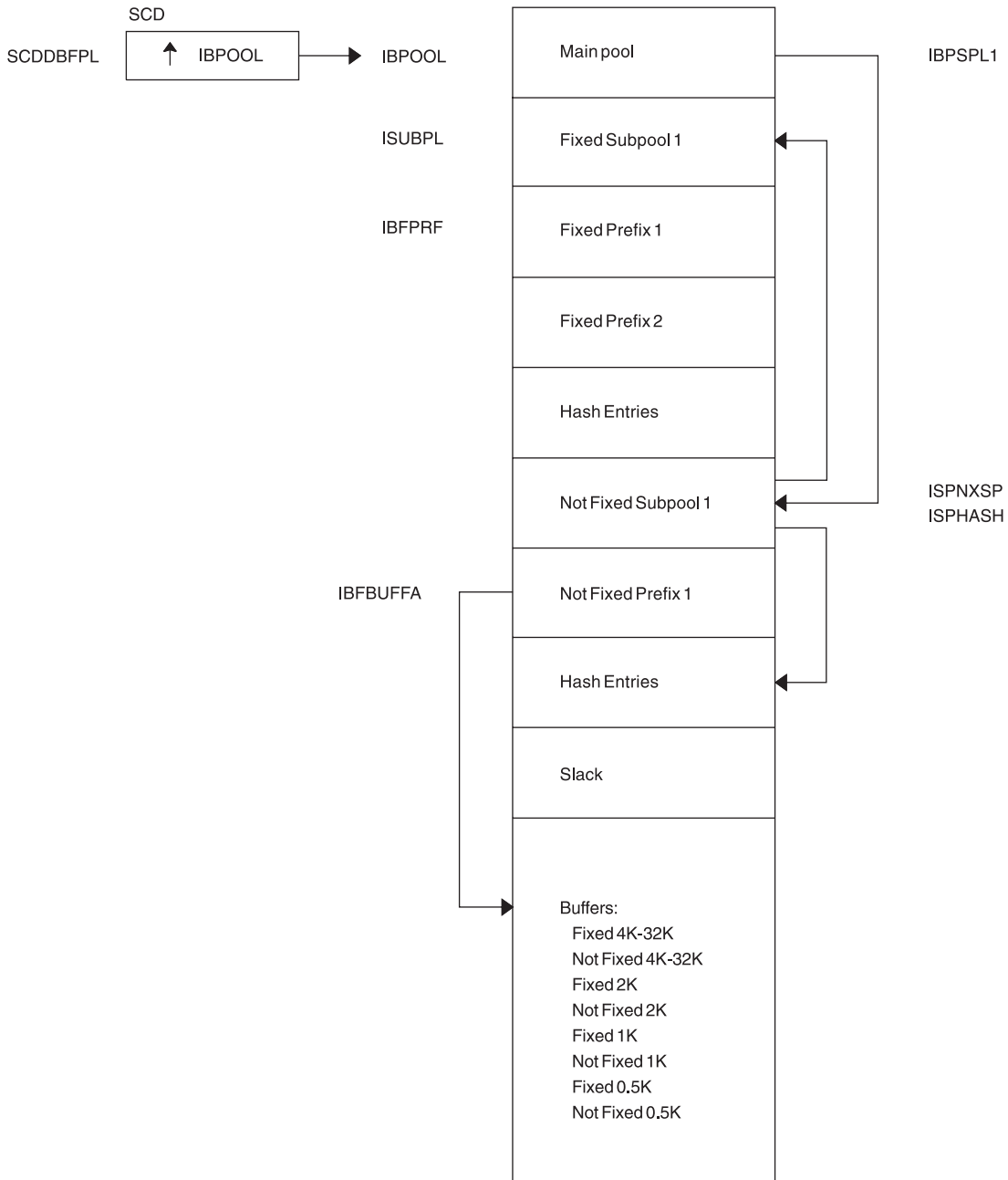


Figure 4. DL/I OSAM Buffer Pool

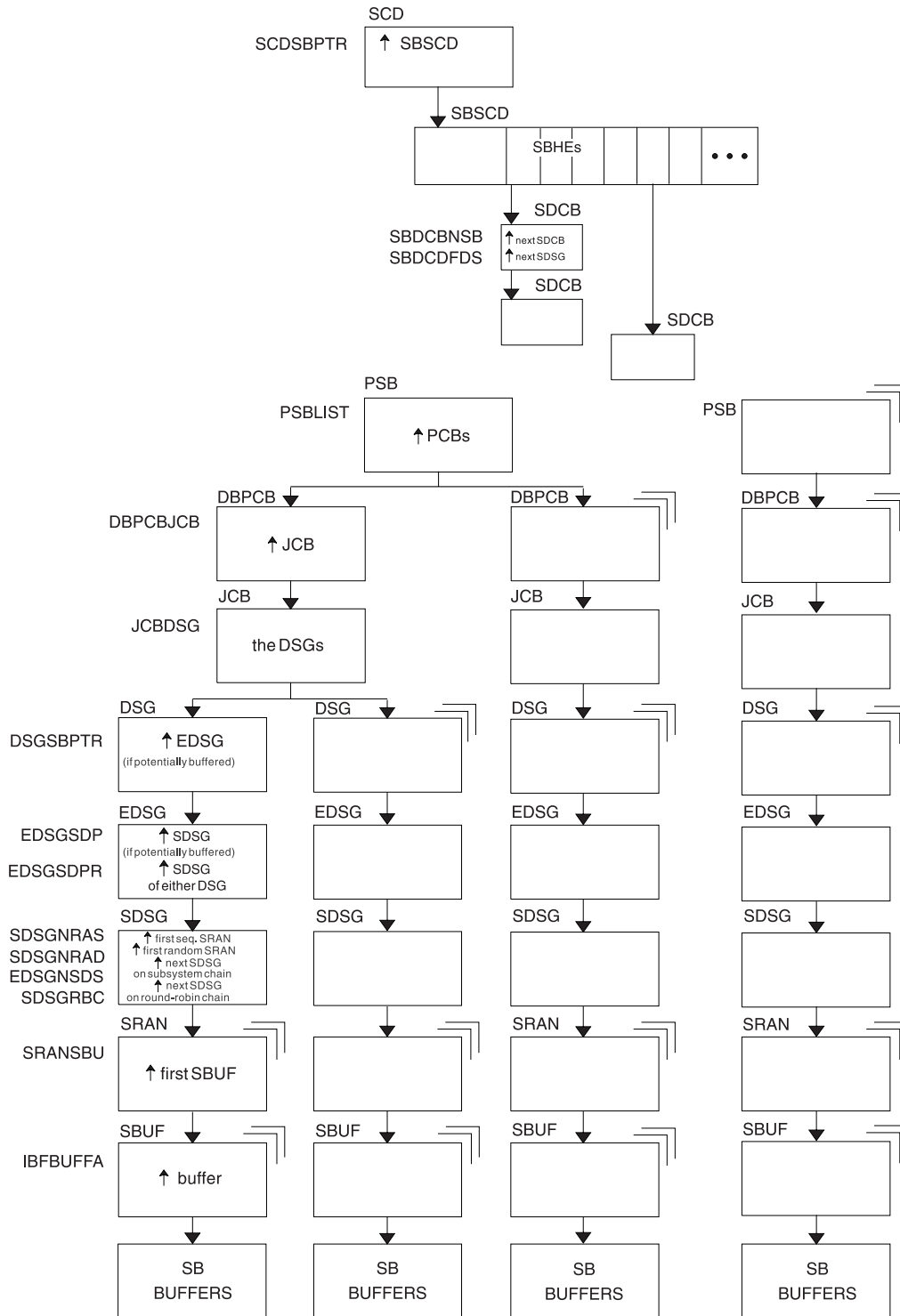


Figure 5. Sequential Buffering Control Blocks

**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 hash entries located within the SBSCD (sequential buffering extension to the systems content directory). IMS uses SBHEs to:

- Anchor the sequential buffering extension to the DCB (SDCB)
  - 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. There must be a separate SDCB for each SBPST that references a HALDB partition, because information in the SDSG will change as the DL/I calls go from partition to partition. As a result, multiple SBPSTs cannot share an SDCB, as is possible for non-HALDB databases. For HALDB, there is one SDCB for each partition used by a PST. 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.

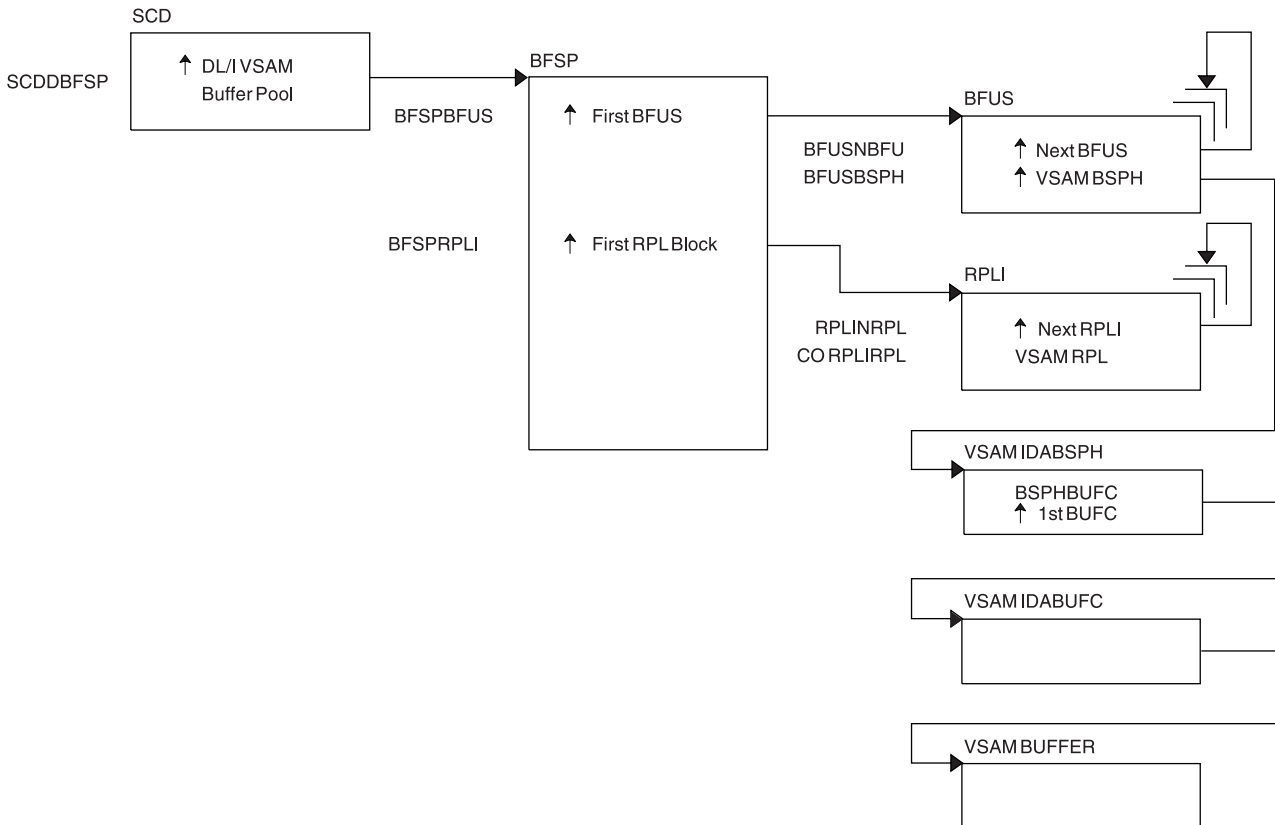


Figure 6. Buffer Handler Pool (VSAM)

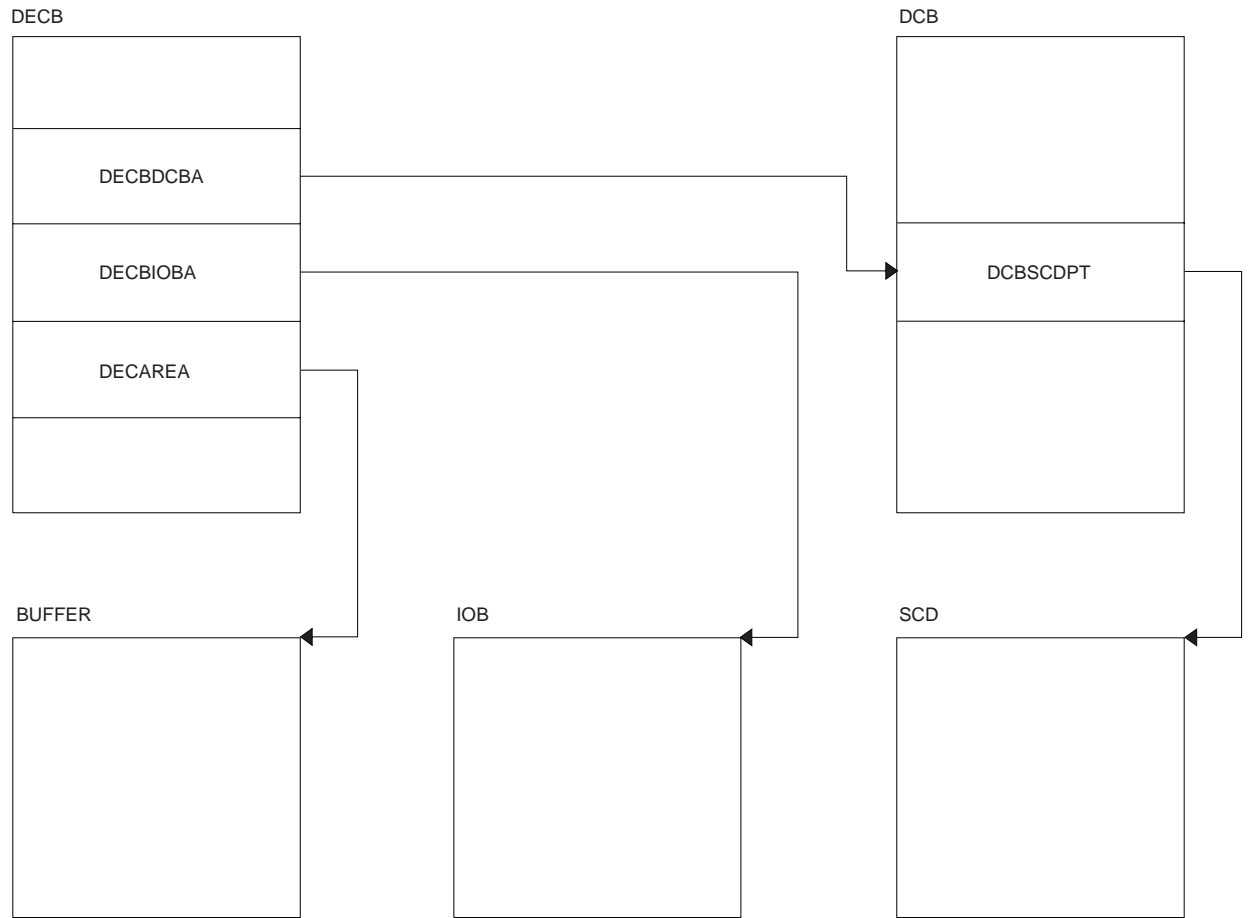


Figure 7. OSAM DECB with IOB in Use

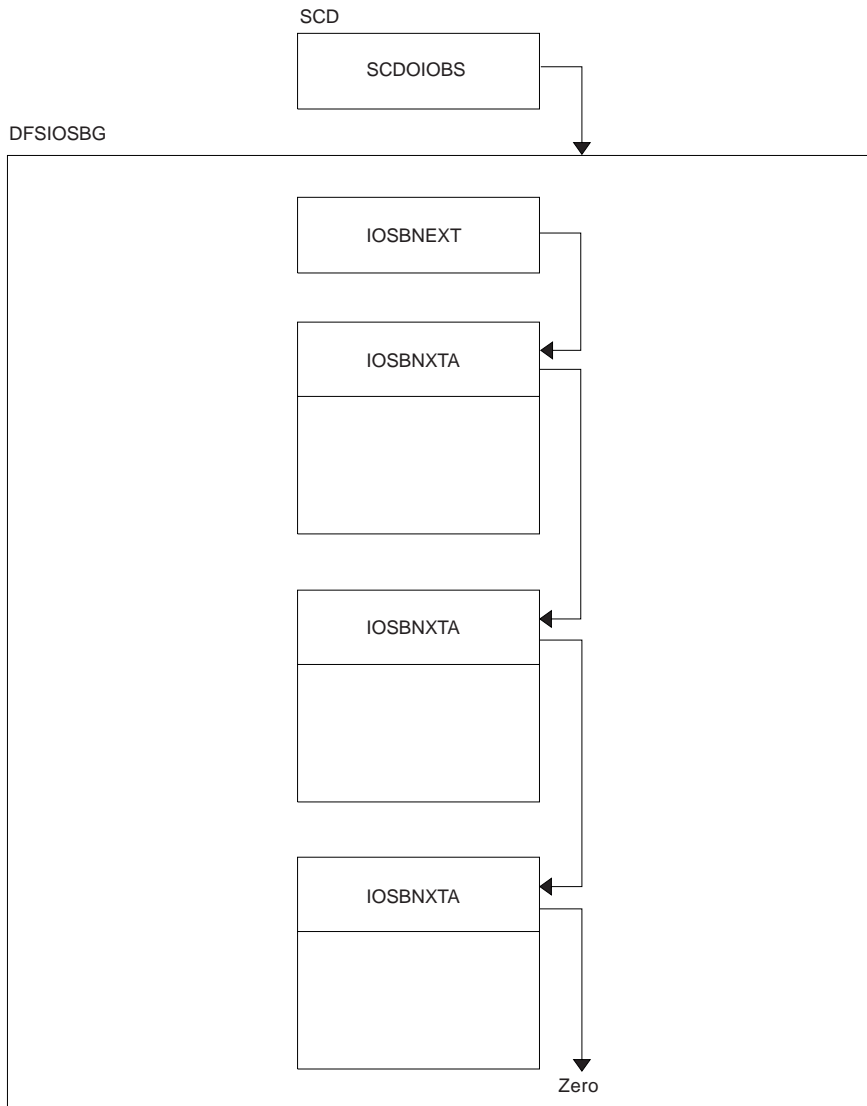


Figure 8. OSAM IOB Pool Showing Available IOBs

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 on page 87.



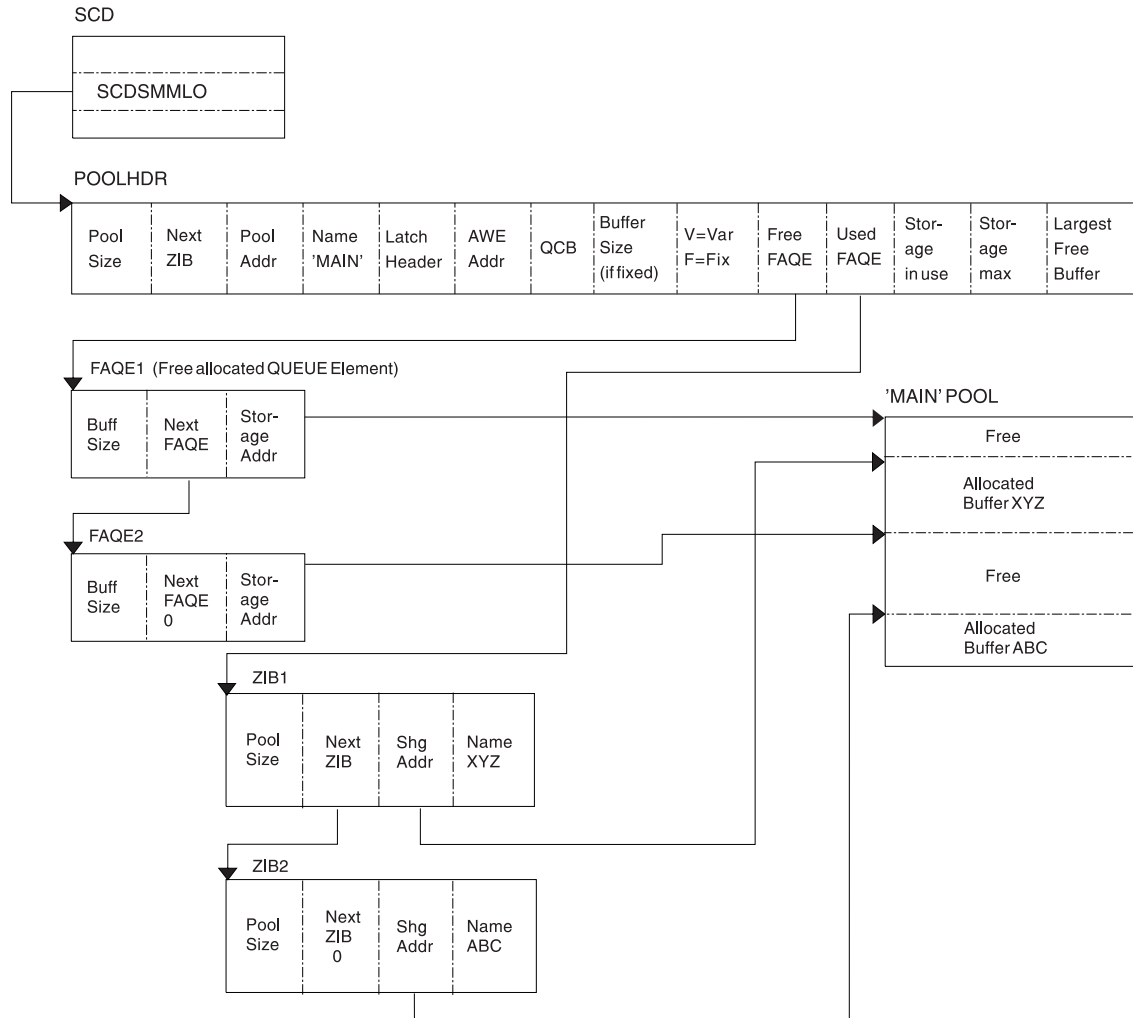


Figure 9. Storage Management Control Block Relationships Created for the MAIN Pool

Figure 10 on page 88 shows the control block relationships for those pools managed by the DFSISMN0 Storage Manager.

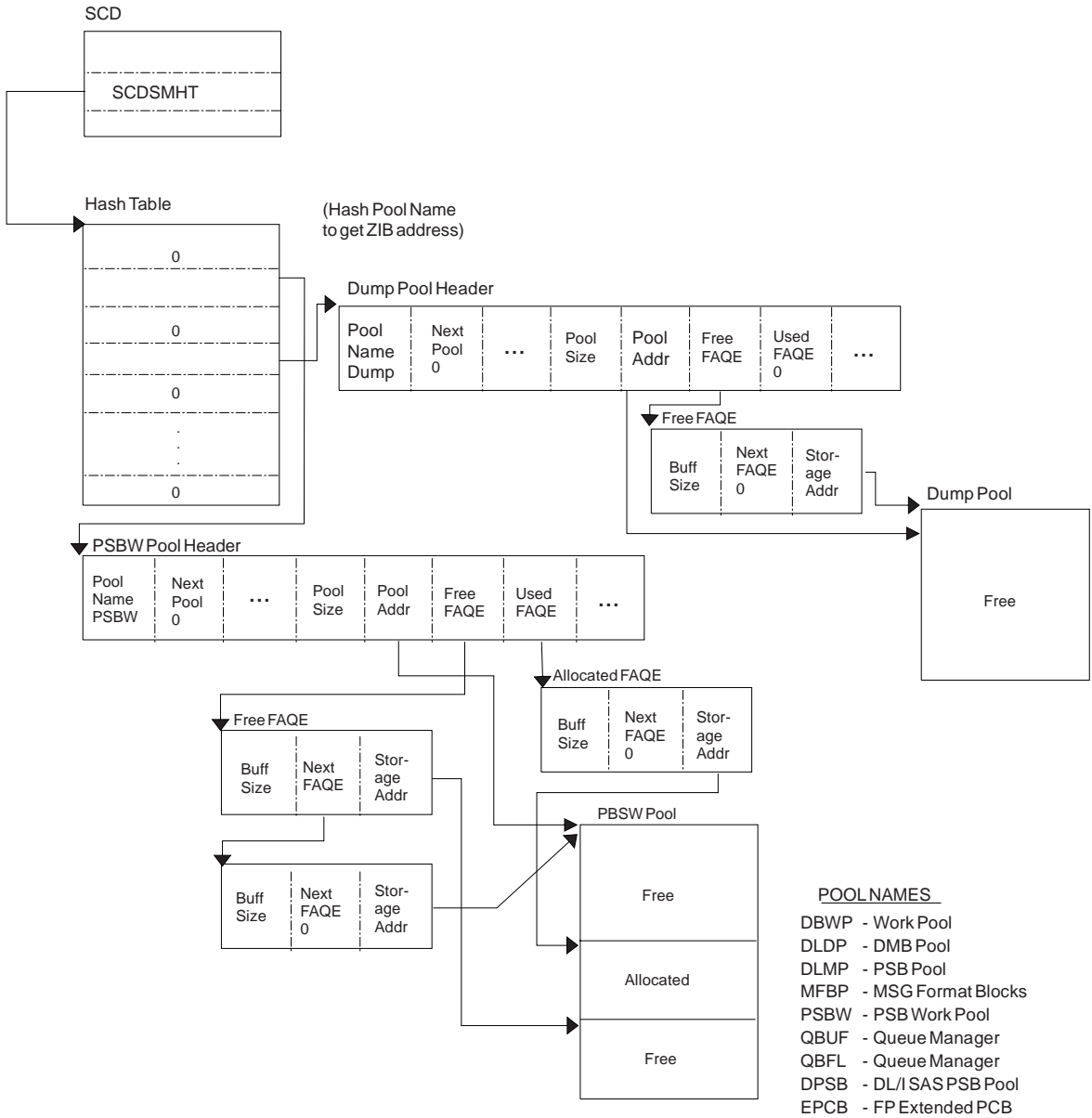


Figure 10. Storage Management Control Block Relationships for Preallocated Storage Blocks

Figure 11 on page 90 shows the control block relationship for pools managed by the DFSPPOOL 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 8-byte prefix and an 8-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 is 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 8-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 8-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.

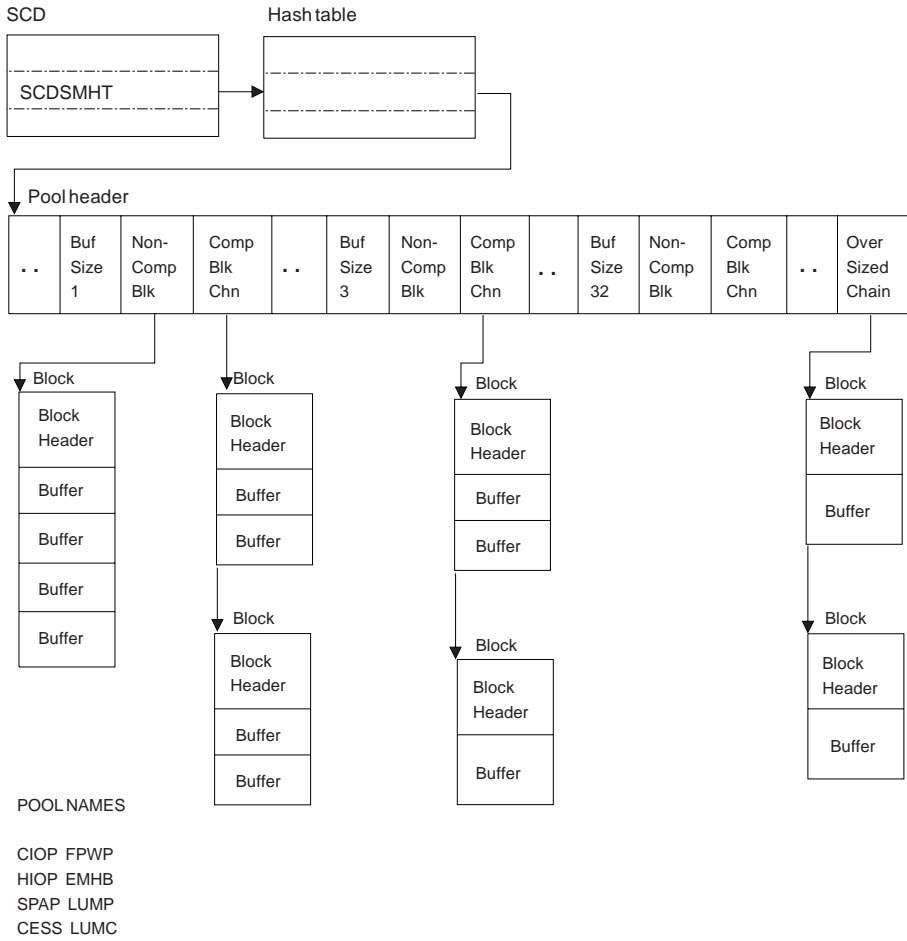


Figure 11. Storage Management Control Block Relationships (DFSPool Pools)

Figure 12 on page 91 shows the Storage Management (DFSCBT00 Pools) control blocks relationships.

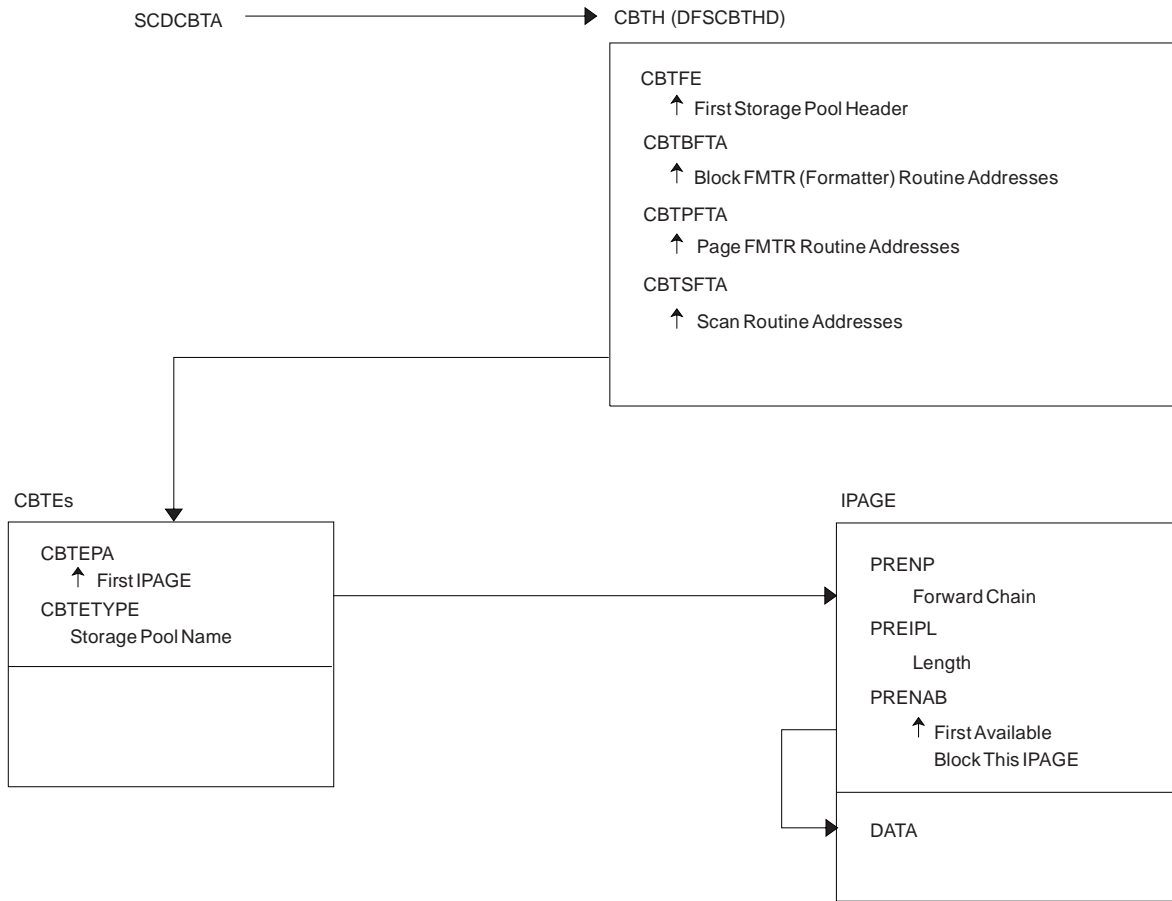


Figure 12. Storage Management Control Block Relationships (DFSCBT00 Pools)

Figure 13 on page 92 shows the Database Manager control blocks for a representative database.

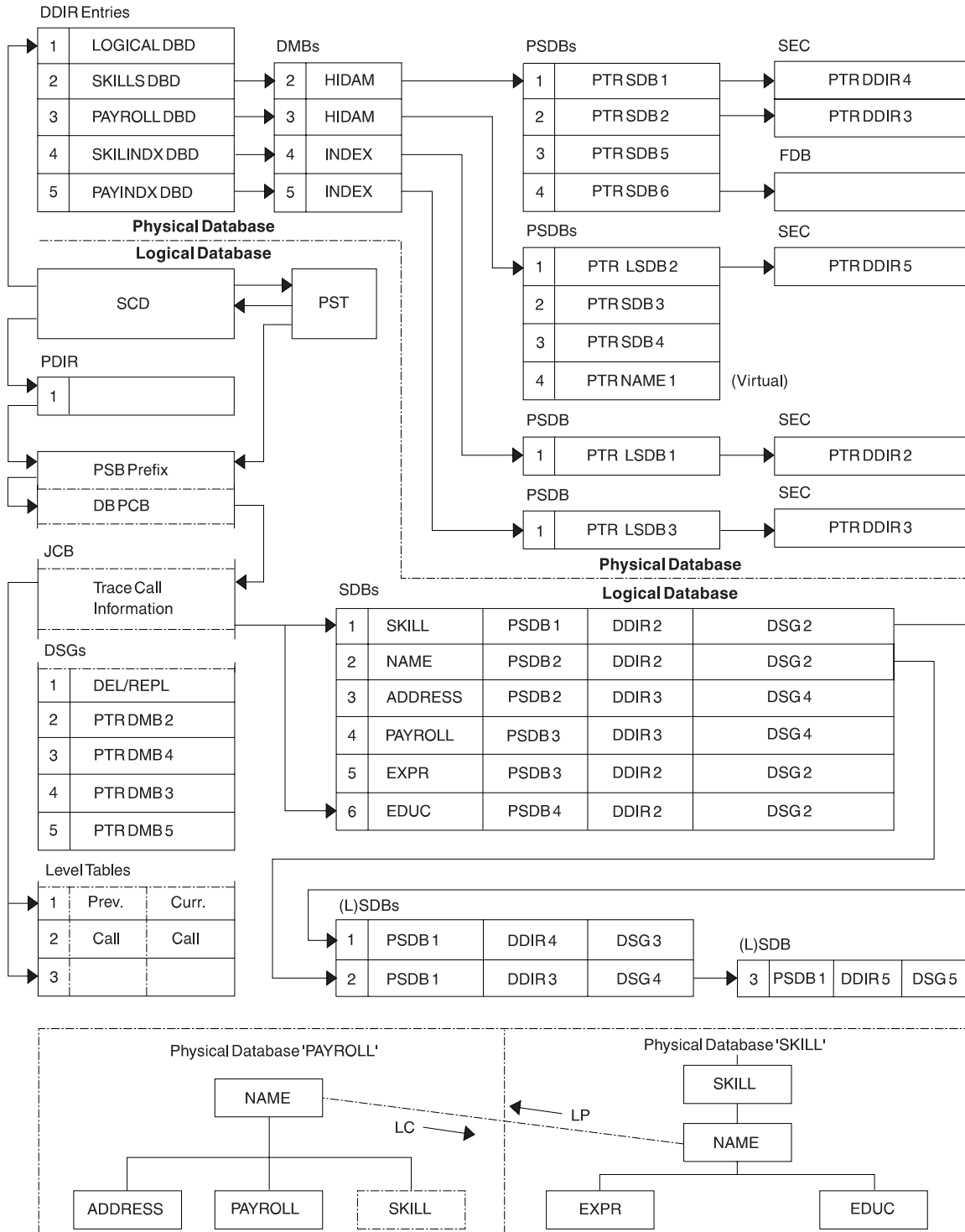


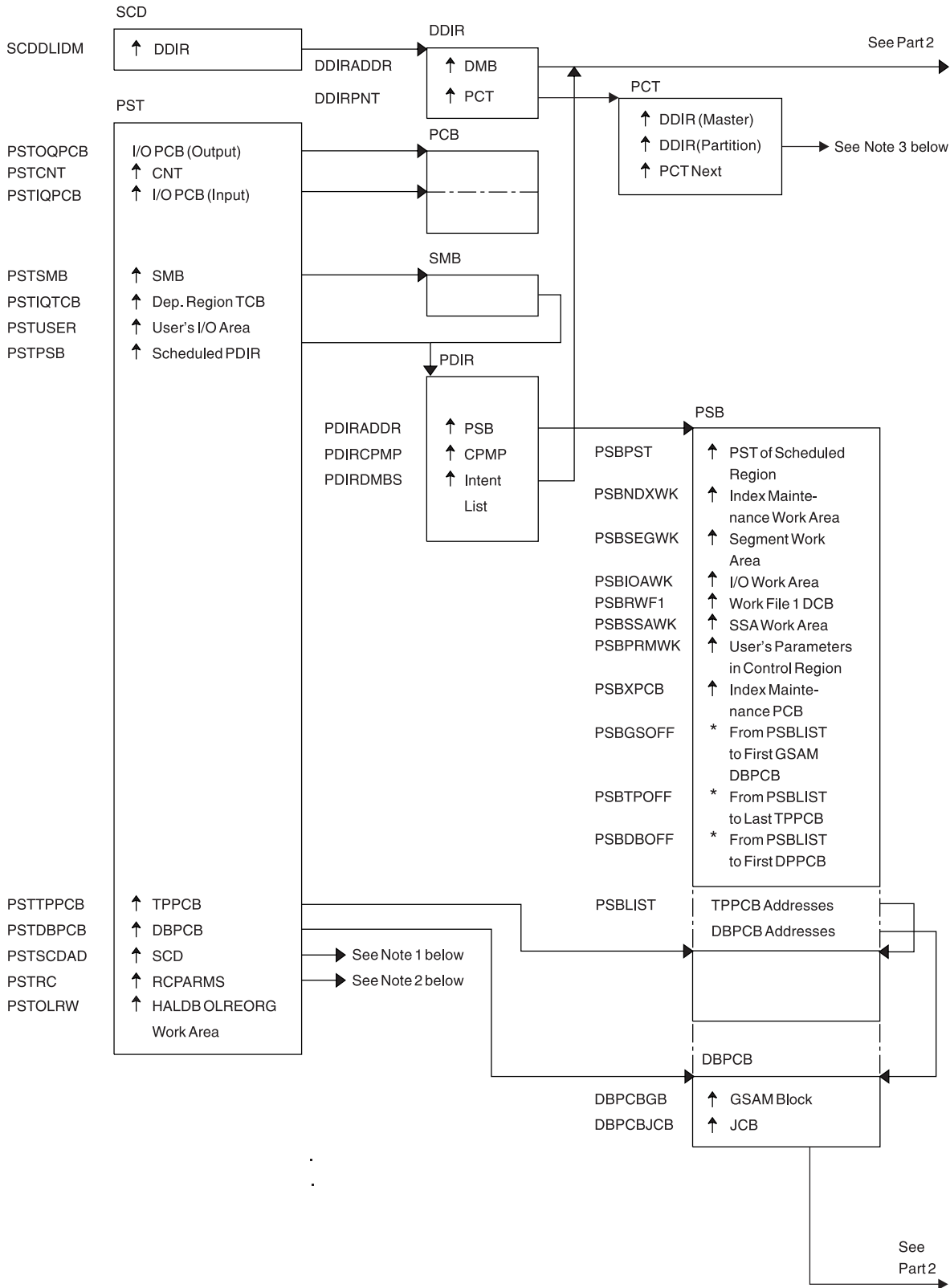
Figure 13. Database Manager Control Blocks for a Representative Database

**Note the following HALDB differences for Figure 13:**

- The SDBs pointer to the DDIR always points to the HALDB Master’s DDIR.
- The PSDBs are under the HALDB master DMB in the DMB pool. The partition DMBs do not contain PSDBs.
- There is no separately defined DDIR or DMB for the primary INDEX database of a PHIDAM. Instead there is an additional AMP in the partition DMB for the primary index.

- There is an ILE DSG for the ILDS which follows the Delete/Replace DSG.

Figure 14 on page 94 shows the relationships between database control blocks.



**Notes:**

1. See Figure 2 on page 75.
2. See Figure 3 on page 81.
3. This is a unique HALDB control block. This control block points the partition DDIR to each other and points the partition DDIR to the master DDIR.

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



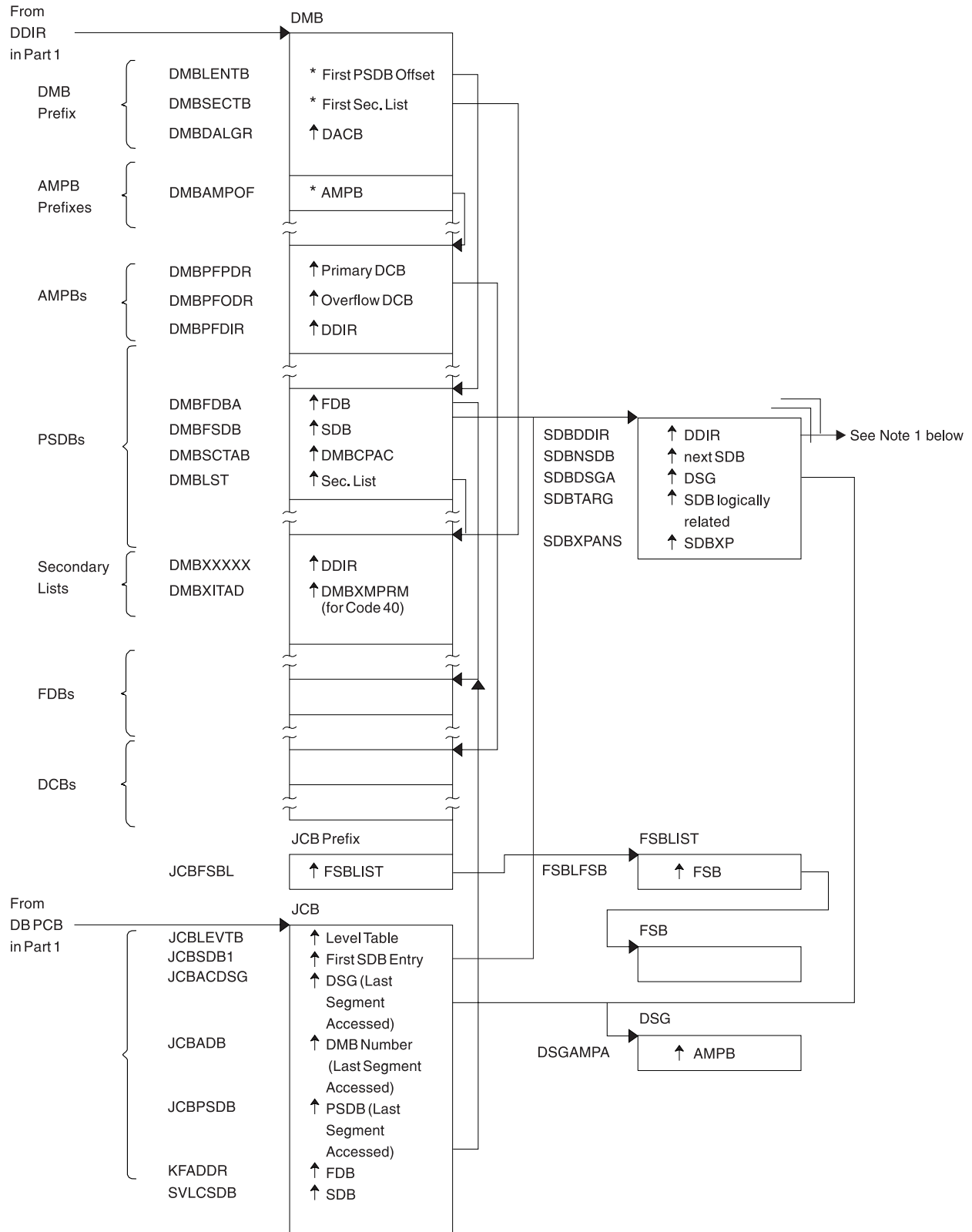


Figure 14. Database Control Blocks (Part 2 of 2)

I Notes to Figure 14 on page 94:

- 1. For HALDB, the SDB points to the Master DDIR.

Figure 15 shows a diagram of a Data Management Block (DMB).

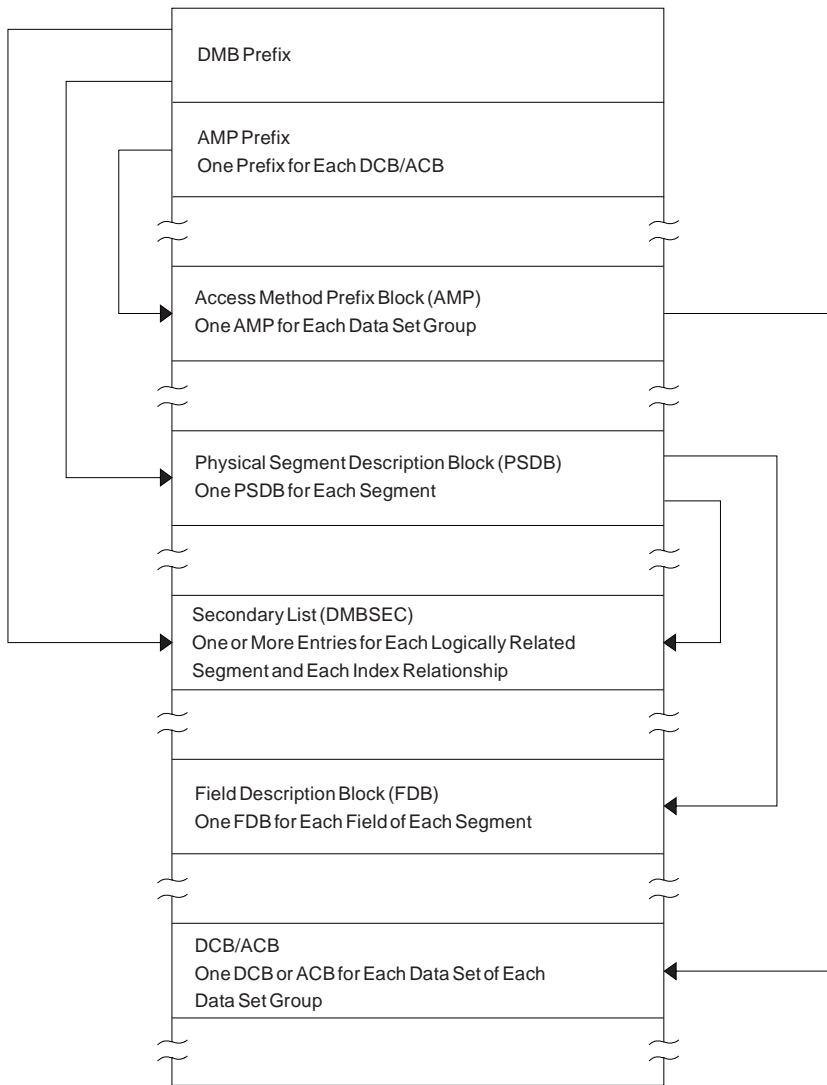


Figure 15. Diagram of a Data Management Block (DMB)

- Note to Figure 15:** For a HALDB, dual DMBs exist in storage. When HALDB Online Reorganization is not in progress, one DMB is active and the other inactive. When HALDB Online Reorganization is in progress, both DMBs are active, with one DMB representing the input data sets, and one DMB representing the output data sets.

Figure 16 on page 97 shows an overview of Fast Path Control Blocks.

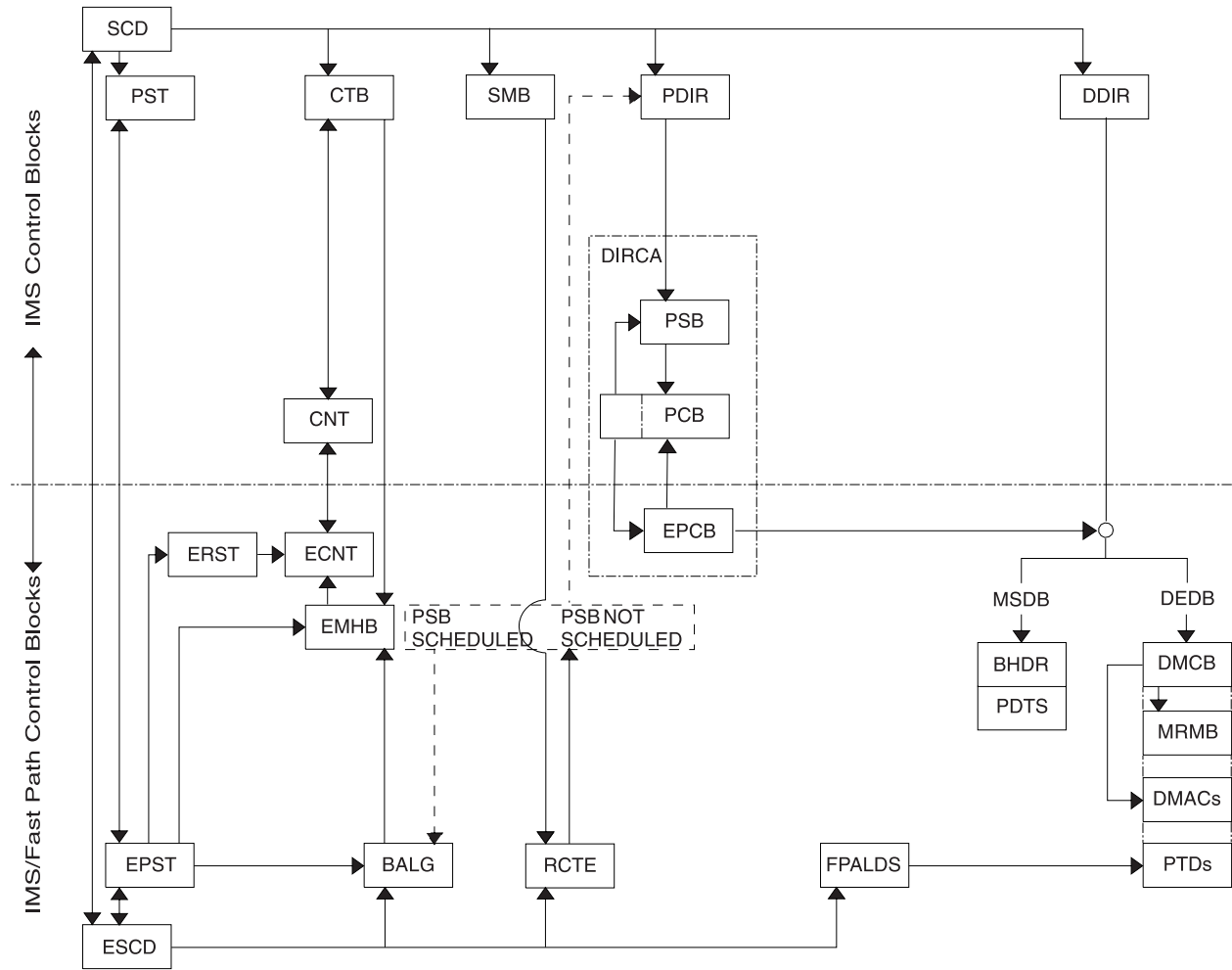
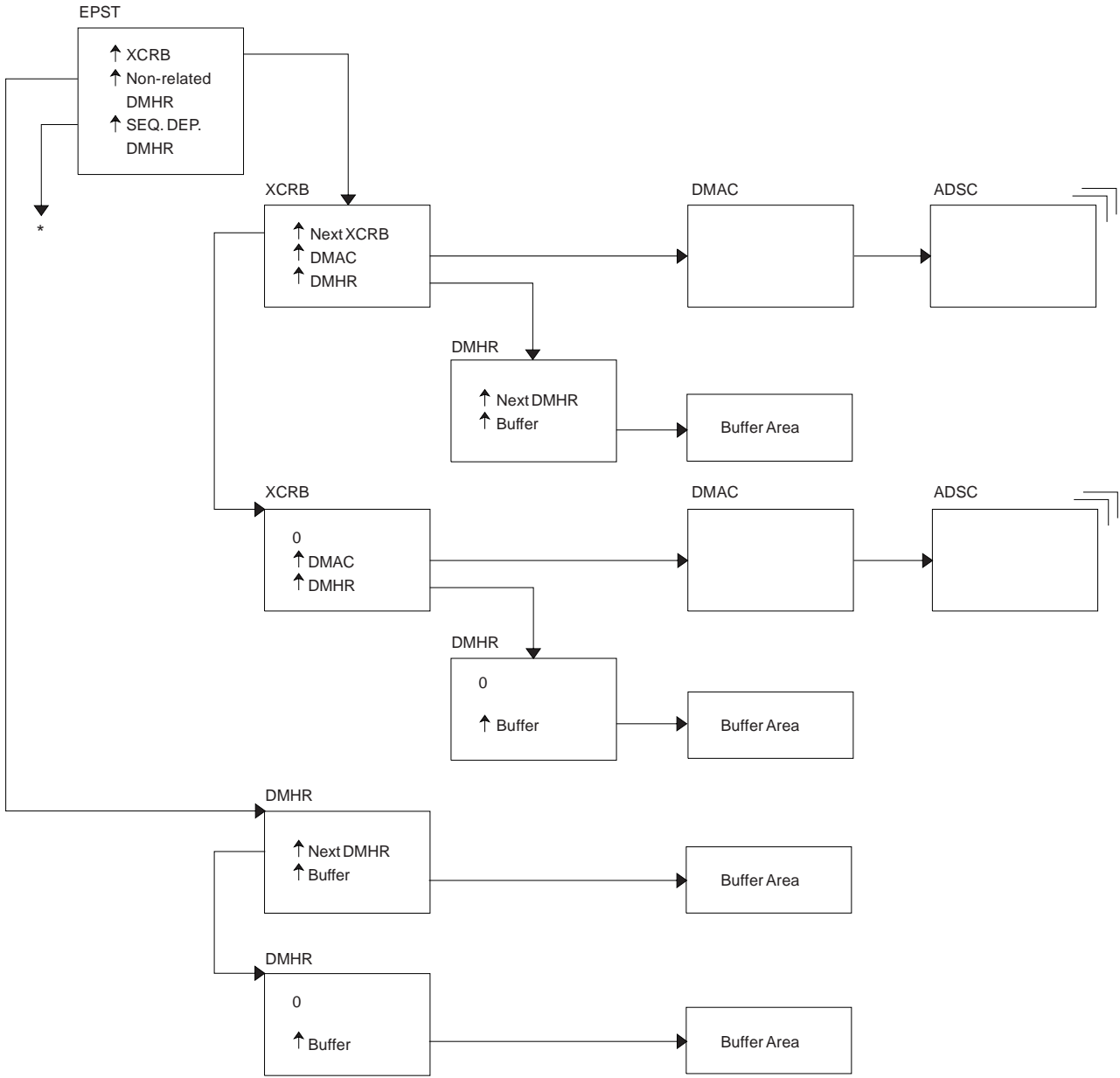


Figure 16. Overview of Fast Path Control Blocks

Figure 17 on page 98 shows the relationships between buffer control blocks for Fast Path databases.



\* EPSTSDBH (This chain is identical to non-related DMHR chain.)

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

Figure 18 on page 99 shows a GSAM control block overview.

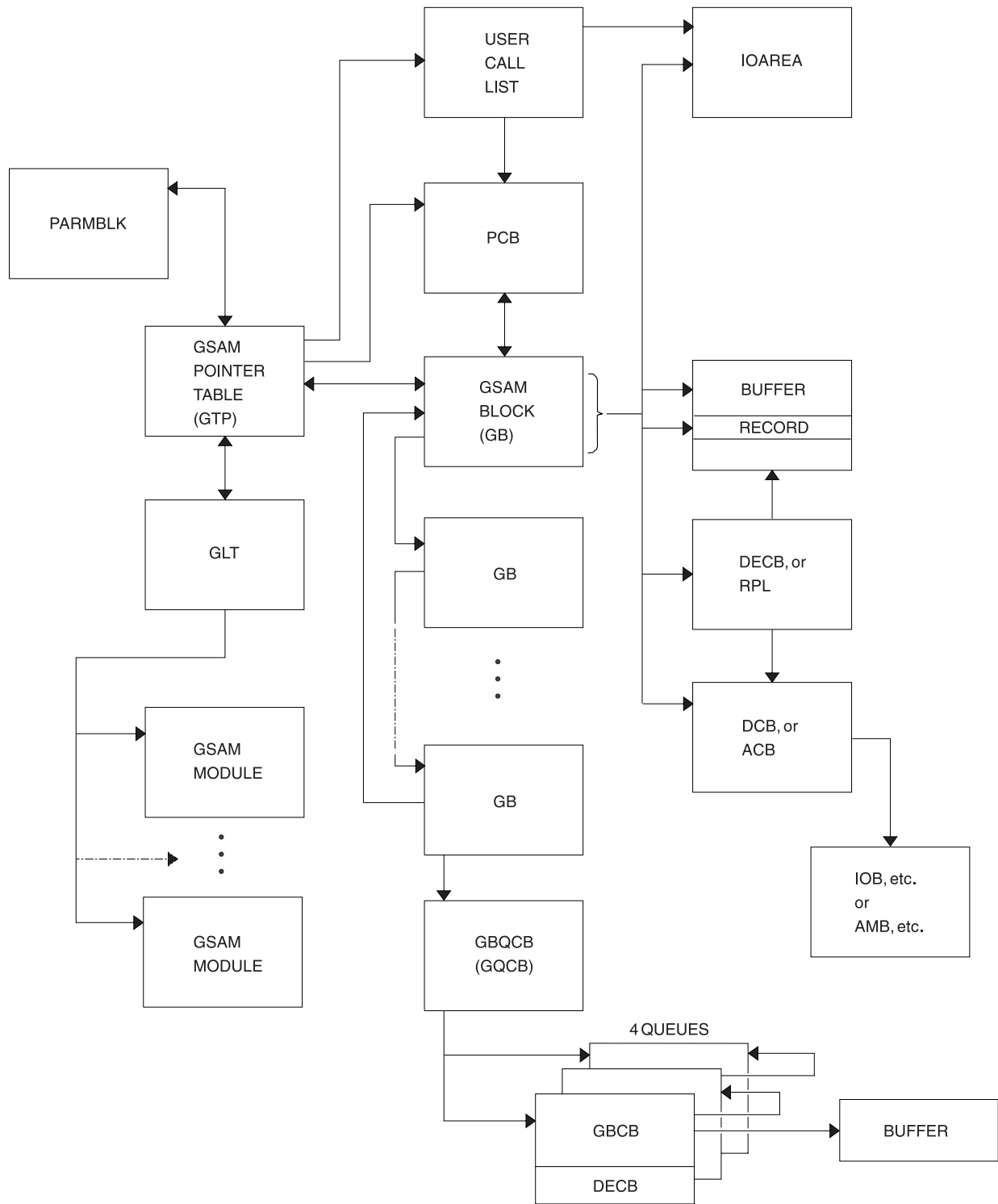


Figure 18. GSAM Control Block Overview

Figure 19 on page 100 shows the GSAM control blocks.

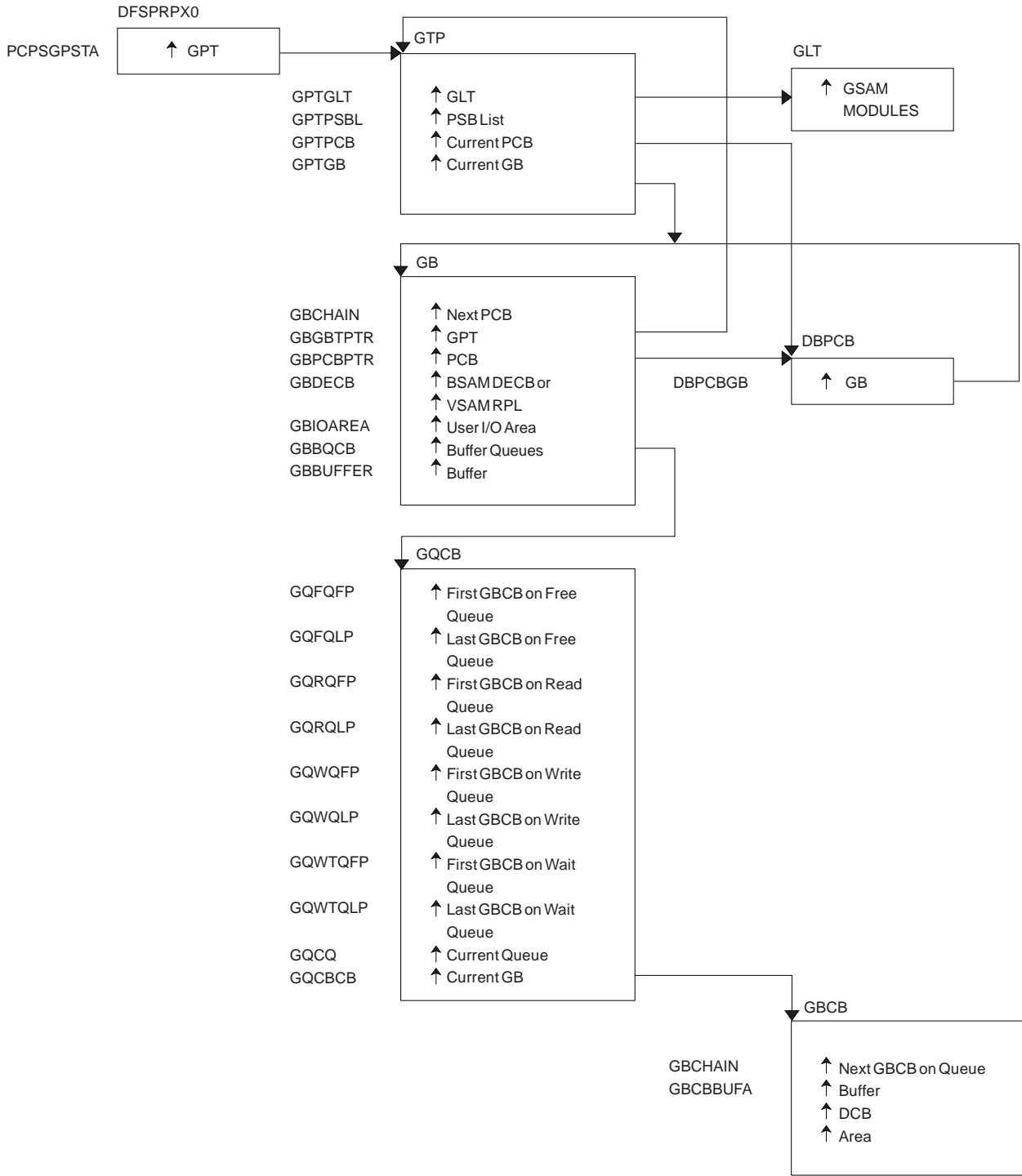


Figure 19. GSAM Control Blocks

Figure 20 on page 101 shows the DL/I control block relationships.

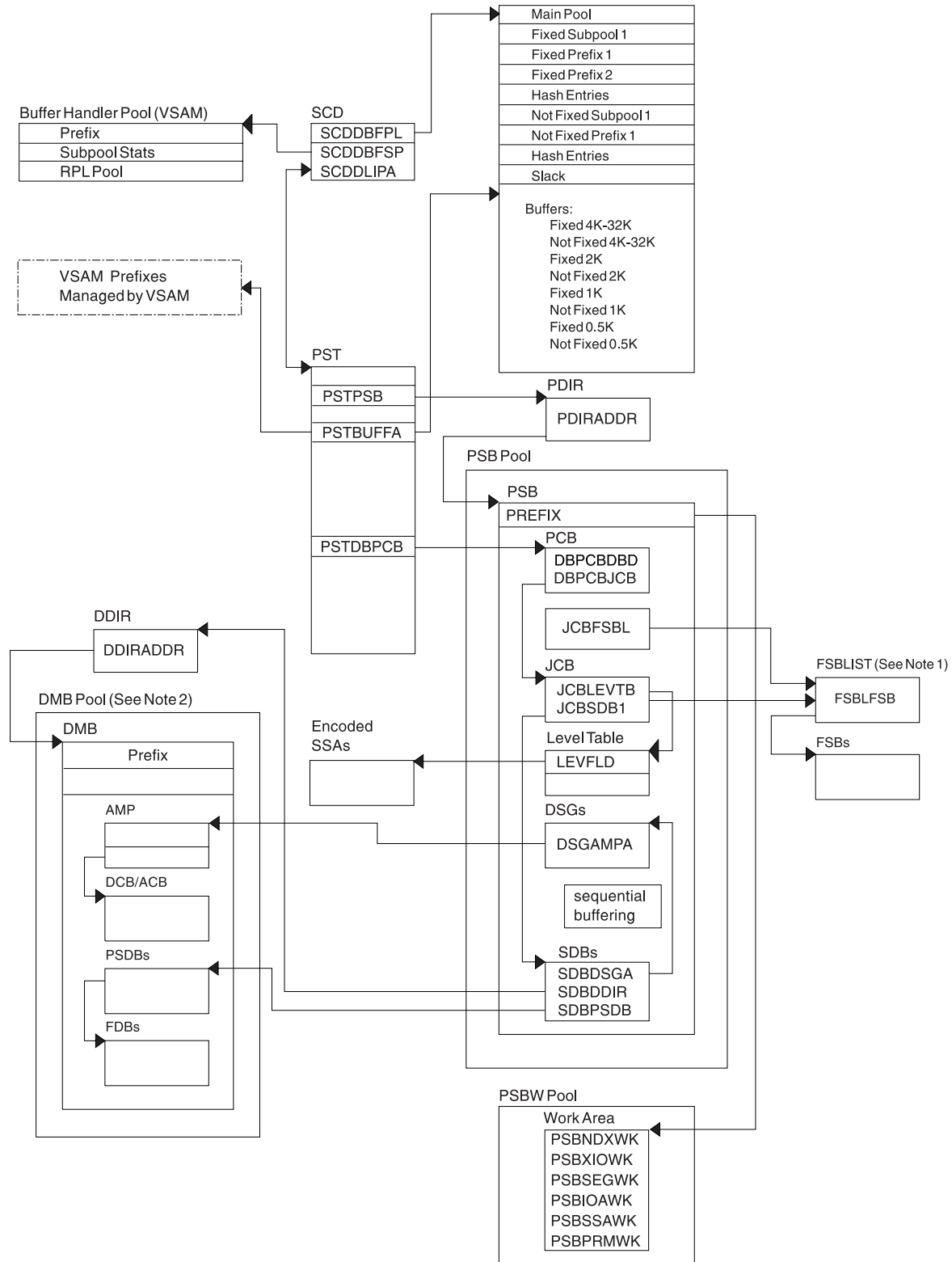


Figure 20. DL/I Control Block Relationships

**Notes to Figure 20:**

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

- 2. A partition HALDB DMB is not in the DMB pool. For HALDB, only the Master DMB is in the DMB pool.

Figure 21 shows the IMS Transaction Manager control blocks.

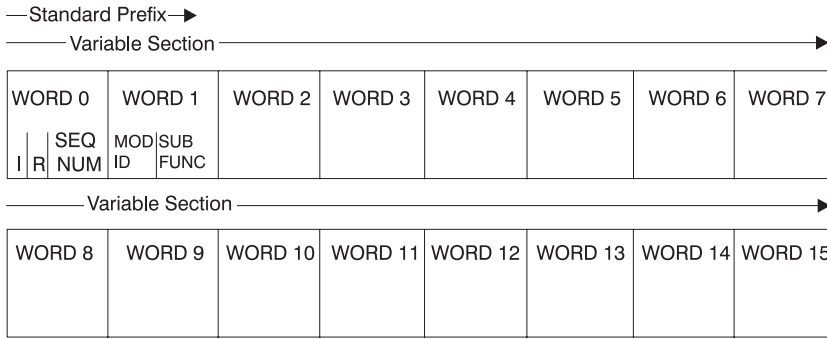
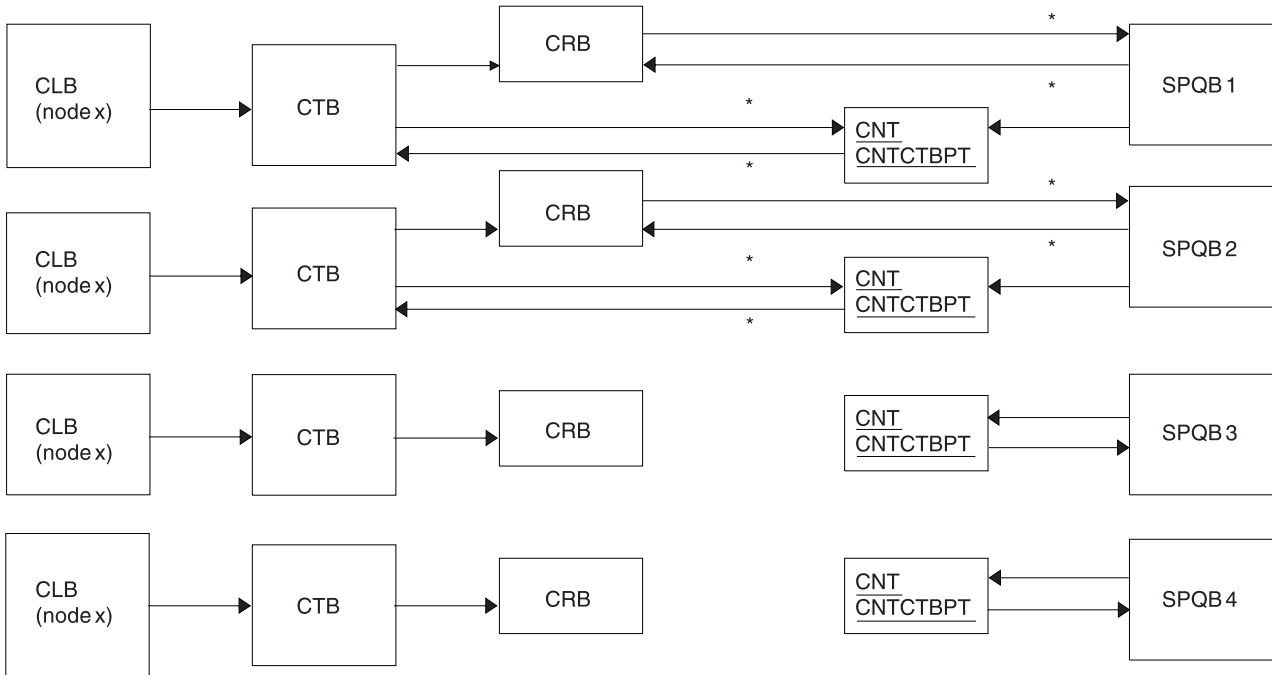


Figure 21. IMS Transaction Manager Control Blocks

Figure 22 shows the intersystem communication control block structure.



**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 on page 103 shows the VTCB Load Module.



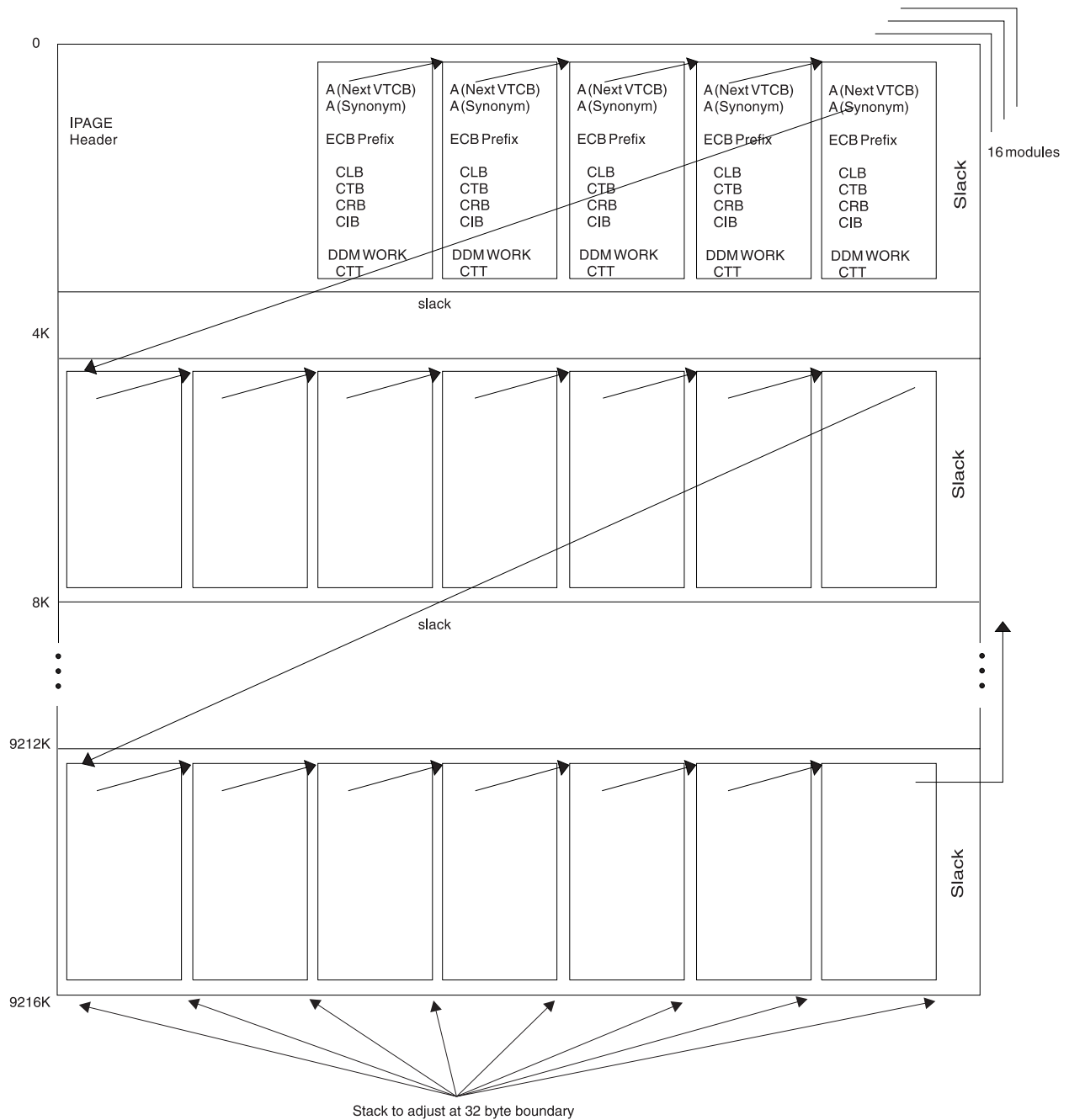


Figure 23. VTCB Load Module

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 on page 105 shows the Multiple Systems Coupling (MSC) control block overview.

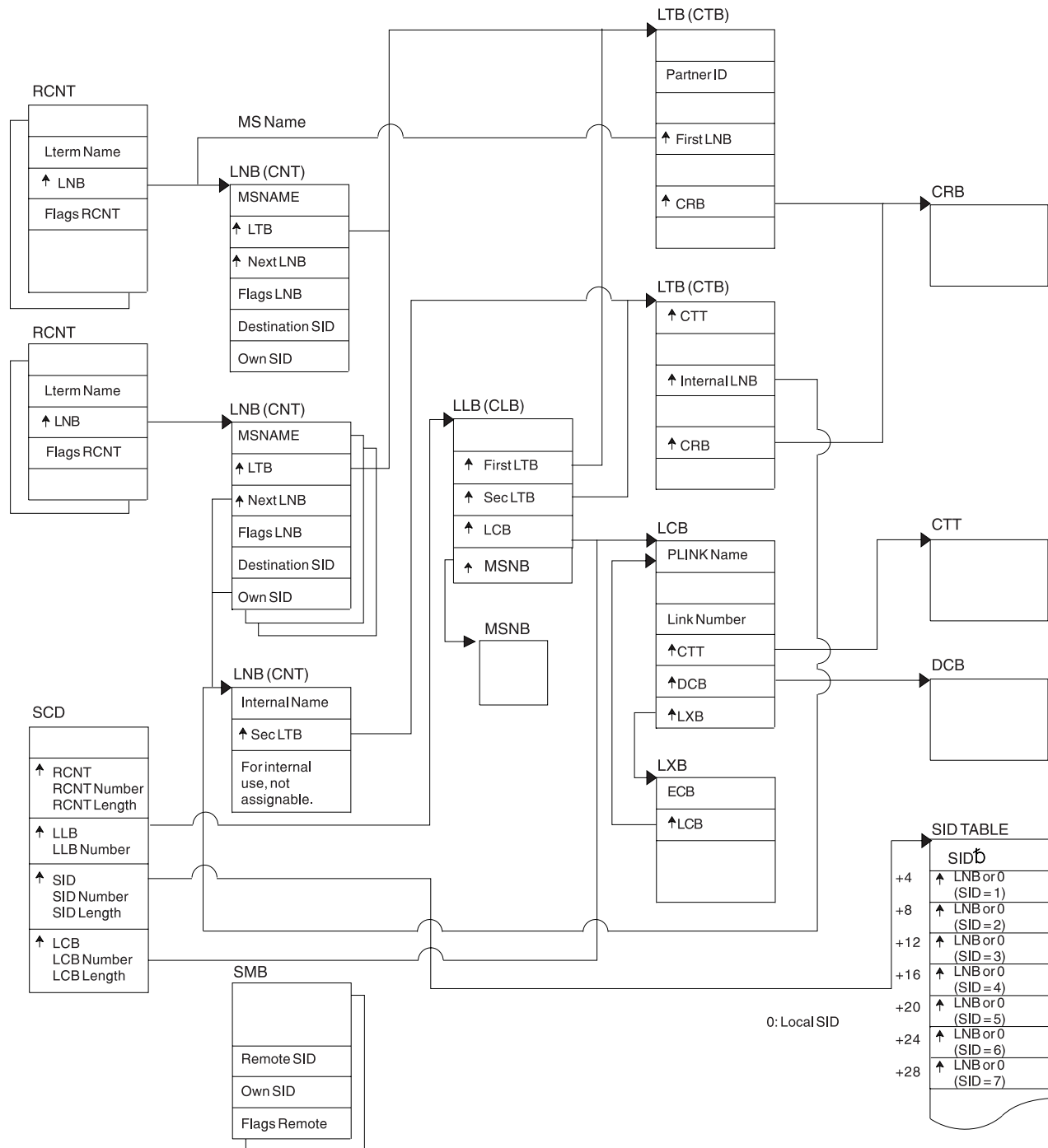


Figure 24. Multiple Systems Coupling (MSC) Control Block Overview

Figure 25 on page 106 shows the Multiple Systems Coupling (MSC) Main Storage-to-Main Storage control block overview.

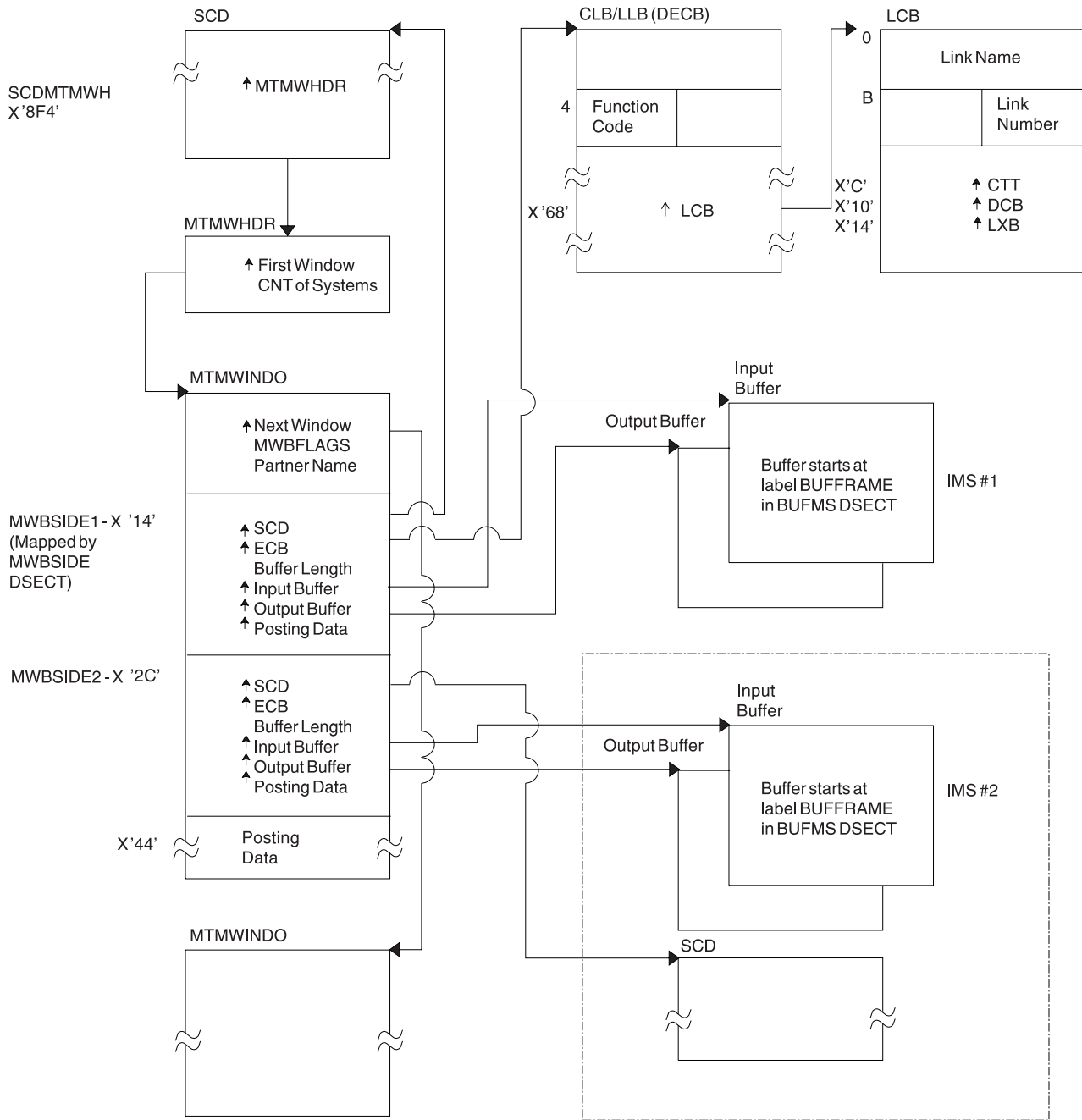


Figure 25. Multiple Systems Coupling (MSC) Main Storage-to-Main Storage Control Block Overview

Figure 26 on page 107 shows a z/OS Storage map displaying IMS-to-IRLM interrelationships.

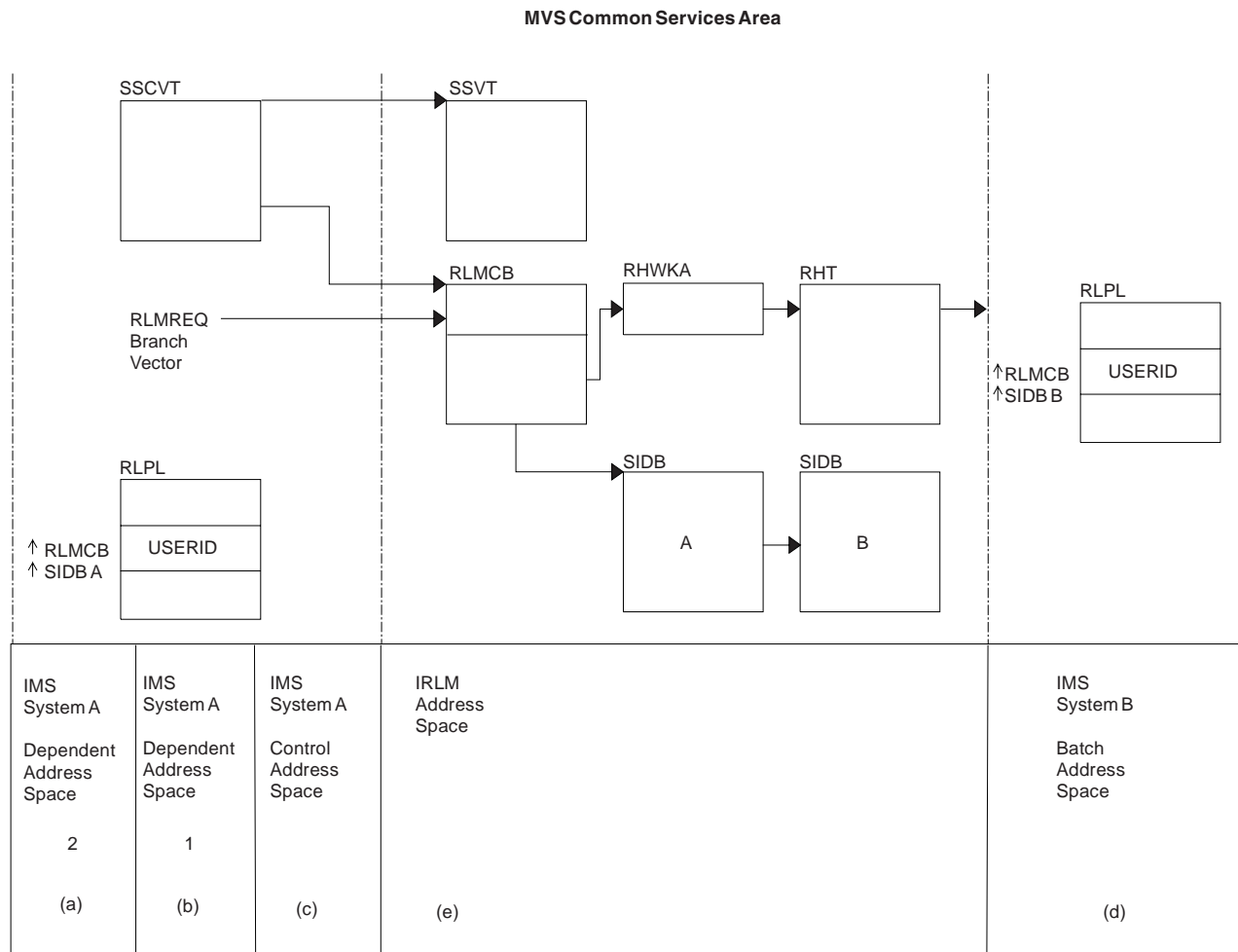


Figure 26. z/OS Storage Map Showing IMS-to-IRLM Interrelationships

**Notes to Figure 26:**

1. (a), (b), and (c) are z/OS address spaces that make up one online IMS subsystem.
2. (d) is a z/OS 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 z/OS 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 on page 108 shows the overall control block structure of IRLM.

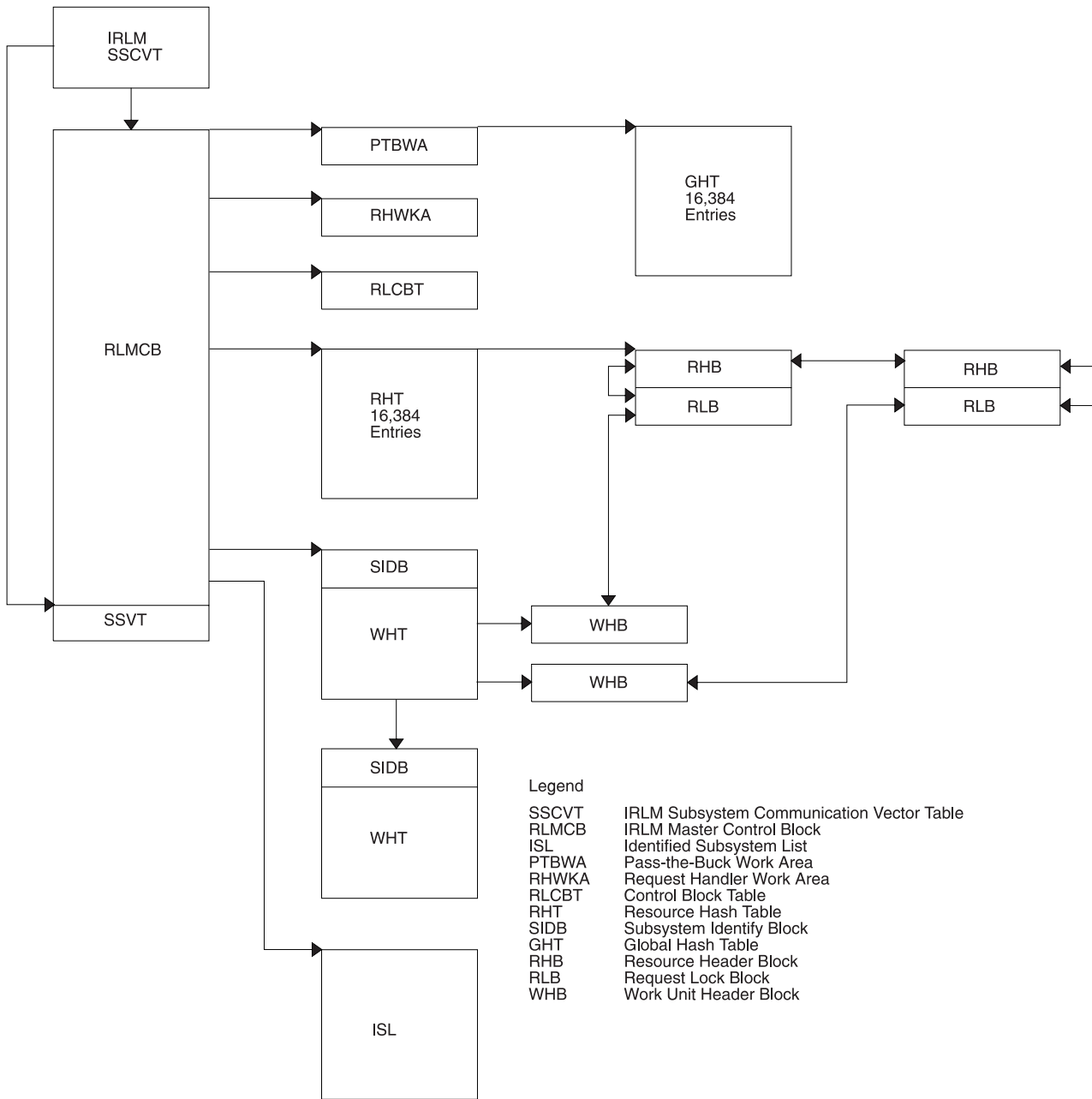


Figure 27. IRLM Overall Control Block Structure

Figure 28 on page 109 shows the IRLM Storage Manager pools.

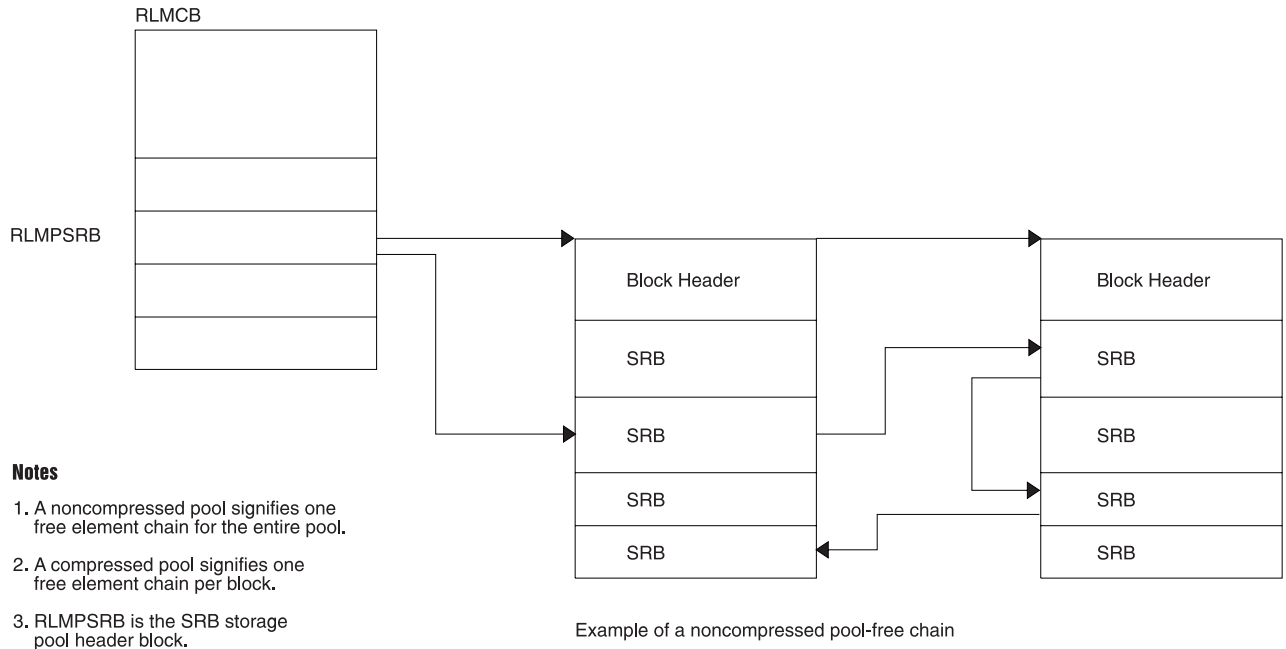


Figure 28. IRLM Storage Manager Pools

Figure 29 shows examples of IRLM lock requests.

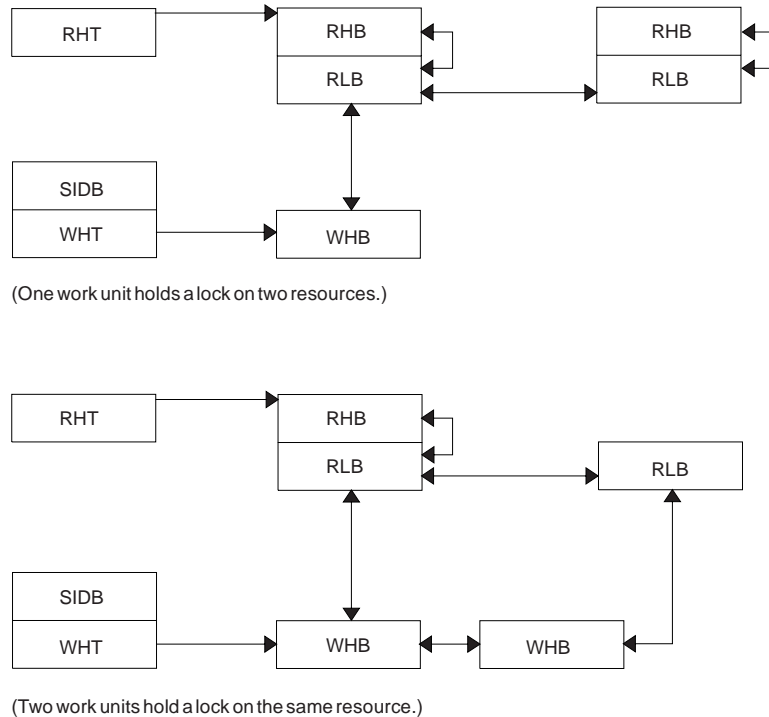


Figure 29. IRLM Lock Request Examples

Figure 30 on page 110 shows an overview of the Database Recovery Control (DBRC) control blocks.

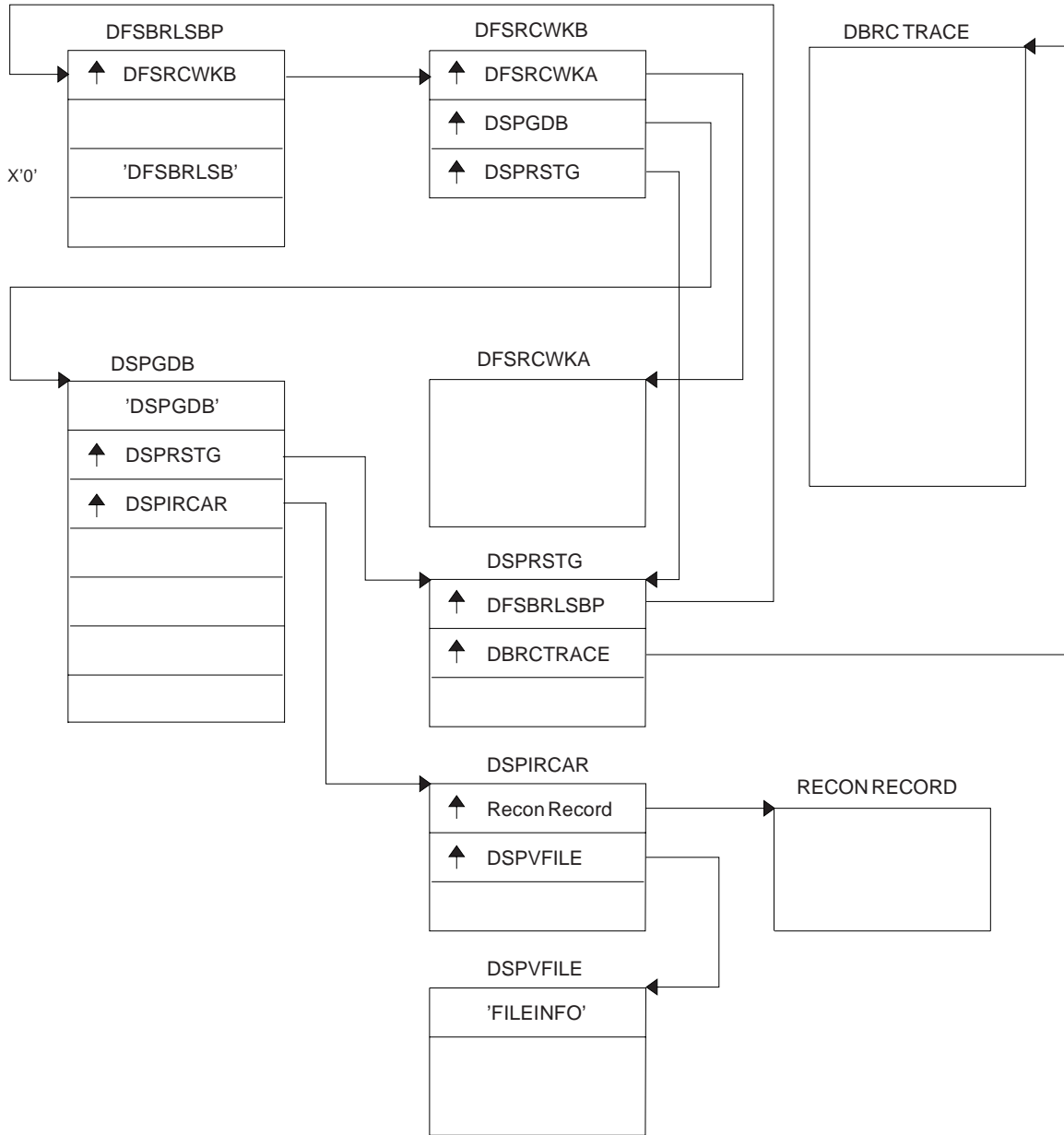


Figure 30. Control Block Overview of Database Recovery Control (DBRC)

Figure 31 on page 111 shows the organization and basic linkages of Description Output Format (DOF) and Message Output Descriptor (MOD).



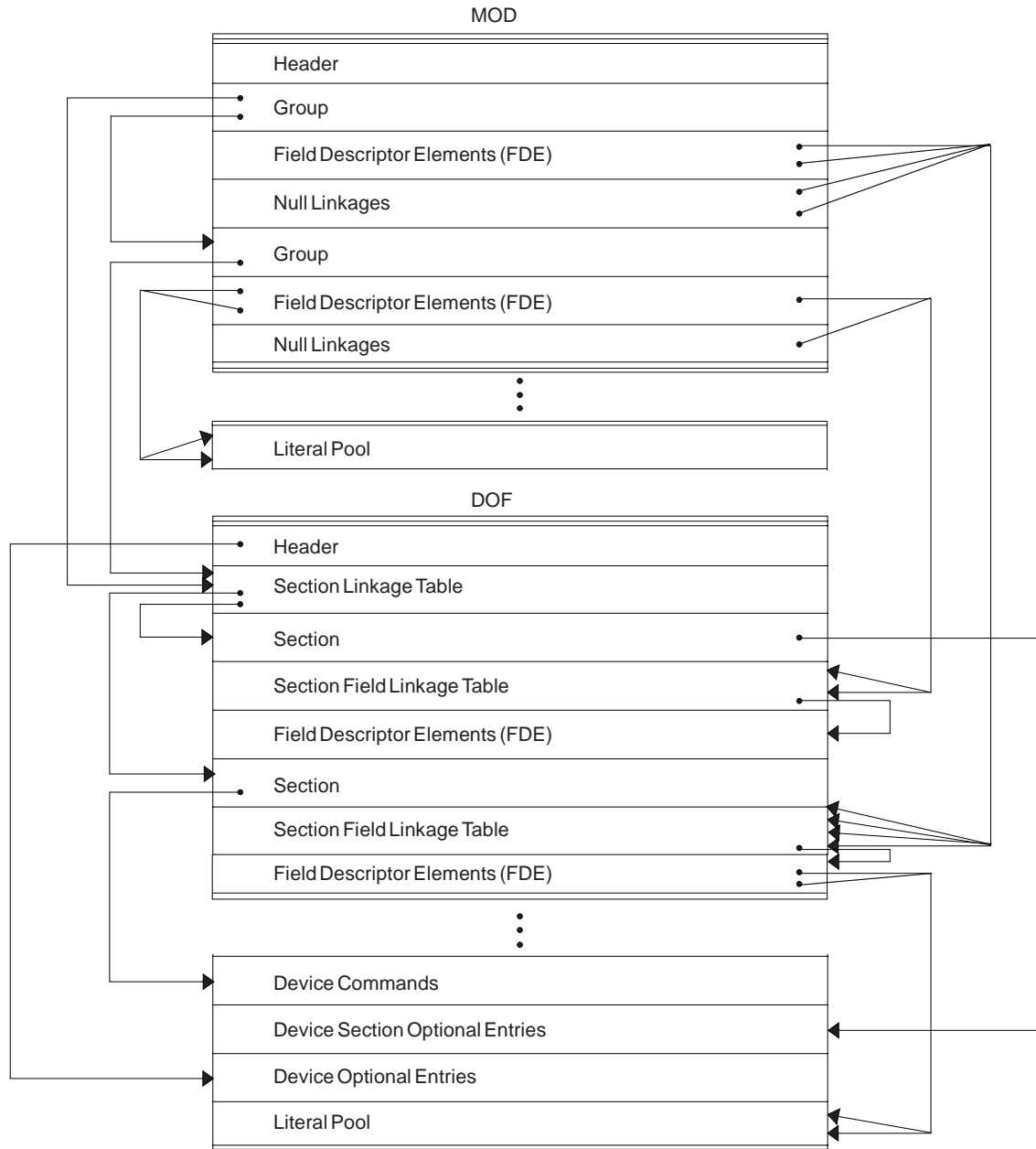


Figure 31. Organization and Basic Linkages: DOF (Device Output Format) and MOD (Message Output Descriptor)

Figure 32 on page 112 shows the organization and basic linkages between Device Input Format (DIT) and Message Input Descriptor (MID).

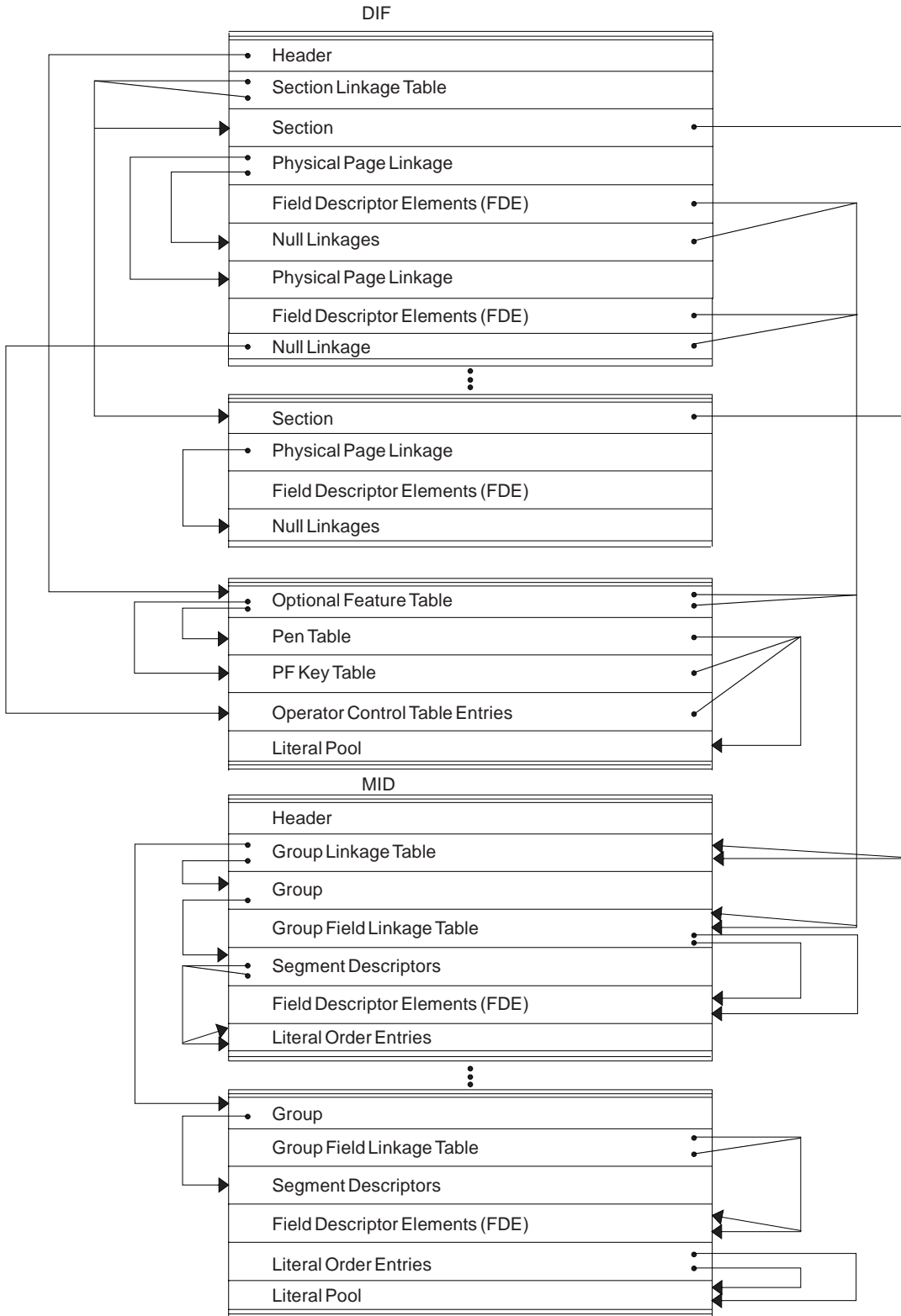


Figure 32. Organization and Basic Linkages: DIF (Device Input Format) and MID (Message Input Descriptor)

## DL/I Record Formats

This topic describes these DL/I data record formats:

- “HSAM and SHSAM Database”
- “HISAM and SHISAM Database” on page 114
- “HDAM, HIDAM, PHDAM, or PHIDAM Database” on page 115
- “OSAM and VSAM ESDS Block Format” on page 117
- “VSAM LRECL for a Primary Index” on page 117
- “Secondary Index or PSINDEX Database (VSAM Only)” on page 118
- “Variable-Length Segments” on page 119

## HSAM and SHSAM Database

### Segment Formats

Figure 33 shows the DL/I data record formats for HSAM and SHSAM databases.

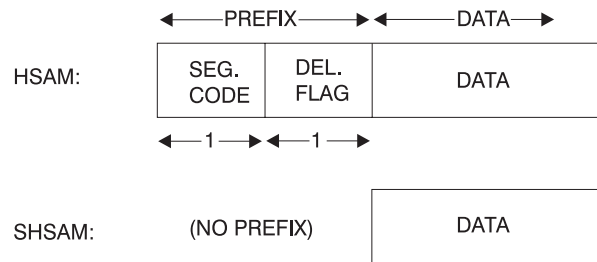


Figure 33. HSAM and SHSAM Segment Format

### Delete Byte (Flag) Format

Figure 34 shows the delete byte (flag) format.

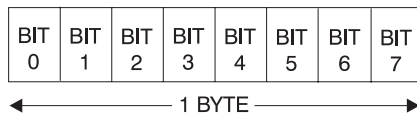


Figure 34. Delete Byte (Flag) Format

Bit	Description
0	Segment deleted (HISAM or index).
1	DB record deleted (HISAM or index).
2	Segment processed by DELETE.
3	Reserved.
4	Data and prefix are separated in storage.
5	Segment has been deleted on its physical path.
6	Segment has been deleted on its logical path.
7	Segment space available to be freed; bits 5 and 6 must also be set on.

### Block Format for HSAM and SHSAM

For SHSAM there are no dependent segments. Block size must be a multiple of segment size. Figure 35 on page 114 shows the block format for HSAM and SHSAM.

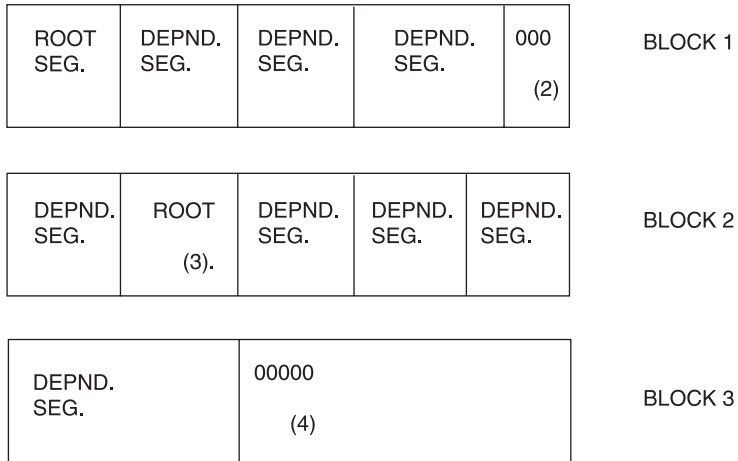


Figure 35. Block Format for HSAM and SHSAM

**Notes:**

1. Pad with zeros if no room for next segment.
2. Next database record starts immediately.
3. Pad with zeros in last block, after last segment.

## HISAM and SHISAM Database

### Segment Format

Figure 36 and Figure 37 show the segment format of HISAM and SHISAM.

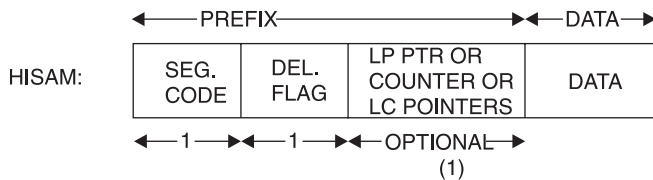


Figure 36. HISAM Segment Format

**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).



Figure 37. SHISAM Segment Format

I      **Note:** This is a root-only database.

### LRECL Format

Figure 38 shows the LRECL format.

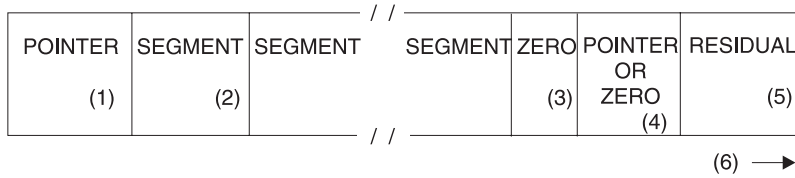


Figure 38. LRECL Format

**Notes:**

1. 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 Figure 37 on page 114)
3. 1-byte of zeros indicates the end of segments in this LRECL.
4. This field is omitted.
5. Space not used.
6. VSAM LRECLs must have an even length.

### VSAM Block Formats

Figure 39 shows the VSAM block formats.

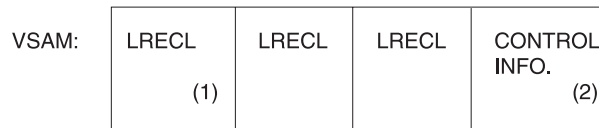


Figure 39. VSAM Block Formats

**Notes:**

1. LRECL length might change between KSDS and ESDS, depending on user definition.
2. Ten bytes if blocked data set; 7 bytes if unblocked data set.

## HDAM, HIDAM, PHDAM, or PHIDAM Database

### Segment Format

Figure 40 shows the segment format of HDAM, HIDAM, PHDAM, or PHIDAM.

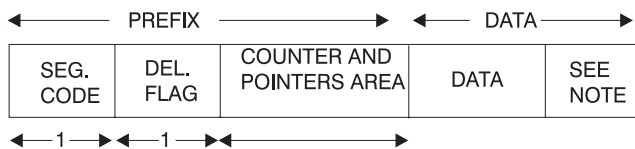


Figure 40. HDAM, HIDAM, PHDAM, or PHIDAM Segment Format

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 41 maps the prefix of a segment.

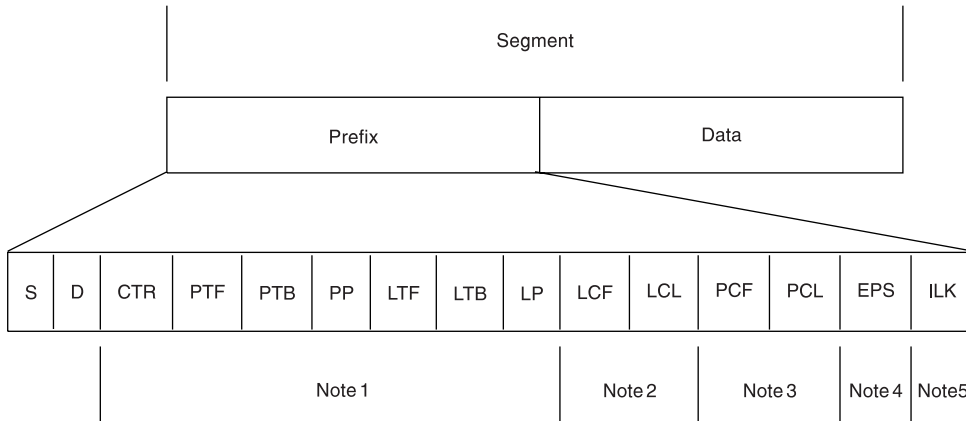


Figure 41. Mapping the Prefix of a Segment

#### Notes to Figure 41:

1.

Prefix Flag	Prefix Flag Description
-------------	-------------------------

	Segment code (S)
--	------------------

	Delete flag (D)
--	-----------------

The pointers that exist in this section of the prefix are identified in the PSDB field DMBPTR, as shown in the following list:

**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 (For 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, 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 DMBSUDE (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 DMBSUDE (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 PHIDAM or PHDAM.

## OSAM and VSAM ESDS Block Format

Figure 42 shows the OSAM and VSAM ESDS block format.

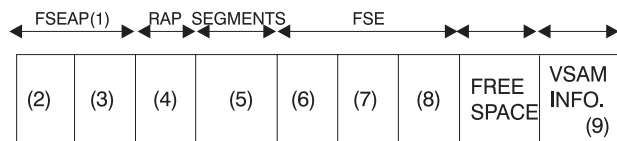


Figure 42. OSAM and VSAM ESDS Block Format

### 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 or PHIDAM with TB or NT is pointing at the 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. 4-byte identification of the 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 for a Primary Index

### On Storage Device and in Buffer Pool

Figure 43 on page 118 shows the format of the on storage device and in buffer pool

DEL. FLAG	PTR (1)	ROOT KEY VALUE
--------------	------------	----------------

Figure 43. 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**

Figure 44 shows the VSAM LRECL format as returned by buffer handler (1).

**Notes:**

(1)	PTR (2)	SEG. CODE	DEL. FLAG (3)	PTR (4)	ROOT KEY VALUE
-----	------------	--------------	---------------------	------------	----------------

Figure 44. VSAM LRECL Format As Returned by Buffer Handler

1. Same as buffer pool format, except for pointer and segment code in front.
2. Four-byte pointer with value of zero.
3. The segment code value is 01.
4. 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.

**VSAM Block Format on Device and in Buffer Pool**

Figure 45 shows the VSAM block format on device and in buffer pool.

LRECL	LRECL	LRECL	VSAM INFO.
-------	-------	-------	------------

Figure 45. VSAM Block Format on Device and in Buffer Pool

**Secondary Index or PSINDEX Database (VSAM Only)**

**LRECL Format on Device and in Buffer Pool**

One segment per LRECL. Figure 46 shows the LRECL Format on Device and in Buffer Pool.

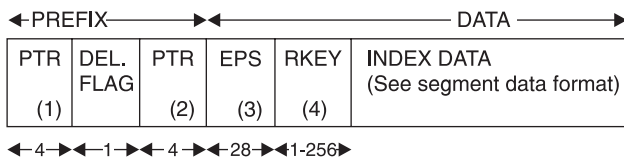


Figure 46. LRECL Format on Device and in Buffer Pool

**Notes:**

1. Nonunique keys: This points to ESDS LRECL with the same key value. Unique keys or PSINDEX: This field is omitted.
2. Direct pointer to index target segment. Omit this field if symbolic 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 RKEY means root key. 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 256 bytes.

**LRECL as Returned by Buffer Handler**

Figure 47 shows LRECL as returned by buffer handler.

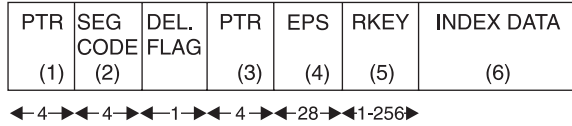


Figure 47. LRECL as Returned by Buffer Handler

**Notes:**

- 1. Four-byte pointer contains zeros.
- 2. Code value is 01.
- 3. Direct pointer to index target segment. Omit this field if symbolic 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 256 bytes.
- 6. Sequential segment data format.

**Block Format on Device and in Buffer Pool**

Figure 48 shows the block format on device and in buffer pool.

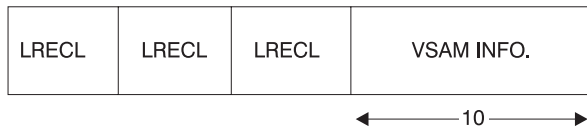


Figure 48. VSAM Block Format on Device and in Buffer Pool

**Segment Data Format**

Figure 49 shows the segment data format.

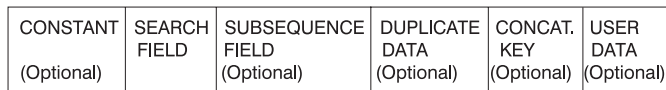


Figure 49. Segment Data Format

**Variable-Length Segments**

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

Figure 50 on page 120 shows the HISAM, HDAM, HIDAM, PHDAM, and PHIDAM Segment Format.

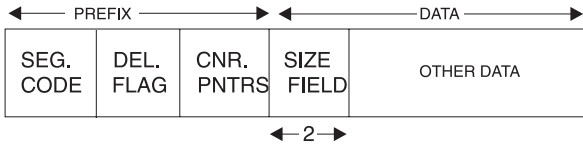


Figure 50. HISAM, HDAM, HIDAM, PHDAM, and PHIDAM Segment Format

**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. Figure 51 shows HDAM, HIDAM, PHDAM, and PHIDAM.

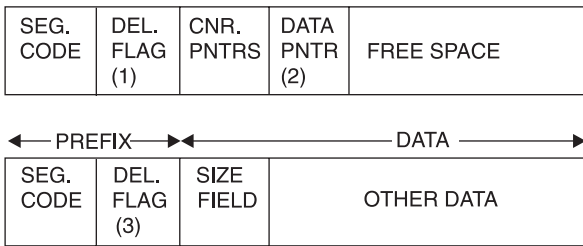


Figure 51. HDAM, HIDAM, PHDAM, and PHIDAM

**Notes:**

1. DEL FLAG containing X'08' indicates that the data has been separated from the prefix.
2. DATA PNTR is a direct pointer to the segment containing the "other data".
3. The flag value is X'FF'.

## Part 3. Diagnostic Aids

<b>Chapter 7. SYS—System Service Aids</b>	127
Log Records	127
Format of X'29' Log Record	141
Format of X'67' Log Record	150
File Select and Formatting Print Utility	152
Log Merge Utility	154
Formatting IMS Dumps Offline	154
Introduction to the Offline Dump Formatter.	154
Solving IMS Problems with the Dump Formatter.	155
Using the Formatted Dump	170
Edited Command Buffer Format.	177
Interactive Dump Formatter	178
Using Interactive Dump Formatter Menus	179
Using the Other IMS Components Formatting Panels	182
Using the Other IMS-Related Products Formatting Panels	182
IMS IPCS Symbols	182
Using IMS Enhanced Dump Analysis	183
Formatting IMS Dumps Online	183
Formatted Dump for the CTL Address Space	184
Formatted Dump for the DL/I Address Space	187
SNAP Call Facility	189
/DIAGNOSE Command SNAP Function	190
Common Trace Table Interface	190
Finding the Trace Tables in a Dump	192
Format of Trace Records	194
IMS Trace Function Codes	194
Common Service Layer Trace	197
Dispatcher Trace	203
ITASK ECB Posting	211
System Post Codes	212
External Subsystem Trace	212
Layout of the X'57' Variable Section	214
Layout of the X'58' Variable Section	219
Resource Recovery Services Trace	225
Scheduler Trace	238
Storage Manager Trace	241
Latch Trace	242
Queue Manager Trace	246
Shared Queues Interface Trace	250
Fast Path Trace	251
<b>Chapter 8. DB—Database Service Aids</b>	253
The Job Control Block (JCB) Trace	253
Sample JCB Trace	254
JCB Trace Call Function Codes	254
Data Language/I Test Program—DFSDDLTO	255
COMPARE Statement SNAPs	255
SNAPs on Exceptional Conditions	256
DL/I Call Image Capture	257
Batch Environment	258
Online Environment	258
How to Retrieve DL/I Call Image Capture Data from the Log Data Set	258
DL/I Analysis	258

IMS Abends . . . . .	259
Dump Analysis—General . . . . .	259
Dump Analysis—Detailed . . . . .	259
Generalized DL/I Problem Analysis . . . . .	261
Locating Database-Related Traces . . . . .	262
DL/I Trace. . . . .	263
Using the DL/I Trace . . . . .	264
DL/I Trace Formats . . . . .	264
DELETE/REPLACE—DL/I Trace Information . . . . .	300
Retrieve Trace . . . . .	301
I Online Recovery Manager Trace . . . . .	306
Starting the Online Recovery Manager Trace . . . . .	306
Format of the Online Recovery Manager Trace . . . . .	306
Online Recovery Manager Trace Example . . . . .	310
Program Isolation-Related Problem Analysis . . . . .	311
Limiting Locking Resources Used by an Application Program . . . . .	311
Program Isolation (PI) Trace . . . . .	311
DL/I Call Image Capture Program . . . . .	312
Log Analysis (Database Related) . . . . .	312
Sequential Buffering Service Aids . . . . .	316
SBSNAP Option . . . . .	317
SBESNAP Option . . . . .	318
SB IMAGE CAPTURE Option and SB Test Program (DFSSBHD0 Utility) . . . . .	318
SB COMPARE Option . . . . .	318
GSAM Control Block Dump—DFSZD510 . . . . .	319
Example of a Formatted GSAM Control Block Dump . . . . .	319
Example of an Unformatted GSAM Control Block Dump . . . . .	321
Recovering from Out-of-Space Sx37 Abends on GSAM Data Sets . . . . .	322
<b>Chapter 9. DC—Data Communication Service Aids . . . . .</b>	<b>325</b>
Terminal Communication Task Trace . . . . .	325
Entry Points . . . . .	326
Trace Records . . . . .	326
Trace Output. . . . .	327
DC Trace . . . . .	327
Starting the Trace . . . . .	327
Stopping the Trace . . . . .	328
Printing the Trace Records . . . . .	329
Content of the Trace Records . . . . .	330
Diagnosing Line and Terminal Problems. . . . .	334
Diagnosing Problems in the Queue Control Facility/Message Requeuer . . . . .	341
The Query Control Facility Interface . . . . .	342
Using SCRAPLOG Diagnostic Records . . . . .	343
Using 6701-MRQE Diagnostic Records . . . . .	345
Obtaining Diagnostics in Addition to SCRAPLOG and 6701-MRQE . . . . .	347
How to Tell When Messages Have Been Successfully Requeued . . . . .	348
Diagnosing Message Routing Problems . . . . .	348
DFS070 UNABLE TO ROUTE MESSAGE RSN=xyxy. . . . .	349
Using the DFSMSCE0 Routing Exit Trace . . . . .	354
Using the Transaction/Program Trace to Diagnose Routing Errors . . . . .	356
Using the DC LINE/NODE/LINK TRACE to Diagnose Routing Problems . . . . .	357
Using 01/03 Log Record Trace . . . . .	358
I IMS Transaction Trace . . . . .	358
Receive-Any Buffer Analysis . . . . .	361
Finding the Active Save Set . . . . .	363
IMS-VTAM Interface . . . . .	363

IBM 3270 Error Recovery Analysis . . . . .	363
Message Format Service Normal BTAM Path . . . . .	364
Diagnosing Message Format Service Problems . . . . .	369
Message Format Service Module Traces . . . . .	371
Tracing Errors in Module DFSCNXA0 . . . . .	372
Location Codes for DFSCNXA0 Error Messages . . . . .	373
Qualifier Codes . . . . .	379
IDC0 Trace Table Entries . . . . .	379
APPC/IMS Diagnostic Aids . . . . .	382
LU Manager Trace . . . . .	382
LU 6.2 Module-to-Code Cross-Reference Table . . . . .	390
APPC/MVS Verb-to-Code Cross-Reference Table . . . . .	391
DFS1959E Message Information . . . . .	392
DFS1965 APPC/MVS Call Failure . . . . .	399
Diagnostics for Use with Synchronous APPC and OTMA with Shared Queues . . . . .	400
SNAPs and Dumps . . . . .	400
OTMA Diagnostic Aids . . . . .	400
OTMA Trace . . . . .	400
OTMA Trace Entry for User Exits . . . . .	402
OTMA Module-to-Code Cross-Reference Table . . . . .	403
OTMA Verb-to-Code Cross-Reference Table . . . . .	404
DFS1269E Message Information . . . . .	405
OTMA Log Records . . . . .	405
SNAPs and Dumps . . . . .	405
Diagnosing Errors Diagnosing Fast Path Problemsrelated to Print Data Set Options: IMS Spool API Support . . . . .	405
Understanding Parsing Errors . . . . .	405
Debugging and Diagnostic Aids Provided by IMS Spool API . . . . .	409
<b>Chapter 10. IRLM Service Aids . . . . .</b>	<b>411</b>
IRLM Lock Trace Analysis Using KBLA . . . . .	411
IRLM Dumps. . . . .	412
SYS1.LOGREC. . . . .	412
z/OS Component Trace. . . . .	412
<b>Chapter 11. FP—Fast Path Service Aids . . . . .</b>	<b>415</b>
Diagnosing Fast Path Problems. . . . .	415
ABENDU1026 Analysis . . . . .	415
Fast Path Transaction Retry . . . . .	418
DEDB Control Interval (CI) Problem Assistance Aids . . . . .	419
CI Type Identification. . . . .	419
DEDB CI Formats . . . . .	419
Locating Fast Path Control Blocks and Tables . . . . .	422
Fast Path External Trace . . . . .	424
Trace Activation . . . . .	424
Trace Deactivation . . . . .	424
Diagnostic Data . . . . .	425
Fast Path Trace Entries. . . . .	425
Fast Path External Trace Examples . . . . .	425
<b>Chapter 12. MSC—Multiple Systems Coupling Service Aids . . . . .</b>	<b>433</b>
Multiple Systems Coupling Communication Task Trace . . . . .	433
Multiple Systems Coupling Device-Dependent Module . . . . .	433
Multiple Systems Coupling Traces . . . . .	435
Diagnosing Link Problems . . . . .	435
MSS1 and MSS2 Records. . . . .	440

Channel-to-Channel Access Method Trace Stack (LXB Trace)	441
DFSCMC00 (MSC Analyzer)	442
DFSCMC50 (Shutdown Processing Routine)	442
DFSCMC40 (Attention DIE Routine)	442
DFSCNC40 (I/O Request DIE Routine)	443
DFSCMC10 (Channel-End Appendage)	443
DFSCMC10 (Abnormal-End Appendage)	444
DFSCMC10 (Shutdown Appendage)	444
MSC Routine Trace—BUFMSVID	446
<b>Chapter 13. DBRC—Database Recovery Control Service Aids</b>	<b>447</b>
Diagnosing from a RECON List	447
RECON Record Types	447
DBRC Internal Trace	450
Trace Input	452
Locating the Trace	452
Trace Output	452
Trace Header Record	452
Module Call, Module Return, and DSPSTACK Trace Entries	453
BGNCABN0, DSPCABN0, BGNRETRY, DSPCRTR0, and CRTR0XIT Trace Entries	455
DSPURI00 Trace Entries	457
DBRC Internal Unformatted Trace Example	462
DBRC External Trace	469
Examples of Output	470
Samples of JCL to Create Trace Output	471
I DBRC API Return and Reason Codes	472
<b>Chapter 14. DRA—Database Resource Adapter Service Aids</b>	<b>477</b>
DRA Dumps	477
SDUMP	478
SNAPs	478
Recovery Tokens	478
Analyzing DRA Problems	478
Procedure	478
Notes on Dumping	479
<b>Chapter 15. RSR—Remote Site Recovery Service Aids</b>	<b>481</b>
Determining Last Non-MSC Message Recorded	481
Determining Last MSC Message Recorded	483
Fast Path Tracker Trace Entries	483
I X'D4': Database Tracker Trace Entries (D4)	489
I Buffer Handler Trace Entries at Database Tracker	492
Log Router Trace Data	492
X'4930': Database Tracker FSE Error Log Record Format	519
<b>Chapter 16. CQS Diagnosis</b>	<b>521</b>
I Diagnosing a CQS Related Problem	521
I    CQS Additional Manual Intervention for Dump Creation	522
I    CQS Structure Dump Contents	522
I    CQS - z/OS Log Stream	524
I    CQS Structure Recovery Data Set	524
I    CQS Checkpoint Problems	524
I CQS Structure Rebuild Problems	525
I CQS Trace records	526
CQS Log Records	529
Printing CQS Log Records	531

Copying CQS Log Records for Diagnostics . . . . .	532
I <b>Chapter 17. CSL Diagnosis</b> . . . . .	535
I CSL Trace Records . . . . .	535
I RM Trace Record Example . . . . .	537





## Chapter 7. SYS—System Service Aids

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

### In this section:

- “Log Records” discusses the log records, their formats, and the modules that issue them.
- “File Select and Formatting Print Utility” on page 152 discusses the File Select and Formatting Print utility (DFSERA10) which prints various log records from the IMS log data set.
- “Formatting IMS Dumps Offline” on page 154 discusses the Offline Dump Formatter.
- “Edited Command Buffer Format” on page 177 discusses the edited command buffer.
- “Interactive Dump Formatter” on page 178 discusses the interactive dump formatter.
- “Formatting IMS Dumps Online” on page 183 discusses the Online Dump Formatter.
- “SNAP Call Facility” on page 189 discusses the SNAP call facility.
- “/DIAGNOSE Command SNAP Function” on page 190 discusses the /DIAGNOSE command SNAP function.
- “Common Trace Table Interface” on page 190 discusses 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 8 lists each log record and:

- The MACRO or COPY file that maps the record
- The conditions that cause the record to be created
- The module that issues the record

Table 8. IMS Log Records Used to Analyze IMS Problems

Type	Mapping Macro Name	DSECT Name	Why Written (Issuing Module)
X'01'	QLOGMSGP	QLOGMSGP	Data was put in a message queue buffer. Caller is data communication. (DFSQLOG0)
X'02'	DFSLOG02	CMLOG	A /LOG command or a command that alters data required for restart was successfully completed. (DFSICLP0)
X'03'	QLOGMSGP	QLOGMSGP	Data was put in a message queue buffer. Caller is DL/I. (DFSQLOG0)
X'06'	DFSLOG06	ACLOGREC	IMS was started or stopped, or FEOV was issued. The VTAM TPEND exit routine 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. (DFSFLG0, DFSFDLM0, DFSICA20, DFSICL40, DFSRDSH0)

Table 8. IMS Log Records Used to Analyze IMS Problems (continued)

Type	Mapping Macro Name	DSECT Name	Why Written (Issuing Module)
X'07'	DFSLOG07	DLREC	An application program terminated. (DFSRBLB0, DFSRBOI0, DFSSABN0, DFSDABN0, DFSDLA30, DFSTMAD0)
X'08'	DFSLOG08	LINTREC	An application program was scheduled. (DFSSMSC0, DFSSBMP0, DFSDASP0, DFSDLA30, DFSTMAD0)
X'09'	SBLOGREC	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)  <b>X'01'</b> Sequential buffering summary statistic for the PST. <b>X'02'</b> Sequential buffering detailed statistics for each SDSG.
X'0A07'	DFSLOG0A	L0AREC	A CPI-communications driven application program terminated. (DFSSABN0)
X'0A08'	DFSLOG0A	L0AREC	A CPI-communications driven application program was scheduled. (DFSSMSC0)
X'10'	DFSLOG10	SCREC	A security violation occurred. (DFSICIO0, DFSCMD30, DFSICLZ0, DFSTMAD0)
X'11'	LCONVERS	LCONVERS	A conversational program started. (DFSCON00)
X'12'	LCONVERS	LCONVERS	A conversational program terminated. (DFSCON20)
X'13'	DFSLOG13	LOG13	This log record contains conversational CCBs at logon for static non-ISC terminal, signon for ETO user, or static ISC allocation. (DFSRMD00)
X'14'	DFSLOG14	LNREC	A dial line was disconnected. (DFSICIO0, DFSICLA0)
X'15'	DFSLOG14	LNREC	A dial line was connected. (DFSICA10)
X'16'	DFSLOG16	LOG16	A /SIGN command successfully completed. (DFSICLZ0, DFSCBDL0)
X'18'	DFSLOG18	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'	DFSLOG20	ILRDOC	A database was opened. (DFSDLOC0)
X'21'	DFSLOG20	ILRDOC	A database was closed. (DFSDLOC0)
X'24'	DFSLOG24	ERLGDST	The buffer handler detected an I/O error. (DFSDVSM0, DBFMER00)
X'25'	DFSLOG25	EEQLOG	An EEQE was created or deleted. (DFSTOLG0)
X'26'	DFSLOG26	IOTBUF	An I/O toleration buffer was created. (DFSTOLG0)
X'27'	DFSLOG27	DBXLOG	A data set was extended, according to these subcodes:  <b>X'01'</b> Data set extend phase 1. (DFSDVSM0) <b>X'02'</b> Data set extend phase 2. (DFSDBH10)
X'28'	DFSLOG28	PH1DC	The IMS restart facility updated the sequence numbers of input messages for response mode non-Fast Path transactions from STSN devices. (DFSFXC40)

Table 8. IMS Log Records Used to Analyze IMS Problems (continued)

Type	Mapping Macro Name	DSECT Name	Why Written (Issuing Module)
X'29'	DFSLOG29	DFSLOG29	<p>The progress of a HALDB online reorganization is represented in the following subcodes:</p> <p><b>X'00'</b> The OLR command was received. (DFSORC00, DFSORC10)</p> <p><b>X'10'</b> Ownership of the reorganization for a partition was established through DBRC. (DFSORP60)</p> <p><b>X'20'</b> The UPDATE OLREORG command updated either the RATE option or the [NO]DEL option for a HALDB partition. (DFSORC00, DFSORC10)</p> <p><b>X'30'</b> The output data sets were successfully validated or created. One record includes all output data sets. (DFSORA00, DFSRDBL0)</p> <p><b>X'40'</b> Cursor active. Initialization of the reorganization of the partition was completed successfully, two sets of data sets exist, and copying is about to begin. The partition is now in cursor-active status. (DFSORP60, DFSORP70)</p> <p><b>X'50'</b> The cursor was updated, but the unit of reorganization was not committed. (DFSORP20)</p> <p><b>X'70'</b> Cursor inactive. Copying from the input to the output data sets has completed. The output data sets become active, and the input data sets become inactive. (DFSORP60, DFSORP70)</p> <p><b>X'90'</b> Ownership of the reorganization for a partition was relinquished. This is followed by the X'07' log record for OLR ITASK termination. (DFSORP60)</p> <p><b>Related Reading:</b> For more information on X'29' log records, see "Format of X'29' Log Record" on page 141.</p>
X'30'	QLOGMSGI	QLOGMSGI	A message prefix was changed. (DFSQLOG0)
X'31'	QLOGGETU	QLOGGETU	A GU was issued for a message. (DFSQLOG0)
X'32'	QLOGREJE	QLOGREJE	A message was rejected. It was presumed to have been the cause of an application program abend. (DFSQLOG0)
X'33'	QLOGFREE	QLOGFREE	The queue manager released a record. (DFSQLOG0)
X'34'	QLOGCANC	QLOGCANC	A message was canceled. (DFSQLOG0)
X'35'	QLOGENQU	QLOGENQU	A message was enqueued or re-enqueued. (DFSQLOG0)
X'36'	QLOGDEQS	QLOGDEQS	A message was dequeued or saved or deleted. (DFSQLOG0)
X'37'	DFSXFER QLOGXFER	DFSXFER QLOGXFER	<p>Records marked as NO INPUT and NO OUTPUT are written by the sync point coordinator when all resource managers have completed Phase 1. (DFSFXC30, DBFSLG20)</p> <p>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)</p> <p>Phase 2 DC processing. One or more output messages were transferred from a queue block anchored off the PST temporary output queue to a permanent destination. There is a X'37' record for each destination that has messages transferred. (DFSQLOG0)</p>

| Table 8. IMS Log Records Used to Analyze IMS Problems (continued)

Type	Mapping Macro Name	DSECT Name	Why Written (Issuing Module)
X'38'	QLOGRELI	QLOGRELI	<p data-bbox="699 289 1419 342">An input message was put back on the input queue when the application abnormally terminated. (DFSQLOG0)</p> <p data-bbox="699 369 1419 453">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.</p> <p data-bbox="699 480 1419 564">A Protected Conversation has been put in doubt, and the input message has been moved to an RRE until the unit of work is aborted or committed.</p> <p data-bbox="699 592 1419 646">This record is logged for each message returned to its original anchor block (SMB or CNT) after QCF has abnormally terminated.</p>
X'39'	QLOGRELO	QLOGRELO	The output queue was freed during cleanup processing of a RELEASE call. (DFSQLOG0)
X'3A'	QLFXFREE	QLFXFREE	A bitmap record was replaced after a queue record was freed at the end of DFSQFIX0 processing. (DFSQFIX0)
X'3B'	QLFXRERR	QLFXRERR	An invalid message record or a nonrecoverable message response was detected during queue validation. (DFSQFIX0)
X'3C'	QLFXBERR	QLFXBERR	A control block was changed during validation by DFSQFIX0. (DFSQFIX0)
X'3D'	QLFXQBLK	QLFXQBLK	A QBLK record was altered during DFSQFIX0 processing. (DFSQFIX0)

Table 8. IMS Log Records Used to Analyze IMS Problems (continued)

Type	Mapping Macro Name	DSECT Name	Why Written (Issuing Module)
X'40'	DFSCHKPT	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.
			<b>X'01'</b> Checkpoint information begins here. (DFSRCP00)
			<b>X'02'</b> Message queue checkpoint record. (DFSQCP00)
			<b>X'03'</b> CNTs or LNTs, or both, follow. (DFSRCP30)
			<b>X'04'</b> SMBs follow. (DFSRCP30)
			<b>X'05'</b> Non-VTAM CTBs follow. (DFSRCP30)
			<b>X'06'</b> DMBs follow. (DFSRCP40)
			<b>X'07'</b> PSB follows. (DFSRCP40)
			<b>X'08'</b> Non-VTAM CLB, LLB, or both, follow. (DFSRCP30)
			<b>X'09'</b> Password table and SMUPs follow. (DFSRCP30)
			<b>X'0A'</b> Password matrix follows. (DFSRCP30)
			<b>X'0B'</b> CTM matrix follows. (DFSRCP30)
			<b>X'0C'</b> CVB follows. (DFSRCP30)
			<b>X'0D'</b> CCBs follow. (DFSRCP30)
			<b>X'0F'</b> Message queues TTR and LCB follow. (DFSRCP30)
			<b>X'10'</b> Non-VTAM CRBs follow. (DFSRCP30)
			<b>X'14'</b> SPQBs and related CNTs follow. (DFSRCP30)
			<b>X'20'</b> Non-VTAM CIBs follow. (DFSRCP30)
			<b>X'21'</b> VTAM VTCBs follow. (DFSRCP30)
			<b>X'22'</b> Subcode for Queue Anchor Block (QAB). (DFS6CKP0)
			<b>X'23'</b> Subcode for LU 6.2 descriptors modified by /CHANGE DESCRIPTOR command. (DFS6CKP0)
			<b>X'24'</b> Subcode for LU 6.2 TIB. (DFS6CKP0)
			<b>X'25'</b> EEQE follows. (DFSTOLG0)
			<b>X'26'</b> I/O toleration buffer follows. (DFSTOLG0)
			<b>X'27'</b> Contains database updates for an in-doubt unit of recovery. (DFSRCP40)
			<b>X'28'</b> Error queue elements (EQEL) for recovery in-doubt structure (RIS). (DFSRCP40)
			<b>X'30'</b> RREs follow. (DFSRCP50)
			<b>X'31'</b> SIDXs follow. (DFSRCP50)
			<b>X'32'</b> TPIPE/YQAB follow. (DFSYCKP0)
			<b>X'33'</b> MTE follow. (DFSYCKP0)
			<b>X'34'</b> TIB follow. (DFSYCKP0)
			<b>X'40'</b> UOWEs follow. (DFSRCP30)
			<b>X'70'</b> MSDB record follows. (DBFHDMPO)
			<b>X'71'</b> ECNT follows. (DBFHDMPO)

Table 8. IMS Log Records Used to Analyze IMS Problems (continued)

Type	Mapping Macro Name	DSECT Name	Why Written (Issuing Module)
X'40' (cont'd)	DFSCHKPT	LOG01	<b>X'72'</b> MSDB header follows. (DBFHDMPO)
			<b>X'73'</b> Page fixed MSDBs follow. (DBFHDMPO)
			<b>X'74'</b> Pageable MSDBs follow. (DBFHDMPO)
			<b>X'79'</b> MSDB record ends. (DBFHDMPO)
			<b>X'80'</b> Fast Path checkpoint information begins here. (DBFCHKP0)
			<b>X'82'</b> EMHB follows. (DBFCHKP0)
			<b>X'83'</b> RCTE follows. (DBFCHKP0)
			<b>X'84'</b> DMCB and DMAC follow. (DBFCHKP0)
			<b>X'85'</b> MTO buffer follows. (DBFCHKP0)
			<b>X'86'</b> DMHR and DEDB buffers follow. (DBFCHKP0)
			<b>X'87'</b> ADSC follows. (DBFCHKP0)
			<b>X'88'</b> Fast Path IEEQEs. (DBFCHKP0)
			<b>X'89'</b> Fast Path checkpoint information ends here. (DBFCHKP0)
			<b>X'98'</b> Checkpoint information ends here. (DFSRCP10)
			<b>X'99'</b> The message queue checkpoint information ends here. (DFSQCP00)
X'41'	DFSLOG41	LOG41DSC	A batch program or BMP program issued a checkpoint. (DFSRDBL0)
X'42'	DFSLOG42	ATLOGREC	IMS switched from one OLDS to another, or a checkpoint was taken, or a shutdown checkpoint was taken. (DFSFDLS0, DFSRDS00, DFSRCP00)
X'43'	DFSLOG43	ADSETLOG	The log manager or the Log Archive utility created this log record. The following subcodes identify each type of record:
			<b>X'01'</b> Record contains status of current online log data set. (DFSFDLS0)
			<b>X'02'</b> 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. (DFSUARPO)

Table 8. IMS Log Records Used to Analyze IMS Problems (continued)

Type	Mapping Macro Name	DSECT Name	Why Written (Issuing Module)
X'45'	DFSLOG45	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).  <b>X'01'</b> Dynamic database log statistics. <b>X'02'</b> Queue buffer statistics. <b>X'03'</b> Format pool statistics. <b>X'04'</b> DL/I buffer pool statistics. <b>X'05'</b> Variable storage pool statistics. <b>X'06'</b> Application scheduling statistics. <b>X'07'</b> Logging statistics. <b>X'08'</b> VSAM buffer pool statistics. <b>X'09'</b> Program isolation statistics. <b>X'10'</b> RCF multi-TCB statistics. <b>X'0A'</b> Latch management statistics. <b>X'0B'</b> Selected dispatcher statistics. <b>X'0C'</b> Storage pool statistics. (DFSCBT00) <b>X'0D'</b> Receive Any (RECA) Buffer statistics. <b>X'0E'</b> Fixed storage pool usage statistics. <b>X'0F'</b> Dispatcher statistics. <b>X'10'</b> RCF Multi-TCB statistics. <b>X'21'</b> IRLM subsystem statistics. (DXRRSTAT) <b>X'22'</b> IRLM system statistics. (DXRRSTAT) <b>X'FF'</b> End of statistics records.
X'47'	DFSLOG47	CAPLOG	A checkpoint was just taken. This log record contains all the PSTs that were in the system. (DFSRCP10)
X'48'	DFSPALOG	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. (DFSFLG0)  <b>X'00'</b> OLDS padding X'48' record. <b>X'01'</b> X'4301' record space holder. <b>X'02'</b> Archived OLDS X'48' record. <b>X'03'</b> Batch SLDS padding X'48' record. <b>X'04'</b> Archived batch SLDS X'48' record.

Table 8. IMS Log Records Used to Analyze IMS Problems (continued)

Type	Mapping Macro Name	DSECT Name	Why Written (Issuing Module)
X'49'	DFSLOG49	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).  <b>X'00'</b> Definition. <b>X'01'</b> Begin stream record. <b>X'02'</b> Begin OFR record. <b>X'03'</b> OFR milestone record. <b>X'04'</b> Log truncation start record. <b>X'05'</b> XRC tracking record. <b>X'06'</b> Data set services create data set record. <b>X'07'</b> Takeover record. <b>X'08'</b> Auto Archive Init Request record. <b>X'0A'</b> Last LSN of prilog record. <b>X'0B'</b> Data set sequence number record. <b>X'0C'</b> Open data set record. <b>X'0D'</b> DBRC hash table state record. <b>X'0E'</b> FF DB Tracker Update Sequence Number (USN). <b>X'20'</b> FP DB Tracker statistics record. <b>X'30'</b> FF DB Tracker FSE Error record. See "X'4930': Database Tracker FSE Error Log Record Format" on page 519 for more information. <b>X'31'</b> FF DB Tracker statistics record. <b>X'50'</b> OFR Stream Processing Time.
X'4C'	DFSLOG4C	STDBLOG	Activity related to database processing, according to these subcodes:  <b>X'01'</b> A backout for token was done. (DFSRBOI0) <b>X'02'</b> A backout error occurred. (DFSRBOI0) <b>X'04'</b> First update flag was reset. (DFSDBDR0) <b>X'08'</b> A share level or held state was changed. (DFSDBAU0, DFSDLOC0) <b>X'10'</b> A write error occurred. (DFSDBH40, DFSDVSM0) <b>X'20'</b> A program was stopped. (DFSRBOI0) <b>X'40'</b> A database was started. (DFSDBDR0) <b>X'80'</b> A database was stopped. (DFSDBDR0) <b>X'82'</b> A database backout failure occurred. (DFSRESP0)
X'4E'	DFSLOG4E	DFSLOG4E	An event occurred during monitoring. This record is in the monitor log and contains statistical information about the system. (DFSMNTR0)



Table 8. IMS Log Records Used to Analyze IMS Problems (continued)

Type	Mapping Macro Name	DSECT Name	Why Written (Issuing Module)
X'50'	DFSDLOG	DLOGDB	<p>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. (DFSRDBLO)</p> <p><b>X'52'</b> 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. (DFSRDBLO)</p>
X'53'	DFSLOG53	SPLLOG	<p>Bitmap write done for log record for alternate IMS tracking CI split on active IMS. (DFSRCHB0, DFSGGSP0, DFSFRSP0, DFSDVSM0)</p>
X'55'	DFSETPCP	DFSETPCP	<p>Record reserved for external subsystem information. (DFSESS30)</p>
X'56'	DBFLGRIM	DBFLGRIM	<p>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 the 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.</p> <p><b>X'000001'</b> IMS began the commit process. (DFSESP10)</p> <p><b>X'000002'</b> IMS finished the commit process. (DFSESP20)</p> <p><b>X'000003'</b> IMS signed on to an external subsystem. (DFSESS00)</p> <p><b>X'000004'</b> IMS created a thread for external subsystem. (DFSESECT0)</p> <p><b>X'000005'</b> IMS resolved a RID. (DFSESI60)</p> <p><b>X'000006'</b> An IMS dependent region abended. (DFSFESP0)</p> <p><b>X'000007'</b> IMS deleted a residual recovery element (RRE) through the /CHA command. (DFSESI70)</p> <p><b>X'000008'</b> IMS deleted a residual recovery element (RRE) by a restart or start command. (DFSIESI0)</p> <p><b>X'000009'</b> An external subsystem disconnected. (DFSESI30)</p> <p><b>X'00000A'</b> Commit found no work to do.</p> <p><b>X'08'</b> A CCTL connected to DBCTL. (DFSDASI0) Mapping macro is DFSETPCP.</p>

Table 8. IMS Log Records Used to Analyze IMS Problems (continued)

Type	Mapping Macro Name	DSECT Name	Why Written (Issuing Module)
X'56' (cont'd)	DBFLGRIM	DBFLGRIM	<b>X'09'</b> A CCTL disconnected from DBCTL. (DFSDASD0) Mapping macro is DFSGTPCP.
			<b>X'10'</b> Phase 1 commit processing started. (DFSDSC00, DFSTMS00)
			<b>X'11'</b> Phase 1 commit processing ended. (DFSDSC00, DFSTMS00)
			<b>X'12'</b> Phase 2 commit processing ended. (DFSDSC00, DFSFXC30, DFSSLOG0, DFSSMSC0, DFSTMS00)
			<b>X'13'</b> Recoverable in-doubt structure (RIS) created. (DFSDRIS0)
			<b>X'14'</b> Recoverable in-doubt structure (RIS) deleted. (DFSDRID0)
			<b>X'15'</b> IMS restarted with RRS. (DFSRRSI0)
			<b>X'16'</b> Interest has been registered with RRS for this UOW. (DFSRRSI0)
			<b>X'37'</b> Phase 2 commit processing started by a resynchronization request. (DFSDRID0)
			<b>X'38'</b> Phase 2 abort processing started by a resynchronization request. (DFSDRID0)
X'57'	DFSDBUR	DFSDBUR	Database updates in an RSR environment:
			<b>X'01'</b> Begin database update. (DFSRDBL0)
			<b>X'02'</b> End database update. (DFSRDBL0)
X'59'	DBFL59X	L59X	<b>X'10'</b> I/O from a data space has started (DBFVXOC0, DBFVOCIO)
			<b>X'12'</b> A group of CIs (control intervals) from a data space has been written to DASD (DBFVXOC0, DBFVOCIO, DBFERS21)
X'59'	DBFLS9FF	L59FF	<b>X'FF'</b> To track internal IMS FP information in various modules.
			<b>X'51'</b> To indicate that nonrecoverable suppression has taken place.
			<b>Mapping Macro</b> This is a Fast Path log record. The following subcodes, contained within the record, precede information in the log record:
X'59'	DBFLGRIM	FLIM	<b>X'01'</b> An input message was received. (DBFSHSP0)
X'59'	DBFLGROM	FLOM	<b>X'03'</b> An output message was sent. (DBFSHSP0)
X'59'	DBFSQRIM	DBFSQRIM	<b>X'11'</b> An input message was inserted on an EMHQ structure. (DBFHIEL0, DBFSYN20)
X'59'	DBFSQROM	DBFSQROM	<b>X'16'</b> An output message was inserted on an EMHQ structure. (DBFATRM0, DBFHCTR0, DBFHCAS0, DBFERMG0, DBFSYN20)
X'59'	DBFBMSDB	MSUPLOG	<b>X'20'</b> An MSDB was updated. (DBFSLOG0, DBFBMSDB)

Table 8. IMS Log Records Used to Analyze IMS Problems (continued)

Type	Mapping Macro Name	DSECT Name	Why Written (Issuing Module)
X'59'	DBFDOCL	DOCL	<b>X'21'</b> DEDB area data set was opened. (DBFMOCL0)
			<b>X'22'</b> DEDB area data set was closed. (DBFMOCL0)
			<b>X'23'</b> DEDB area data set status was changed. (DBFMOCL0)
X'59'	DBFEQE	EQE	<b>X'24'</b> An ADS error queue element (EQE) was created. (DBFMEQE0)
X'59'	DBFLGRDQ	FLDQ	<b>X'36'</b> 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. (DBFHQMIO, DBFHTMG0)
X'59'	DBFLGSYN	SYNC	<b>X'37'</b> A synchronization point operation completed. (DBFSLG20)
			<b>X'38'</b> A synchronization point operation was unsuccessful. (DBFSLG20)
X'59'	DBFLGRIC	HICL5947	<b>X'47'</b> Contains a bit map of CIs that have updates in an HSSP image copy data set. (DBFSLGE1)
X'59'	DBFLSRT	LSRT	<b>X'50'</b> A DEDB was updated—DMAC status log record for DMACOCNT or DMACNXTS. (DBFSLOG0, DBFARDB0, DBFMLOP0)
			<b>X'53'</b> An online utility updated a DEDB. (DBFUMAL0, DBFUMAI0)
			<b>X'54'</b> A log record is created each time an area containing sequential dependent buffers was opened. (DBFMLOG0)
X'59'	DBFLFRSD	FLSD	<b>X'55'</b> A new buffer for sequential dependent segments was obtained. (DBFSYP20)
X'59'	DBFLSRT	LSRT	<b>X'56'</b> Indoubt SDEP buffer from the resynchronization process. (DBFMLOG0) (DBFSYP20)
			<b>X'57'</b> Local/Global portion of DMAC logged. (DBFARDB0, DBFUMAL0)
X'59'	DBFL56X	L56X	<b>X'58'</b> An SDEP buffer was successfully written. (DBFSYP20)
X'59'	DBFLGRRE	FLRE	<b>X'70'</b> The MSDB relocation factor for XRF is shown. (DFSRLP00)
X'5E'	DFSLOG5E	SBLI	Sequential buffer image capture record. A sequential buffer-handler function has been called, according to these subcodes (DFS SBIC0):
			<b>X'00'</b> Application start record.
			<b>X'04'</b> Search/Read.
			<b>X'0C'</b> OSAM buffer-handler crossed a buffer boundary.
			<b>X'18'</b> New logical position.
<b>X'1C'</b> Application stop record.			
X'5F'	DFSLOG5F	DLTRLOGR	A DL/I call was completed. This record contains DL/I call image capture trace data. (DFSDDL0)

Table 8. IMS Log Records Used to Analyze IMS Problems (continued)

Type	Mapping Macro Name	DSECT Name	Why Written (Issuing Module)
X'63'	LOGCSQ	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'	DFSMSREC	SMREC	An inconsistency was found in processing associated with MSC. (DFSCMS00)
X'65'	DFSLOG65	SSREC	A message is about to be enqueued (applicable for System/3 and System/7 only). (DFSCRSV0)
X'66'	LOG3600	SXREC	A message is about to be enqueued or dequeued (applicable for 3614, FINANCE, and SLU P nodes, MSC links, or ISC sessions). (DFSCVFD0, DFSCVFIO, DFSCVFN0, DFSCVLG0, DFSCMSV0, DFSCMSF0)
X'67'	DFSL6701	CTLDESC	<p>This log record is a service trace record (see Figure 52 on page 150 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:</p> <p><b>X'01'</b> There are three situations in which X'6701' is written:</p> <ul style="list-style-type: none"> <li>• 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)</li> <li>• Errors were detected in AOI module DFSAOUE0.</li> <li>• Errors were detected in AOI module DFSAOE00.</li> </ul> <p><b>X'03'</b> A 3270 error was detected. More information about this condition is contained in "Terminal Communication Task Trace" on page 325. (DFSCFEZ0)</p> <p><b>X'04'</b> 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.</p> <p><b>X'06'</b> 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.</p>
X'67'			<p><b>X'05'</b> 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)</p>
X'67'	DFSL6740	DFS6740	<p><b>X'40'</b> 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 COLDCOMM) 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)</p>

Table 8. IMS Log Records Used to Analyze IMS Problems (continued)

Type	Mapping Macro Name	DSECT Name	Why Written (Issuing Module)
X'67'	DFS67D0	DFS67D0	<p><b>X'D0'</b> Indicates the diagnostic record of a failed service request.</p> <p><b>X'01'</b> Failure during a DB DL/I call.</p> <p><b>X'02'</b> Failure during a DC DL/I call. (DFSCPY00, DFSDLA30, DBFHGU10, DFSTMAP0)</p> <p><b>X'03'</b> Failure during a SYS DL/I call.</p> <p><b>X'04'</b> An exit failure occurred. (DFSRRS10)</p> <p><b>X'05'</b> Failure during SPOOL API processing. (DFSIAFP0)</p> <p><b>X'06'</b> Failure during Transaction Manager schedule processing. (DFSTMAS0, DFSTMCD0)</p> <p><b>X'07'</b> Failure during Service Logical Unit Manager (SLUM) processing.</p> <p><b>X'08'</b> Failure during Asynchronous Logical Unit Manager (ALUM) processing.</p> <p><b>X'09'</b> Failure during coupling facility processing. (DFSDCFR0, DFSDMAW0)</p> <p><b>X'0A'</b> Failure during queue manager processing.</p> <p><b>X'0B'</b> Failure during shared queues interface processing. (DBFIPQS0, DFSITQS0, DBFILQS0, DFSILQS0)</p> <p><b>X'0C'</b> Failure during NDM user exit interface processing. (DFSNDMI0)</p> <p><b>X'0D'</b> Failure during shared queues CQSINFRM processing.</p> <p><b>X'0E'</b> Failure during shared queues request processing. (DBFHCAS0, DBFHGU10, DBFHSQS0)</p> <p><b>X'0F'</b> Failure during UOWE resync processing. (DBFHGU10, DBFHCAS0)</p> <p><b>X'10'</b> Shared EMH XCF communication error. (DBFHXCS0)</p> <p><b>X'11'</b> An unsolicited output message was detected. (DBFHSQS0)</p> <p><b>X'12'</b> In-flight input message deleted. (DBFHCAS0)</p> <p><b>X'13'</b> Fast Path Queue Manager Diagnostics. (DBFHQM10)</p> <p><b>X'14'</b> System Termination Diagnostics. (DFSSDA20, DFSTRM00)</p> <p><b>X'15'</b> System Service Error. (DFSOCMD0)</p> <p><b>X'16'</b> Unexpected return or reason code from RM, OM, SCI, or CQS request.</p> <p><b>X'17'</b> Failure during RM update, query, or delete processing.</p>

Table 8. IMS Log Records Used to Analyze IMS Problems (continued)

Type	Mapping Macro Name	DSECT Name	Why Written (Issuing Module)
X'67'	DFSL67FD	SNELDESC	<b>X'ED'</b> Sequential buffering SNAP, created during a periodic evaluation of the sequential buffering process by the SBESNAP option. (DFSSBSN0)
			<b>X'EE'</b> SNAP of a call to the sequential buffering buffer-handler created by the SBSNAP option. (DFSSBSN0)
			<b>X'EF'</b> 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)
			<b>X'FB'</b> 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.
			<b>X'FD'</b> A SNAP call was issued. (DFSERA20)
			<b>X'FF'</b> A pseudoabend or dependent region abnormal termination occurred. Further information of this condition is contained in "SNAP Call facility—DFSERA20L". (DFSERA20)
X'67'	DFSL67FA	DFSTRHD	<b>X'FA'</b> Contains images of the in memory 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'	DFSLOG69	JM	An unauthorized 3275 terminal dialed into a line specified as VERIFY=YES. (DFSDS060)
X'6C'	DFSMSCRC	CMSCREC	MSC partner systems were started. (DFSCMSW0)
X'6D'	DFSLOG6D	SURVLOG	This log record is used in an XRF environment when: <ul style="list-style-type: none"> <li>• XRF surveillance was started or stopped.</li> <li>• A write error occurred on the active subsystem.</li> <li>• The interval or time-out values on the active subsystem were changed by a /CHANGE command. (DFSHIC40, DFSHSRV0, DFSISL60)</li> </ul>
			<b>X'04'</b> 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'	DFSLOG6E	LUMLOG	One of the following SNA commands was processed: QEC, QC, RELQ, RSHUT, SHUTD, SHUTC, LUS. (DFSHCLG0)
X'70'	DFSLOG70	OLCREC	<b>X'00'</b> An online change /MODIFY command sequence completed successfully. The IMS.MODSTAT data set is being updated. (DFSICV80)
			<b>X'01'</b> Allows the XRF primary to signal the alternate that the transaction has been stopped (PSTOP) by module DFSSMSC0. (DFSICV90)

Table 8. IMS Log Records Used to Analyze IMS Problems (continued)

Type	Mapping Macro Name	DSECT Name	Why Written (Issuing Module)
X'72'	DFSLOG72	USRREC	<p>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:</p> <p><b>X'01'</b> ETO user structure dynamically created. Contains the SPQB name and one or more CNTs.</p> <p><b>X'02'</b> ETO user structure dynamically deleted. Contains only the SPQB name.</p> <p><b>X'03'</b> ETO user structure modified. Contains the SPQB name and one or more CNTs.</p> <p><b>X'04'</b> One or more CNTs added to an ETO user structure. Contains the SPQB name and the CNTs that were added.</p>
X'99'	DFSDXBLK	DFSDXBLK	<p>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:</p> <p><b>X'04'</b> Changed data</p> <p><b>X'28'</b> End of job (EOJ)</p> <p><b>X'30'</b> SETS call</p> <p><b>X'34'</b> ROLS call</p> <p>This log record is mapped by the macro, DFSDXBLK, which is not shipped. The log record layouts are explained in <i>IMS Version 9: Customization Guide</i>.</p>

The following topics provide additional information:

- “Format of X'29' Log Record”
- “Format of X'67' Log Record” on page 150

## Format of X'29' Log Record

This topic shows the log record formats for:

- “X'2900': OLR Command Received”
- “X'2910': Ownership Established” on page 142
- “X'2920': UPDATE OLREORG Command” on page 143
- “X'2930': Output Data Set Information” on page 143
- “X'2940': Cursor-Active Status Set” on page 146
- “X'2950': Cursor Movement” on page 147
- “X'2970': Cursor-Active Status Reset” on page 148
- “X'2990': Ownership Relinquished” on page 149

### X'2900': OLR Command Received

A X'2900' log record is written to indicate the receipt of a HALDB Online Reorganization command. Only one X'2900' log record is written for each command. See Table 9 on page 142 for the X'2900' log record layout.

| *Table 9. X'2900' Log Record Layout*

Offset (Hex.)	Length (Dec.)	Field Name	Field Description
00	2	HORLENG	Length of this record, including this length field and the sequence number
02	2	HORRSV1	X'0000' Reserved
04	1	HORTYPE	X'29' Record type
05	1	HORSTYPE	X'00' Record sub-type
06	2	HORRSV2	X'0000' Reserved
08	8	HORRSENM	RSE name or IMS ID
10	16	HOROMCT	OM command token if from a type-2 command, or zeros if from a type-1 command
20	1	HORCTYPE	Command type flags:  "1000 ...." INITIATE "0100 ...." UPDATE "0010 ...." QUERY "0001 ...." TERMINATE
21		HOROCMD	OM command instance block (OCMD) if from a type-2 command, or zeros if from a type-1 command

| **X'2910': Ownership Established**

| Ownership of the online reorganization for a partition was established through DBRC. See Table 10 for the X'2910' log record layout.

| *Table 10. X'2910' Log Record Layout*

Offset (Hex.)	Length (Dec.)	Field Name	Field Description
00	2	HORLENG	Length of this record, including this length field and the sequence number
02	2	HORRSV1	X'0000' Reserved
04	1	HORTYPE	X'29' Record type
05	1	HORSTYPE	X'10' Record sub-type
06	2	HORPSTNO	PST number
08	8	HORRSENM	RSE name or IMS ID
10	8	HORDBD	DBD name
18	8	HORPSB	PSB name
18	1	HORPSB0	C'0'
19	7	HORPART	Partition name



Table 10. X'2910' Log Record Layout (continued)

Offset (Hex.)	Length (Dec.)	Field Name	Field Description
20	1	HOROFLG1	Flags:  "10.. ...." The INITIATE OLREORG command had the NODEL option. "01.. ...." The INITIATE OLREORG command had, or defaulted to, the DEL option. "..1. ...." The INITIATE OLREORG command had the RATE option. "...0 ...." A new reorganization was started. "...1 ...." The reorganization was restarted. ".... .0.." The database is not RSR covered. ".... .1.." The database is RSR covered.
21	1	HORORATEV	RATE value (1 through 100 percent)

### X'2920': UPDATE OLREORG Command

The UPDATE OLREORG command was processed. The X'2920' log record is written once for each HALDB partition affected by the UPDATE OLREORG command. See Table 11 for the X'2920' log record layout.

Table 11. X'2920' Log Record Layout

Offset (Hex.)	Length (Dec.)	Field Name	Field Description
00	2	HORLENG	Length of this record, including this length field and the sequence number.
02	2	HORRSV1	X'0000' Reserved.
04	1	HORTYPE	X'29' Record type.
05	1	HORSTYPE	X'20' Record sub-type.
06	2	HORPSTNO	PST number.
08	8	HORRSENM	RSE name or IMS ID.
10	8	HORDBD	DBD name.
18	8	HORPSB	PSB name.
18	1	HORPSB0	C'0'
19	7	HORPART	Partition name
20	1	HORUFLG1	Flags:  "10.. ...." The NODEL option is now in effect. "01.. ...." The DEL option is now in effect.
21	1	HORORATEV	RATE value (1 through 100 percent) that is now in effect.

### X'2930': Output Data Set Information

The output data sets have been successfully validated or created. This X'2930' log record contains various characteristics of all of the output data sets, both those that were preexisting and those that were automatically created. There is enough information to recreate any of these output data sets. See Table 12 on page 144 for the X'2930' log record layout.

Table 12. X'2930' Log Record Layout

Offset (Hex.)	Length (Dec.)	Field Name	Field Description
00	2	HORLENG	Length of this record, including this length field and the sequence number.
02	2	HORRSV1	X'0000' Reserved.
04	1	HORTYPE	X'29' Record type.
05	1	HORSTYPE	X'30' Record sub-type.
06	2	HORPSTNO	PST number.
08	8	HORRSENM	RSE name or IMS ID.
10	8	HORDBD	DBD name.
18	8	HORPSB	PSB name.
18	1	HORPSB0	C'0'
19	7	HORPART	Partition name
20	4	HORDUSN	Update sequence number (USN).
24	4	HORDUSID	Update set ID (USID).
28	1	HORDFLG1	Flags: ". . . . 0 . ." The database is not RSR covered. ". . . . 1 . ." The database is RSR covered. ". . . . . 0 ." The A-thru-J and X data sets are the output data sets. ". . . . . 1 ." The M-thru-V and Y data sets are the output data sets. ". . . . . 0 " PHDAM database. ". . . . . 1 " PHIDAM database.
29	1	HORDDSECT	Number of following entries.
2A		HORDDSE	The group of fields shown in Table 13 is repeated for each output data set. There are two entries for the primary index data set of a PHIDAM database.

The group of fields shown in Table 13 is repeated for each output data set. There are two entries for the primary index data set of a PHIDAM database.

Table 13. X'2930' Log Record Layout — Repeated Data Set Fields

Offset (Hex.)	Length (Dec.)	Field Name	Field Description
00	2	HORDENTL	Entry length, including this length field.
02	1	HORDDCBN	DCB number, with A-thru-J,X or M-thru-V,Y indicator: "0 . . . . ." One of the A-thru-J or X data sets. "1 . . . . ." One of the M-thru-V or Y data sets. ". . . . nnnn" DCB number.
03	8	HORDDDNAM	The DD name used for allocation.

Table 13. X'2930' Log Record Layout — Repeated Data Set Fields (continued)

Offset (Hex.)	Length (Dec.)	Field Name	Field Description
0B	1	HORDDFL1	Data set flags:
			"0... .." OSAM data set.
			"1... .." VSAM data set.
			".0.. .." Data set existed before INITIATE OLREORG command.
			".1.. .." Data set was created automatically.
			"...0. ...." Non-SMS-managed data set.
			"...1. ...." SMS-managed data set.
			"1..0 ...." VSAM ESDS.
			"1..1 0..." VSAM KSDS data component (DCB number X'05' or X'85').
			"1..1 1..." VSAM KSDS index component (DCB number X'04' or X'84').
			"1..1 11.." For VSAM KSDS index component, REPLICATE. Replicate index records.
			"..... .0.." NOREPLICATE. Don't replicate index records (or replication not applicable).
0C	1	HORDDFL2	Data set space allocation type flags:
			"1000 ...." Primary amount: number of VSAM records or OSAM blocks.
			"0100 ...." Primary amount: number of bytes.
			"0010 ...." Primary amount: number of kilobytes.
			"0001 ...." Primary amount: number of megabytes.
			"..... 1000" Secondary amount: number of VSAM records or OSAM blocks.
			"..... 0100" Secondary amount: number of bytes.
			"..... 0010" Secondary amount: number of kilobytes.
			"..... 0001" Secondary amount: number of megabytes.
0D	4	HORDRCSZ	VSAM record size or OSAM block size.
11	4	HORDCISZ	For VSAM, control interval size. For OSAM: 0.
15	8	HORDDATC	For SMS-managed, data class if present. Otherwise, blanks.
1D	8	HORDSTGC	For SMS-managed, storage class. Otherwise, blanks.
25	8	HORDMGTC	For SMS-managed, management class if present. Otherwise, blanks.
2D	4	HORDPRIA	Primary allocation amount. See HORDDFL2.
31	4	HORDSECA	Secondary allocation amount. See HORDDFL2.
35	1	HORDFSCI	For VSAM KSDS data component, freespace percentage in each control interval.
36	1	HORDFSCA	For VSAM KSDS data component, freespace percentage in each control area.
37	1	HORDKYLN	For VSAM KSDS data component, key length.
38	2	HORDKYOF	For VSAM KSDS data component, key offset.

Table 13. X'2930' Log Record Layout — Repeated Data Set Fields (continued)

Offset (Hex.)	Length (Dec.)	Field Name	Field Description
3A	1	HORDSHRR	For VSAM, SHAREOPTIONS value, cross-region.
3B	1	HORDSHRS	For VSAM, SHAREOPTIONS value, cross-system.
3C	1	HORDVOLR	Number of existing or requested DASD volumes.
3D	1	HORDVOLC	Number of volume serial numbers following.
3E	<i>n</i>	HORDVOLS	Volume serial numbers.
3E	6	HORDVOL	Volume serial number (repeated). HORDVOLC contains the number of these volume serial number entries.

### X'2940': Cursor-Active Status Set

The cursor is active. Initialization of the reorganization of the partition was completed successfully, two sets of data sets exist, and copying is about to begin. The reorganization was recorded through DBRC as being in a cursor-active status. See Table 14 for X'2940' log record layout.

Table 14. X'2940' Log Record Layout

Offset (Hex.)	Length (Dec.)	Field Name	Field Description
00	2	HORLENG	Length of this record, including this length field and the sequence number
02	2	HORRSV1	X'0000' Reserved
04	1	HORTYPE	X'29' Record type
05	1	HORSTYPE	X'40' Record sub-type
06	2	HORPSTNO	PST number
08	8	HORRENM	RSE name or IMS ID
10	8	HORDBD	DBD name
18	8	HORPSB	PSB name
18	1	HORPSB0	C'0'
19	7	HORPART	Partition name
20	4	HORDUSN	Update sequence number (USN)
24	4	HORDUSID	Update set ID (USID)
28	1	HORAFLG1	Flags:
		".... 0..."	This log record was created at the IMS doing the HALDB Online Reorganization.
		".... 1..."	This log record was created by an IMS doing data sharing with the IMS doing the HALDB Online Reorganization.
		".... .0..."	The database is not RSR covered.
		".... .1..."	The database is RSR covered.
		".... ..0..."	The A-thru-J and X data sets are the input data sets.
		".... ..1..."	The M-thru-V and Y data sets are the input data sets.
		".... ...0"	PHDAM database.
		".... ...1"	PHIDAM database.

Table 14. X'2940' Log Record Layout (continued)

Offset (Hex.)	Length (Dec.)	Field Name	Field Description
29	12	HORARATIM	Time stamp of reorganization active. This is the time of the DBRC ALLOC for the first output data set.

### X'2950': Cursor Movement

The cursor was updated. The X'2950' log record appears before the X'3730' log record that indicates that a unit of reorganization was committed. See Table 15 for X'2950' log record layout.

Table 15. X'2950' Log Record Layout

Offset (Hex.)	Length (Dec.)	Field Name	Field Description
00	2	HORLENG	Length of this record, including this length field and the sequence number
02	2	HORRSV1	X'0000' Reserved
04	1	HORTYPE	X'29' Record type
05	1	HORSTYPE	X'50' Record sub-type
06	2	HORPSTNO	PST number
08	8	HORRENM	RSE name or IMS ID
10	8	HORDBD	DBD name
18	8	HORPSB	PSB name
18	1	HORPSB0	C'0'
19	7	HORPART	Partition name
20	8	HORMUORS	Segments moved in this UOR
28	8	HORMUORZ	Size moved in this UOR
30	8	HORMRSEG	Total segments moved before this UOR
38	8	HORMRSEZ	Total size moved before this UOR
40	4	HORMORSA	Number of roots moved this UOR
44	4	HORMLOCK	Lock count for this OLR
48	4	HORMSTT	UOR start time, in unsigned binary format
4C	12	HORMUTST	UOR start time, in UTC format
58	4	HORMTIME	Execution time
5C	4	HORMWAIT	Wait time
60	4	HORMORSZ	UOR size calculated
64	4	HORMORW1	Not used
68	4	HORMORW2	Not used
6C	8	HORMCHNG	DFSORP20 CHANGEID

Table 15. X'2950' Log Record Layout (continued)

Offset (Hex.)	Length (Dec.)	Field Name	Field Description
74	1	HORMFLG1	Flags: ".... ..0." The A-thru-J or X data sets are the input data sets ".... ..1." The M-thru-V or Y data sets are the input data sets ".... ...0" PHDAM database ".... ...1" PHIDAM database
75	4	HORMRBA	PHDAM cursor RBA
75	1	HORMKLN	Length of root key for PHIDAM
76	<i>n</i>	HORMKEY	PHIDAM cursor root key. The length <i>n</i> is the length of the root key.

**X'2970': Cursor-Active Status Reset**

The cursor is inactive. Copying from the input to the output data sets has completed. The reorganization was recorded through DBRC as no longer being in a cursor-active status. See Table 16 for X'2970' log record layout.

Table 16. X'2970' Log Record Layout

Offset (Hex.)	Length (Dec.)	Field Name	Field Description
00	2	HORLENG	Length of this record, including this length field and the sequence number
02	2	HORRSV1	X'0000' Reserved
04	1	HORATYPE	X'29' Record type
05	1	HORSTYPE	X'70' Record sub-type
06	2	HORPSTNO	PST number
08	8	HORRSENM	RSE name or IMS ID
10	8	HORDBD	DBD name
18	8	HORPSB	PSB name
18	1	HORPSB0	C'0'
19	7	HORPART	Partition name
20	4	HOREUSN	Update sequence number (USN)
24	4	HOREUSID	Update set ID (USID)

Table 16. X'2970' Log Record Layout (continued)

Offset (Hex.)	Length (Dec.)	Field Name	Field Description
28	1	HOREFLG1	Flags:
			"10.. ...." The NODEL option is now in effect.
			"01.. ...." The DEL option is now in effect.
			".... 0..." This log record was created at the IMS doing the HALDB Online Reorganization.
			".... 1..." This log record was created by an IMS doing data sharing with the IMS doing the HALDB Online Reorganization.
			".... .0..." The database is not RSR covered.
			".... .1..." The database is RSR covered.
			".... ..0." The A-thru-J or X data sets are the input data sets.
			".... ..1." The M-thru-V or Y data sets are the input data sets.
29	12	HORCITIM	Time stamp of cursor inactive

### X'2990': Ownership Relinquished

Ownership of the reorganization for a partition was relinquished through DBRC. See Table 17 for X'2990' log record layout.

Table 17. X'2990' Log Record Layout

Offset (Hex.)	Length (Dec.)	Field Name	Field Description
00	2	HORLENG	Length of this record, including this length field and the sequence number
02	2	HORRSV1	X'0000' Reserved
04	1	HORTYPE	X'29' Record type
05	1	HORSTYPE	X'90' Record sub-type
06	2	HORPSTNO	PST number
08	8	HORRSENM	RSE name or IMS ID
10	8	HORDBD	DBD name
18	8	HORPSB	PSB name
18	1	HORPSB0	C'0'
19	7	HORPART	Partition name
22	1	HORREAS	Reason for relinquishing ownership:
			"100. ...." Normal completion of the reorganization.
			"010. ...." Pseudo-abend, with code in HORABTRM.
			"001. ...." TERMINATE OLREORG command.
21	4	HORABTRM	Pseudo-abend code
25	4	HORSEGCT	Number of segments copied

## Format of X'67' Log Record

Figure 52 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.

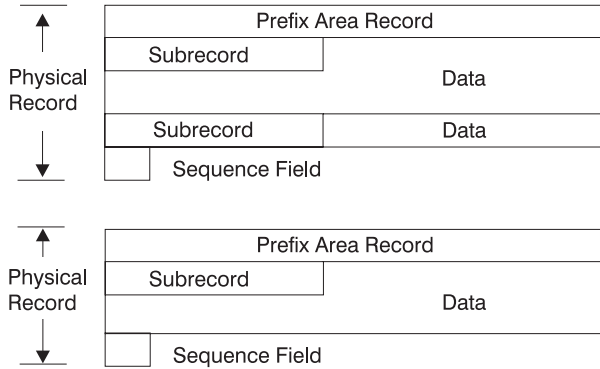


Figure 52. Log Record Layout

### Log Record Prefix Area

The format of the X'67FA', X'67FB', X'67FD', and X'67FF' records are shown in Figure 53 and Table 18.. All other X'67' records have individual differences.

**Log Record Prefix Area Layout:** Figure 53 shows the log record prefix area layout.

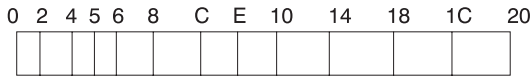


Figure 53. Log Record Prefix Area Layout

### Log Record Prefix Area Format:

Table 18. 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 19.

Table 19. 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

Figure 54 shows the log subrecord and data area. Table 20 shows the log subrecord area format.

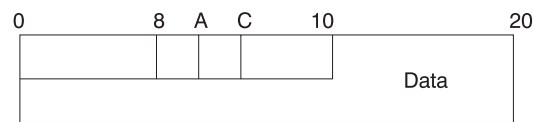


Figure 54. Log Subrecord and Data Area Layout

### Log Subrecord Area Format

Table 20. 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 21. Log Data Area Format

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

## Log Sequence Field

### Log Sequence Field Layout

Figure 55 shows the log sequence field layout and Table 22 on page 152 shows the log sequence field format.

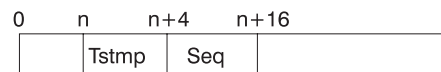


Figure 55. Log Sequence Field Layout

## Log Sequence Field Format

Table 22. 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 and the externalized trace table entries recorded in the DFSTRAXX data set. Formatting of the DFSTRAXX trace entries is similar to formatting trace records contained on the IMS log, however, the external trace data set will only contain records with an ID of X'67FA'. For more information on the DFSTRAXX data sets (DFSTRA01 and DFSTRA02), see the section titled "Data Sets" in the *IMS Version 9: Installation Volume 1: Installation Verification*.

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.

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

The File Select and Formatting Print utility (DFSERA10) is also available using the Knowledge-Based Log Analysis (KBLA) panel interface, option 2.3. For more information on KBLA and the File Select and Formatting Print utility (DFSERA10) see the *IMS Version 9: Utilities Reference: System*.

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.

DFSERA70 also allows the use of the DATA= parm for all record types. This parameter allows the user to select all records that contain the data substring specified on the DATA= parm.

For a detailed description of each exit, see *IMS Version 9: Utilities Reference: System*.



---

## Log Merge Utility

The Log Merge utility (DFSLTMG0) produces one data set that is used as input to the Log Transaction Analysis utility by merging the system log data sets (SLDS) from two or more IMS systems.

The Log Merge utility can merge up to nine IMS system logs. Each log is the output of a uniquely identified IMS system running during the same time span. The order of input to the Log Merge utility is LOG01, LOG02, LOG03, ..., LOG09.

DFSLTMG0 is placed in the IMS.SDFSRESL data set during IMS system definition.

The Log Merge Utility is also available using the Knowledge-Based Log Analysis (KBLA) panel interface, option 1.4. For more information on KBLA, see the section titled “Knowledge-Based Log Analysis” in the *IMS Version 9: Utilities Reference: System*.

**Related Reading:** For more information on the Log Merge Utility, see *IMS Version 9: Installation Volume 2: System Definition and Tailoring*.

---

## Formatting IMS Dumps Offline

This topic 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 183.

The topics include:

- “Introduction to the Offline Dump Formatter”
- “Solving IMS Problems with the Dump Formatter” on page 155
- “Using the Formatted Dump” on page 170

## 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 z/OS 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 155 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 a z/OS 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 156 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 9: Installation Volume 2: System Definition and Tailoring* for more information.

You cannot use the ODF to format z/OS 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 z/OS DUMP command usually produce acceptable input data sets.

For details of the SDUMP support job stream, refer to *IMS Version 9: 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
- 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 9: Utilities Reference: System* for more information on invoking the IMS Offline Dump Formatter.

## **Solving IMS Problems with the Dump Formatter**

This topic outlines how you can use the ODF to help solve IMS problems. The topics “Choosing FMTIMS Parameters” on page 156 and “Sample FMTIMS Statements” on page 158 list the FMTIMS options you could choose for particular problem areas. “Contents Formatted for FMTIMS Options” on page 161 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 158 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 z/OS 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 160 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 job name 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 23 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 161.

To use Table 23, 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 23. FMTIMS Parameters for General Problems

Parameters	Problem Area							
	Checkpoint/ Restart	DB	DC	FP	Log	System/ Other	Batch	CICS
CBT		X	X			X	X	X
CBTE			X					

Table 23. FMTIMS Parameters for General Problems (continued)

Parameters	Problem Area							
	Checkpt/ Restart	DB	DC	FP	Log	System/ Other	Batch	CICS
DB		X					X	X
DBRC		X				X	X	X
DC			X				<sup>2</sup>	
DEDB		X		X				
DISPATCH	X	X	X	X	X	X	<sup>3</sup>	
EMH		X	X	X				
LOG					X		X	
MSDB		X		X				
QM			X				<sup>2</sup>	
RESTART	X						<sup>2</sup>	
SAP			X					
SAVEAREA <sup>1</sup>	X	X	X	X	X	X	<sup>2</sup>	
SB		X				X	X	X
SCD <sup>1</sup>	X	X	X	X	X	X	X	X
SPST	X			X			<sup>2</sup>	
SUBS						X	<sup>2</sup>	
SUMMARY <sup>1</sup>	X	X	X	X	X	X	X	X
UTIL			X	X			<sup>2</sup>	

**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 161 for a list of the modules formatted with each of the parameters. See “Syntax Restrictions on the FMTIMS Statement” on page 160 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 170) as a guide. Unless one or both of these fields specify “unknown”, they should indicate why a dump was taken.

Table 24 shows the options you could choose based on valid CALLER= and TCB= information in the dump title.

Table 24. FMTIMS Parameters Based on CALLER= and TCB= Fields

CALLER=	TCB=	Recommended FMTIMS Options <sup>1</sup>
CTL	CTL LOG ESS LSD LSM RDS RST STC STM	DC <sup>2</sup> , Dispatch <sup>2</sup> , QM <sup>2</sup> , Summary, System <sup>2</sup> Dispatch <sup>2</sup> , SPST, System <sup>2</sup> , SUBS, Summary Dispatch, Log, Restart, Summary, System Dispatch <sup>2</sup> , MSDB, Savearea, SCD <sup>2</sup> , Summary Dispatch <sup>2</sup> , MSDB, Savearea, SCD <sup>2</sup> , Summary Restart, Savearea, SCD <sup>2</sup> , Summary Restart, Savearea, SCD <sup>2</sup> , Summary CBT, Dispatch <sup>2</sup> , Savearea, SCD <sup>2</sup> , Summary CBT, Dispatch <sup>2</sup> , Savearea, SCD <sup>2</sup> , Summary

Table 24. FMTIMS Parameters Based on CALLER= and TCB= Fields (continued)

CALLER=	TCB=	Recommended FMTIMS Options <sup>1</sup>
CURR <sup>3</sup>	DYA	Dispatch <sup>2</sup> , System <sup>2</sup>
DBRC	DBR	DBRC <sup>2</sup> , System <sup>2</sup>
DL/I	DLI STC	DB <sup>2</sup> , Dispatch <sup>2</sup> , SB <sup>2</sup> , System <sup>2</sup> CBT, Dispatch <sup>2</sup> , Savearea, SCD <sup>2</sup> , Summary
DP	BMP DEP	DB <sup>2</sup> , System <sup>2</sup> DB <sup>2</sup> , System <sup>2</sup>
FP	BMP DEP <sup>4</sup> XFP	DB <sup>2</sup> , DEDB, MSDB, System <sup>2</sup> DB <sup>2</sup> , DEDB, MSDB, System <sup>2</sup> DB <sup>2</sup> , SPST, System <sup>2</sup>
LOG	LOG	Log <sup>2</sup> , System <sup>2</sup>

**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 z/OS 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 job name 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 170.) 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.



**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 *z/OS MVS Diagnosis: Tools and Service Aids* if you need a z/OS 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 173. See “Table of Control Block Definitions” on page 67 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. Table 25 shows the areas formatted under the (DB) and (DB,MIN) FMTIMS options.

Table 25. 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 26. This option is skipped if the CTL address space is not included in the dump data set.

Table 26. Data Communication Areas Formatted by DC and DC,MIN

(DC)	(DC,MIN) <sup>1</sup>
All CLBs, LXBs, and LCBs, with subordinate control blocks: <ul style="list-style-type: none"> <li>• Current CTB or LTB, and CNT</li> <li>• Allocated I/O buffers</li> <li>• CIB, if using MFS processing</li> <li>• CCB, if using conversational processing</li> <li>• MFS work buffers</li> <li>• ECNT, EMHB, and message buffer, if the CTB shows a Fast Path terminal</li> </ul>	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 27.

Table 27. 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 28 on page 164 shows the areas formatted under this FMTIMS option.

Table 28. 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:

- job name
- 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 29. The CTL address space must be included in the dump data set for this option to be formatted.

Table 29. 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 30. These areas, except for the WADS and the DLOG trace, are repeated in the dump when the IMS Monitor is active.

Table 30. 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 31.

Table 31. 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 32.

Table 32. 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 33 shows which sections are formatted with the SB and SB,MIN options. A description of the sections follows Table 33.

*Table 33. Sections Formatted by SB and SB,MIN*

<b>(SB)</b>	<b>(SB,MIN)</b>
Subsystem overview	same
PST overview <sup>1</sup>	same <sup>2</sup>
Sorted blocks <sup>1</sup>	same <sup>2</sup>
Sorted buffers <sup>1</sup>	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 34.

*Table 34. Areas Formatted by SCD and SCD,MIN*

<b>(SCD)</b>	<b>(SCD,MIN)</b>
SCD	same
Latch Extensions	same
Scheduler Sequence Queues	not formatted
Synchronous APPC/OTMA Shared Message Queue SCD Extension	same
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
- IMSs 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 2-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 topic 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 topic.

### 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 five 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.
- jjjjjjj* Job name.
- sss* System abend code.
- uuuu* User abend code.
- rrr* Optional user abend reason code.
- ddd* Julian day of year.
- ttttt* 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.
- jjjjjjj* Job name.
- ddd* Julian day of year.
- ttttt* Time, in the form HHMMSS.
- cccc* DFSDUMP caller parameter or blanks.

- xxx* Abend 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=rrrrrrrrrrxxxxxxxxxxxxxxxx

where:

*l* Length of title in hexadecimal - here X'5D'.

*jjjjjjj* DBCTL job name.

*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:

**Hex** System abend.

**Decimal**

User abend.

*mmm...m*

Message text (up to 40 characters) that describes the error. See the possible error messages 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. It 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.

- DFSPRA00,** DBCTL FAILURE DURING DRA TERM
- DFSPRA10,** DBCTL FAILURE DURING IDENTIFY
- DFSPRA20,** DBCTL FAILURE DURING RESYNC
- DFSPRA50,** DBCTL FAILURE DURING PURGE
- DFSPINIO,** FAILURE ESTABLISHING ESTAE
- DFSPAT00,** GETMAIN FAILURE
- DFSPINIO,** SSI FAILURE DURING SONCRT
- DFSPINIO,** 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  
**DFSPPTH0,** SSI FAILURE DURING TERMTHD  
**DFSPPTH0,** DBCTL FAILURE DURING TERMTHD  
**DFSPRA40,** PQE CANNOT BE PROCESSED  
**DFSPRA0,** PQE OR PAPL IS INVALID  
**DFSFRA0,** CONTROL TCB ESTAE INVOKED  
**DFSPAT0,** THREAD TCB ESTAE INVOKED  
**DFSFRA0,** DRA ESTAE FAILED TO ESTABLISH ESTAE  
**NO OTHER DRA MESSAGE**

**Title Format 4:** This dump is created by DFSERA20 when a SNAP dump is requested. The format is generated for some pseudoabend SNAP dumps which were taken to the logs in releases prior to IMS Version 9. The title is the format:

`nnnnnnnn IMS USER ABEND uuuu, DATE.TIME: ddd.tttttt,i`

where:

`nnnnnnnn`

IMS name.

`uuuu` The user abend code or UNK if a SNAP was requested, but there was no abend set.

`ddd` Julian day of year.

`ttttt` Time, in the form HHMMSS

`i` Indicator if primary (P) or secondary (S) requested.

**Title Format 5:** This dump is created by DFSERA20 when a SNAP dump is requested. The format is generated for dumps that is taken when an unexpected DL/I status code is returned during HALDB Online Reorganization. The title is the format:

`nnnnnnnn UNEXPECTED STATUS CODE cc, DATE.TIME: ddd.tttttt,i`

where:

`nnnnnnnn`

IMS name.

`cc` The unexpected status code returned during HALDB Online Reorganization.

`ddd` Julian day of year.

`ttttt` Time in the form HHMMSS.

`i` Indicator if primary (P) or secondary (S) requested.

`cc` is the unexpected status code returned during HALDB Online Reorganization.

### 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 z/OS 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 9: Messages and Codes, Volume 1* and *IMS Version 9: Messages and Codes, Volume 2*.

### 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.

**Synchronous APPC/OTMA Shared Message Queue SCD Extension**

The mapping DSECT is DFSCSCD.

**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 166 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 Params 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 167 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 <sup>1</sup>****Fast Path MDSB Formatting <sup>1</sup>****Communication Line Blocks and Subordinate Blocks <sup>1</sup>**

For each CLB line, all the control blocks associated with that line are formatted.

**CTB <sup>1</sup>**

The mapping DSECT is ICLI CTBBASE=0.

**Input Buffer <sup>1</sup>**

A SNAP of the input buffer, if input is active.

**Output Buffer <sup>1</sup>**

A SNAP of the output buffer, if output is active.

**CCB <sup>1</sup>**

Present if a conversation is active or held. The mapping DSECT is ICLI CCBASE=0.

**CIB <sup>1</sup>**

Present if MFS is in use. The mapping DSECT is ICLI CIBBASE=0.

**Communication Terminal Table <sup>1</sup>**

Defines terminal characteristics. The mapping DSECT is ICLI CTTBASE=0.

**SPQB Entries <sup>1</sup>**

Entries on the subpool queue block chain. Unallocated CNTs are also formatted here.

---

1. These areas are not dumped in a DBCTL environment.

**SMB Table <sup>1</sup>**

This table defines transaction characteristics in the IMS system. The mapping DSECT is IAPS SMBBASE=0.

**Queue Manager Pool Prefix and Buffers <sup>2</sup>**

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**

**DBRC Work Areas**

**LUM Trace**

Allows LU 6.2 activities to be analyzed with the MVS/ESA APPC trace entries by the LU manager.

**Edited Command Buffer 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 58 shows the layout of the edited command.

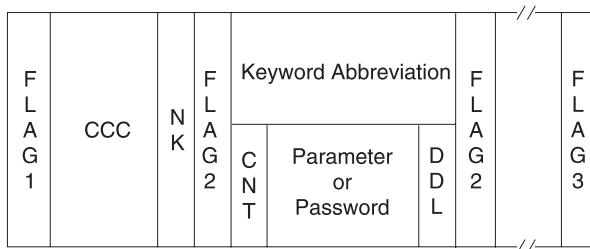


Figure 58. Edited Command Layout

**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.

2. These areas are not dumped in a DBCTL environment.

**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.

## 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 IMS Connect and other IMS component address spaces, such as CQS, OM, RM, and SCI
- An entry for formatting other IMS-related products, such as database recovery service, and their associated BPEs

The following topics provide additional information:

- "Using Interactive Dump Formatter Menus" on page 179

- “Using the Other IMS Components Formatting Panels” on page 182
- “Using the Other IMS-Related Products Formatting Panels” on page 182
- “IMS IPCS Symbols” on page 182
- “Using IMS Enhanced Dump Analysis” on page 183

## 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 59 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 59. IMS Dump Formatting Primary Menu Panel

3. If this is the first time you are reading the dump, select 0 (Initialization). The panel in Figure 60 appears.

```

DFSAAE10 ----- 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 60. IMS Dump Formatting Initialization/Content Panel - Inactive

4. Enter the IMS job name in the row marked CTL, or the DL/I job name in the row marked DL/I, and press enter. Either job name 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 61. 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          NA       0019     N/A
DBRC    DTSDBRCA      N/A      001A     YES
IRLM    N/A          N/A      N/A      N/A
TMS

ABEND CODE = SYS  0C4      USER  0
MODULE     = DFSSCBT0

IMS SDWA ADDRESS - 007BC680   IMS RELEASE - 810
IMS SCD  ADDRESS - 00BA1E30
ABENDED ASID  - 0019
    
```

Figure 61. IMS 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 62 on page 181 and Figure 63 on page 181.

```

----- 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 62. IMS 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.

```

DFSAALLO ----- 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
-----
-  CLB       ADDRESS  -vvvvvvv  CLB/LLB address (hexadecimal)
-  CLB       NODE     CLB/LLB address (hexadecimal)
-  CLB       LTERM    VTAM node name
-  CLB       CID      IMS logical terminal name (CNT)
-  CLB       LINE     VTAM communication ID (hexadecimal)
-  LLB       LINK     BTAM line number (decimal)
-  DPST      ADDRESS  MSC link number (decimal)
-  DPST      NUMBER   Dependent region PST address (hexadecimal)
-  DPST      NAME     Dependent region PST number (hexadecimal)
-  SYSPST    ADDRESS  Dependent region PST jobname
-  SYSPST    NAME     System PST address
-  TRACE     NAME     *P* System PST name
-  SAP       ADDRESS  *P* Trace table ID (2 characters)
-  SAP       ECBADR   Savearea block address (hexadecimal)
-  POOL      NAME     SAP's ECB address (hexadecimal)
-  CBTE      NAME     *P* IMS storage pool name
-  LUB       NAME     Control Block Table name
-  LUB       NAME     LU name
  
```

Figure 63. IMS Low-Level Dump Formatting Selection Panel

```

DFSAALAO ----- 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 64. IMS Analysis Selection Panel

## Using the Other IMS Components Formatting Panels

Some IMS components (for example, the Common Queue Server (CQS), the Operations Manager (OM), the Resource Manager (RM), and the Structured Call Interface (SCI)) 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 through the BPE initialization and status panel under option 6, not by option 0 on the primary menu.

- | For information on using the IMS Connect Dump Formatter (an option under “Other IMS components”),
- | see the “IMS Connect Dump Formatter” section in the *IMS Version 9: Installation Volume 2: System*
- | *Definition and Tailoring*.

## 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. The dump formatter also sets symbols for the registers (R0-R15) and PSW (DFSPSW) at abend for abend dumps. This allows you to quickly locate areas in storage pointed to by the registers and PSW when you are in IPCS browse mode.

IMS creates and lists the IPCS symbols when the job name of an address space using BPE is supplied in the BPE initialization panel (for example, a CQS, OM, RM, or SCI address space).



## 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 65.

```

----- 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 65. IMS 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 66.

```

----- 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 66. Sample Filtering Panel

When you open the generic filtering panel, default values are automatically filled in, as shown in Figure 66; 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 67.

```

<===== AND/OR (A/O)          QUAL =====>
    
```

Figure 67. 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 173 for eye-catcher examples.

**Exception:** Address spaces using BPE (for example, CQS, OM, RM, and SCI) do not provide any online dump formatting output.

- | The following topics provide additional information:
- | • “Formatted Dump for the CTL Address Space”
- | • “Formatted Dump for the DL/I Address Space” on page 187

## 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 187. 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 175 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 167 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**<sup>3</sup>

For each CLB (line), all the control blocks associated with that line are formatted.

**CLB**<sup>3</sup>

The mapping DSECT is ICLI CLBBASE=0.

**CTB**<sup>3</sup>

The mapping DSECT is ICLI CTBBASE=0.

**Input Buffer**<sup>3</sup>

A SNAP of the input buffer, if input is active.

**Output Buffer**<sup>3</sup>

A SNAP of the output buffer, if output is active.

**CCB**<sup>3</sup>

Present if a conversation is active or held. The mapping DSECT is ICLI CCBBASE=0.

**CIB**<sup>3</sup>

Present if MFS is in use. The mapping DSECT is ICLI CIBBASE=0.

**Communication Terminal Table**<sup>3</sup>

Defines terminal characteristics. The mapping DSECT is ICLI CTTBASE=0.

**SPQB Entries**<sup>3</sup>

Entries on the subpool queue block chain. Unallocated CNTs are also formatted here.

### **SMB Table**<sup>3</sup>

This table defines transaction characteristics in the IMS system. The mapping DSECT is IAPS  
SMBBASE=0.

### **Queue Manager Pool Prefix and Buffers**<sup>3</sup>

The mapping DSECTs are ICLI POOLBASE=0 and ICLI BFRBASE=0.

### **Buffer Prefix List**<sup>3</sup>

Contains the address of each buffer's prefix, status byte, and first and last pending and current DRRN.

### **QPOOL Prefix**<sup>3</sup>

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.

### **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.

---

3. These areas are not dumped in a DBCTL environment.

**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:

<b>BFSP</b>	Formats the buffer pool prefix. The mapping DSECT is BFSP.
<b>BFUS</b>	Formats the subpool prefix. The mapping DSECT is BFUS.
<b>RPLI</b>	Formats the DL/I RPL block. The mapping DSECT is RPLI.
<b>DL/I Data</b>	A dump of the DL/I, lock activity and program isolation trace table. The mapping DSECT is IDLIVSAM TRACENT.
<b>Lock Activity Trace Data</b>	See DL/I DATA.
<b>Program Isolation Data</b>	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:

<b>MAINPOOL</b>	Formats the main pool header. The mapping DSECT is IBPOOL.
<b>SUBPOOL</b>	Formats the subpool header. The mapping DSECT is ISUBPL.
<b>Buffer Prefix</b>	Formats the buffer prefix. The mapping DSECT is IBFPRF.
<b>Buffer</b>	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 167 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, DFSDDLTO, 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 319 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 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 9: Utilities Reference: System*.

Status codes are not set for SNAP calls.

## /DIAGNOSE Command SNAP Function

The /DIAGNOSE command SNAP function provides a non-intrusive alternative to taking a console dump. The /DIAGNOSE command SNAP function takes a current snapshot of system resources at any time without negatively impacting IMS. It then sends this system resource information to either OLDS or trace data sets as type X'6701' log records. Using this command can significantly decrease the time required to provide problem determination data to IBM service.

The /DIAGNOSE command SNAP function captures information for the following resources:

- A specific IMS control block
- A user-defined node
- A user-defined transaction
- A user-defined LTERM
- A user-defined USER
- Any area of storage within the control region address space by specifying the address of that storage area

The /DIAGNOSE command is a standard type-1 command. See the *IMS Version 9: Command Reference* for more information.

## Common Trace Table Interface

The common trace table interface consists of the traces shown in Table 35. For each trace, Table 35 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 35. Trace Tables in the Common Trace Interface

Trace	Table Type	ID	What Is Traced	Where Described
	ALL	ALL	All IMS table traces	
Common Service Layer Trace	CSLT	CS	IMS's interaction with the CSL	"Common Service Layer Trace" on page 197
DASD log trace	DLOG	DG	DASD logging	See "DASD log" in Table 36 on page 194
Dispatcher trace (online only)	DISP	DS	Dispatcher activities	"Dispatcher Trace" on page 203
DL/I and lock	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 263
Enhanced Command Trace	OCMD	OC	Activity related to commands that originate from OM	Not available
External subsystem trace (online only)	SUBS	SU	Subsystem activities	"External Subsystem Trace" on page 212
Fast Path Trace		FT	Fast Path activity	"Fast Path Trace" on page 251
Force trace	FORCE	FO	Internal trace for IMS initialization	Not in use



Table 35. Trace Tables in the Common Trace Interface (continued)

Trace	Table Type	ID	What Is Traced	Where Described
Intercommunications trace	IDCO	IC	VTAM exit activity	“Starting the Trace” on page 327
Latch trace (online only)	LATC	LA	Latch activities	“Latch Trace” on page 242
Log router trace	LRTT	LR	Log router activity	“Log Router Trace Data” on page 492
LU trace	LUMI	LU	LU 6.2 activity	“LU Manager Trace” on page 382
Multiple Systems Coupling Trace	MSCT	MS	MSC activities. Not yet used.	Not applicable
Online Recovery Manager trace	ORTT	OR	ORS activity	“Online Recovery Manager Trace” on page 306
Operations Manager (OM) commands	OCMD	OC	Activity related to commands received from OM	Not available
OTMA trace	OTMT	OA	OTMA activity	“OTMA Trace” on page 400
Resource Recovery Service (RRS)	RRST	RR	Resource Recovery Service activity in dependent region(s)	“Resource Recovery Services Trace” on page 225
Queue manager trace	QMGR	QM	Queue manager activity	“Queue Manager Trace” on page 246
Scheduler trace (online only)	SCHD	SC	Scheduler activities	“Scheduler Trace” on page 238
Shared queues interface trace	SQTT	SQ	Shared queues interface activities.	“Shared Queues Interface Trace” on page 250
Storage Manager trace	STRG	SM	Storage Manager activities	“Storage Manager Trace” on page 241
Transport Manager subsystem	Not available	TS	Transport Manager subsystem activity	Not available

- The following topics provide additional information:
- “Finding the Trace Tables in a Dump” on page 192
  - “Format of Trace Records” on page 194
  - “IMS Trace Function Codes” on page 194
  - “Dispatcher Trace” on page 203
  - “ITASK ECB Posting” on page 211
  - “System Post Codes” on page 212
  - “External Subsystem Trace” on page 212
  - “Layout of the X'57' Variable Section” on page 214
  - “Layout of the X'58' Variable Section” on page 219
  - “Resource Recovery Services Trace” on page 225
  - “Scheduler Trace” on page 238
  - “Storage Manager Trace” on page 241
  - “Latch Trace” on page 242
  - “Queue Manager Trace” on page 246
  - “Shared Queues Interface Trace” on page 250

- I • “Fast Path Trace” on page 251

## **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 68 on page 193 explains how to find the location of each of the traces in a dump.

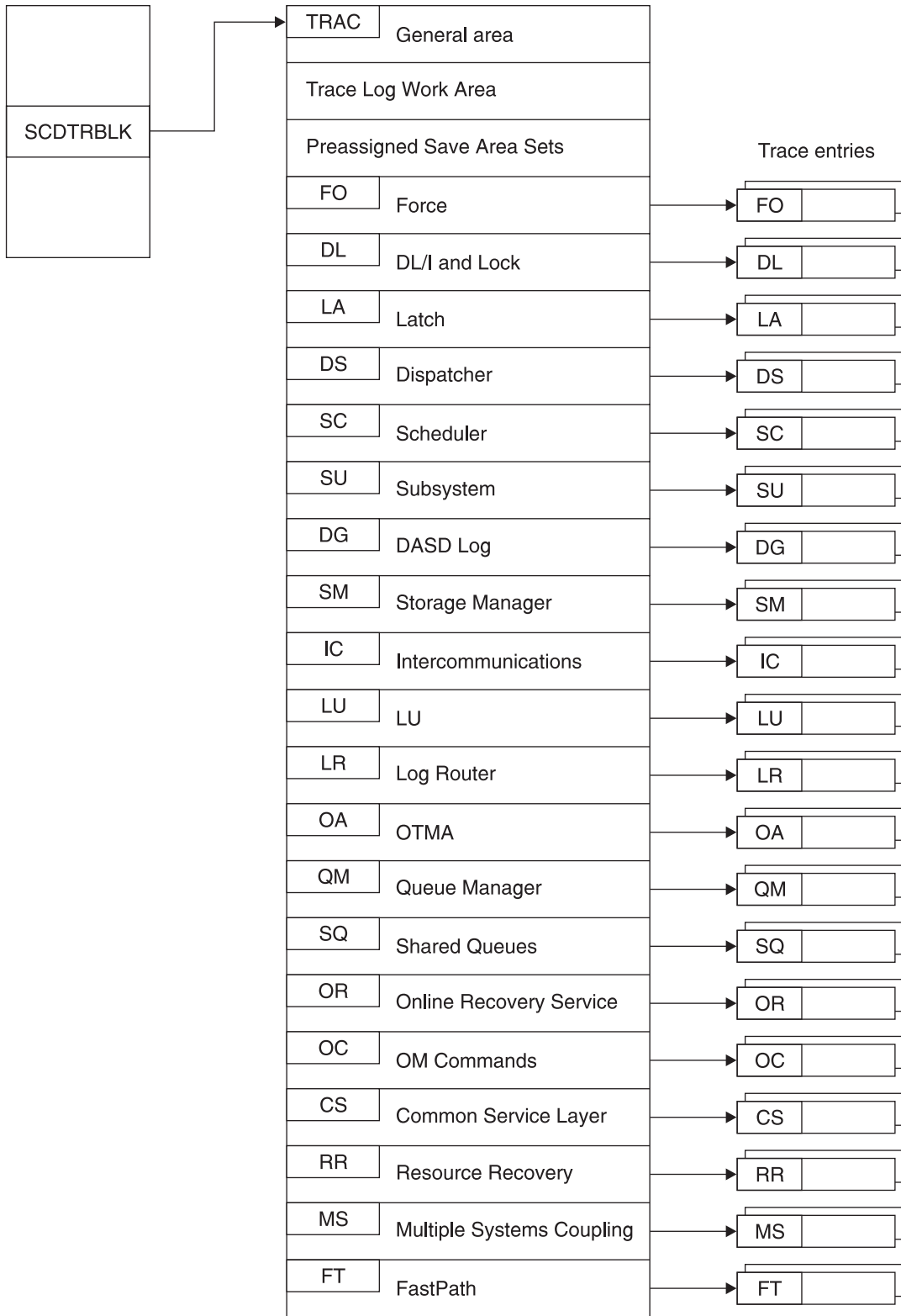


Figure 68. How to Locate Trace Tables

## 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 69, 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.

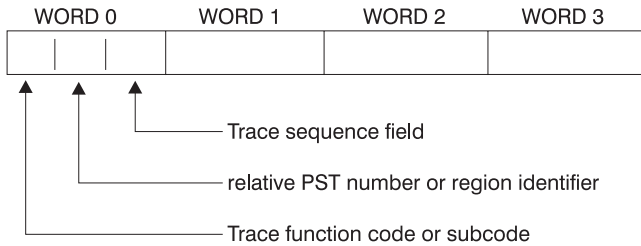


Figure 69. General Trace Record Format

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 36 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 following trace table examples:

- “Dispatcher Trace” on page 203
- “External Subsystem Trace” on page 212,
- “Resource Recovery Services Trace” on page 225,
- “Scheduler Trace” on page 238,
- “Storage Manager Trace” on page 241.
- “Latch Trace” on page 242
- “Queue Manager Trace” on page 246
- “Shared Queues Interface Trace” on page 250
- “Fast Path Trace” on page 251

You can also find a one-line description of each trace code in macro DFSTRAE0.

Table 36. 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

Table 36. Trace Function Codes (continued)

Trace Table	Function Code	Description
	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
	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'D9'	HALDB online reorganization trace entry
	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)

Table 36. Trace Function Codes (continued)

Trace Table	Function Code	Description
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 using PC/PT
	X'0F'	Dispatcher abend issued
	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 (z/OS branch-entry local POST)
	X'14'	DFSCIR called to create an ITASK
	X'15'	DFSKPXT issued z/OS 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	
Resource Recovery Services	X'A5'	Resource Recovery Services (RRS) Calls
Scheduler	X'41'	Scheduling starts
	X'42'	Block mover

Table 36. Trace Function Codes (continued)

Trace Table	Function Code	Description
	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 <sup>1</sup>	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
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
Fast Path	X'9C'	The FP Notify trace code
	X'9D'	The FP General trace code
	X'9E'	Fast Path log router interface
		<b>Note:</b> For more information, see “Fast Path Tracker Trace Entries” on page 483.
	X'9F'	Fast Path log router interface
		<b>Note:</b> For more information, see “Fast Path Tracker Trace Entries” on page 483.
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.

## Common Service Layer Trace

The Common Service Layer trace (CSLT) provides information about activity related to IMSs interaction with the Common Service Layer. This includes IMSs interaction with OM, RM, and SCI.

You can turn on the common service layer trace during online operation by using the /TRACE command. Each trace entry is X'20' bytes long. You can specify trace output destination and tracing volume on the /TRACE command.

If you send the 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 common service layer trace in a dump, look for eyecatcher \*\*CSTR.

### Related Reading:

- | • For more information about the CSLT parameter, see “Tailoring the IMS System to Your Environment” in *IMS Version 9: Installation Volume 2: System Definition and Tailoring*.
- | • For more information about the /TRACE command, see “/TRACE” in *IMS Version 9: Command Reference*.
- | • For more information about the common trace table interface, see “Common Trace Table Interface” on page 190.
- | • For more information about the Offline Dump Formatter, see “Formatting IMS Dumps Offline” on page 154 and the “Offline Dump Formatter Utility (DFSOFMD0)” section in *IMS Version 9: Utilities Reference: System*.
- | • The File Select and Formatting Print Utility, see “File Select and Formatting Print Utility” on page 152 and the “File Select and Formatting Print Utility (DFSERA10)” section in *IMS Version 9: Utilities Reference: System*

### Format of Common Service Layer Trace Records

| The diagrams following Table 37 depict the trace (low level) record format of the following functions with these subfunction codes (SC):

| *Table 37. Common Service Layer Trace Function and Subfunction Codes*

Subfunction Codes (SC)	Function
X'01'	Process flow
X'02'	Storage error
X'03'	Load or delete error
X'05'	Parameter validation error
X'09'	AWE error
X'0A'	Latch error
X'0F'	Miscellaneous MVS service error
X'11'	CQS service error
X'12'	SCI service error
X'13'	RM service error
X'14'	OM service error
X'15'	Unknown function
X'16'	Parameter list version error
X'50'	SCI Input exit
X'51'	SCI Notify exit
X'80'	Terminal logon
X'81'	Terminal logoff
X'82'	User signon
X'83'	User signoff
X'84'	DFSRMAM0 query interface
X'85'	DFSRMUP0 update interface
X'86'	RM resource entry
X'90'	Miscellaneous RM directive processing errors

| The following diagram shows the format of the trace records for each of the subfunction codes listed above. Each trace record has a trace function code of X'A2' and is X'20' bytes long.



```

| Word 0   - byte 1 - Trace function code
|           - byte 2 - Trace function subcode
|           - byte 3-4 - Trace record sequence number
| Words 1-5 - Contains information about the activity being traced.
|           The information recorded in this part of the trace record
|           depends on the trace function subcode of the trace record.
| Words 6-7 - Timestamp (STCK value)

```

The data in words 1-5, which is specific to each trace entry, is described in the following diagrams:

```

| Trace function subcode = X'01'
| Description: Process flow (Begin Process and Normal Process)

```

```

| Word 1      - byte 1 - Service code
|             - byte 2 - Object type
|             - bytes 3-4 - Module identifier
| Word 2      - Not used
| Word 3      - Not used
| Word 4      - Not used
| Word 5      - Thread ECB address

```

```

| Trace function subcode = X'01'
| Description: Process flow (End Process)

```

```

| Word 1      - byte 1 - Service code
|             - byte 2 - Object type
|             - bytes 3-4 - Module identifier
| Word 2      - Not used
| Word 3      - Not used
| Word 4      - Return code
| Word 5      - Reason code

```

```

| Trace function subcode = X'02'
| Description: Storage Request Error

```

```

| Word 1      - byte 1 - Service code
|             - byte 2 - Object type
|             - bytes 3-4 - Module identifier
| Word 2      - Return code
| Word 3      - Storage length
| Word 4      - Storage address
| Word 5      - Thread ECB address

```

```

| Trace function subcode = X'03'
| Description: Module LOAD/DELETE Error

```

```

| Word 1      - byte 1 - Service code
|             - byte 2 - Object type
|             - bytes 3-4 - Module identifier
| Word 2      - Return code
| Word 3      - bytes 1-2 - Target module identifier
|             - bytes 3-4 - Not used
| Word 4      - Not used
| Word 5      - Thread ECB address

```

```

| Trace function subcode = X'04'
| Description: Proclib/Execute Parameter Error

```

```

| Word 1      - byte 1 - Service code
|             - byte 2 - Object type
|             - bytes 3-4 - Module identifier
| Word 2      - Return code
| Word 3      - Not used
| Word 4      - Not used
| Word 5      - Thread ECB address

```

```

| Trace function subcode = X'05'
| Description: Parameter Validation Error
|
| Word 1          - byte 1 - Not used
|                 - byte 2 - Object type
| Words 2-5      - Parameter Value
|
| Trace function subcode = X'07'
| Description: TCB/Thread Error
|
| Word 1          - byte 1 - Service code
|                 - byte 2 - Object type
|                 - bytes 3-4 - Module identifier
| Word 2          - Return code
| Word 3          - Not used
| Word 4          - Not used
| Word 5          - Thread ECB address
|
| Trace function subcode = X'09'
| Description: AWE Error (Create AWE Queue Server, Get AWE, Enq AWE)
|
| Word 1          - byte 1 - Service code
|                 - byte 2 - Object type
|                 - bytes 3-4 - Module identifier
| Word 2          - Return code
| Word 3          - Not used
| Word 4          - Not used
| Word 5          - Thread ECB address
|
| Trace function subcode = X'09'
| Description: AWE Error (Invalid AWE)
|
| Word 1          - byte 1 - Service code
|                 - byte 2 - Object type
|                 - bytes 3-4 - Module identifier
| Word 2          - byte 1 - Function code
|                 - bytes 2-4 - Not used
| Word 3          - Address of invalid AWE
| Word 4          - Enqueuer's ECB
| Word 5          - Thread ECB address
|
| Trace function subcode = X'0A'
| Description: LATCH Error
|
| Word 1          - byte 1 - Service code
|                 - byte 2 - Object type
|                 - bytes 3-4 - Module identifier
| Word 2          - Return code
| Word 3          - Not used
| Word 4          - Not used
| Word 5          - Thread ECB address
|
| Trace function subcode = X'0F'
| Description: Miscellaneous MVS Service Error
|
| Word 1          - byte 1 - Service code
|                 - byte 2 - Object type
|                 - bytes 3-4 - Module identifier
| Word 2          - Return code
| Word 3          - Reason code
| Word 4          - Not used
| Word 5          - Thread ECB address
|
| Trace function subcode = X'11'
| Description: CQS Service Error
|
| Word 1          - byte 1 - Service code
|                 - byte 2 - Object type
|                 - bytes 3-4 - Module identifier

```

| Word 2 - Return code  
 | Word 3 - Reason code  
 | Word 4 - Not used  
 | Word 5 - Thread ECB address

| Trace function subcode = X'12'  
 | Description: SCI Service Error

| Word 1 - byte 1 - Service code  
 | - byte 2 - Object type  
 | - bytes 3-4 - Module identifier  
 | Word 2 - Return code  
 | Word 3 - Reason code  
 | Word 4 - Not used  
 | Words 4-5 - Target member name or zero

| Trace function subcode = X'13'  
 | Description: RM Service Error

| Word 1 - byte 1 - Service code  
 | - byte 2 - Object type  
 | - bytes 3-4 - Module identifier  
 | Word 2 - Return code  
 | Word 3 - Reason code  
 | Word 4 - Not used  
 | Words 4-5 - Target member name or zero

| Trace function subcode = X'14'  
 | Description: OM Service Error

| Word 1 - byte 1 - Service code  
 | - byte 2 - Object type  
 | - bytes 3-4 - Module identifier  
 | Word 2 - Return code  
 | Word 3 - Reason code  
 | Word 4 - Not used  
 | Words 4-5 - Target member name or zero

| There are two formats used for Trace Subcode X'15':

| Trace function subcode = X'15'  
 | Description: Unknown Function Exit Errors

| Word 1 - bytes 1-2 - Function Code  
 | - bytes 3-4 - Module identifier  
 | Words 2-5 - SCI Token

| Trace function subcode = X'15'  
 | Description: Unknown Function Exit Errors

| Word 1 - bytes 1-2 - Function Code  
 | - bytes 3-4 - Module identifier  
 | Words 2-3 - Subject member name  
 | Words 4-5 - Subject member type and subtype

| Trace function subcode = X'16'  
 | Description: Parameter list version errors

| Word 1 - byte 1 - Not used  
 | - byte 2 - Object type  
 | - bytes 3-4 - Module identifier  
 | Words 2 - Parameter version  
 | Words 3-4 - Member name  
 | Word 5 - Member version

| Trace function subcode = X'50'  
 | Description: SCI Input Exit

| Word 1 - byte 1 - Service code  
 | - byte 2 - Flag

```

|           - bytes 3-4 - Source member type
| Word 2    - Function code
| Word 3    - Subfunction code
| Words 4-5 - Source member name

```

```

| Trace function subcode = X'51'
| Description:  SCI Notify Exit

```

```

| Word 1    - byte 1 - Service code
|           - byte 2 - Flag
|           - bytes 3-4 - Source member type
| Word 2    - Source member type
| Word 3    - Event
| Words 4-5 - Source member name

```

```

| Trace function subcode = X'80'
| Description:  Logon Process

```

```

| Word 1    - bytes 1-2 - Return code
|           - byte 3 - CLBSRM1
|           - byte 4 - CLBSRM2
| Words 2-3 - Node name
| Word 4    - Not used
| Word 5    - Thread ECB address

```

```

| Trace function subcode = X'81'
| Description:  Logoff Process

```

```

| Word 1    - bytes 1-2 - Return code
|           - byte 3 - CLBSRM1
|           - byte 4 - CLBSRM2
| Words 2-3 - Node name
| Word 4    - Not used
| Word 5    - Thread ECB address

```

```

| Trace function subcode = X'82'
| Description:  Signon Process

```

```

| Word 1    - bytes 1-2 - Return code
|           - byte 3 - CLBSRM1
|           - byte 4 - CLBSRM2
| Words 2-3 - User structure name
| Word 4    - Not used
| Word 5    - Thread ECB address

```

```

| Trace function subcode = X'83'
| Description:  Signoff Process

```

```

| Word 1    - bytes 1-2 - Return code
|           - byte 3 - CLBSRM1
|           - byte 4 - CLBSRM2
| Words 2-3 - User structure name
| Word 4    - Not used
| Word 5    - Thread ECB address

```

```

| Trace function subcode = X'84'
| Description:  DFSRMAM0 query interface

```

```

| Word 1    - byte 1 - RMAP flag 1
|           - byte 2 - RMAPE flag 1
|           - byte 3 - RMAPE flag 2
|           - byte 4 - RMAPE flag 3
| Words 2-3 - Resource name
| Word 4    - Data pointer
| Word 5    - Return code

```

```

| Trace function subcode = X'85'
| Description:  DFSRMUPO update interface

```

```

| Word 1    - byte 1 - RMAP flag 1

```

```

|           - byte 2 - RMAPE flag 1
|           - byte 3 - RMAPE flag 2
|           - byte 4 - RMAPE flag 3
| Word 2    - Resource pointer
| Word 3    - Not used
| Word 4    - Data pointer
| Word 5    - Return code
| Trace function subcode = X'86'
| Description: RM Resource Entry trace
|
| Word 1    - byte 1 - Service code
|           - byte 2 - Condition code
|           - bytes 3-4 - Module identifier
| Words 2-3 - Resource name
| Word 4    - byte 1 - Resource Type
|           - byte 2 - Not used
|           - byte 3 - Input version number (last byte)
|           - byte 4 - Output version number (last byte)
| Word 5    - Thread ECB address
| Trace function subcode = X'90'
| Description: Miscellaneous Directive Processing errors
|
| Word 1    - byte 1 - Service code
|           - byte 2 - Not used
|           - bytes 3-4 - Module identifier
| Words 2-3 - Process name
| Word 4    - Process type
| Word 5    - Not used

```

## 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.

Table 38 shows the general format of a dispatcher trace entry.

Table 38. Dispatcher Trace Record Format

WORD 0			WORD 1	WORD 2	WORD 3	WORD 4	WORD 5	WORD 6	WORD 7
I	T	SEQ NUM							TIME STAMP

### where

**I**

### represents

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.

The dispatcher trace formatting usually includes the functional area. If you need this information because the trace is in a raw format, the codes can be obtained by assembling the following macro statement: DFSKDT  
FUNC=EQUATES.

**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 – Reinstatement 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 T0= 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 – SAPCNTRL2 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 T0=, 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 00025E44 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 006CFE88 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 70. 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 a z/OS SUSPEND. This task is reactivated by a RESUME invoked by the post exit posting routine or the IMS posting routine.

## System Post Codes

Table 39 lists only a subset of the possible post codes.

Table 39. 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 DFSCP00 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 DFSRDLOG0 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 71 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.

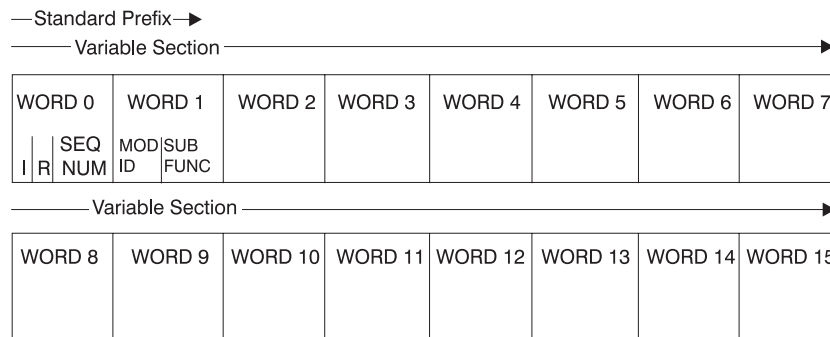


Figure 71. External Subsystem (ESS) Trace Record Format

**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 40 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 40. 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'	DFSESS10	IMS control region identify
	X'0012'		Dependent region identify
	X'0040'		Control region identify error
	X'0041'		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'	DFSESS00	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'	DFSESD50	Dependent region ESS term thread
	X'0003'		Dependent region ESS term thread region
	X'0004'		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'	DFSFESP0	ESS commit processor entered
	X'0017'		ESS commit processor exited
	X'0018'		ESS commit processor R-I-D request

Table 40. 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'	DFSIES10	Mother ITASK request
	X'0002'		Control region ESS attach
X'0506'	X'0006'	DFSESPRO	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 (DFSANE 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
word 4 -- byte 0 GESEGF1 (DFSGESE macro global flag1)
           byte 1 GESEGF2 (DFSGESE macro global flag2)
           byte 2 GESEGF3 (DFSGESE macro global flag3)
           byte 3 not used
           byte 4 SSIDFLG1 (DFSSSIE subsys status flag1)
           byte 5 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'** DFSESECT0 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 LCRESST (DFSLCRE ESS resource manager status byte1)  
     byte 3 LCRESST (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 LCRESST (DFSLCRE ESS resource manager status byte1)  
     byte 3 LCRESST (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 LCRESST (DFSLCRE ESS resource manager status byte1)
        byte 3 LCRESST (DFSLCRE ESS resource manager status byte2)
words 8 through 11 RRETKEN (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 72 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

GLOBAL 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 72. Example of an External Subsystem Trace (SST)

## Resource Recovery Services Trace

The Resource Recovery Service Trace (RRST) provides information about relevant Resource Recovery Service (RRS) events in the IMS dependent REGION. Use the trace under direction of IBM support personnel when problems are suspected in the Resource Recovery Services area.

You can enable the Resource Recovery Service trace by using the /TRACE SET ON TABLE RRST command. When you specify OPTION LOG, IMS writes the trace externally as type X'67FA' records.

- | The following topics provide additional information:
- | • “Format of Trace Records”
- | • “RRST Trace Examples” on page 233
- | • “RRST Entries for OTMA Modules” on page 236

### Format of Trace Records

The diagram below shows the general format of a Resource Recovery Service trace entry. The standard fields trace, present in every trace entry, are described below.

Word 0 -- byte 1 One-byte trace ID field. This byte indicates the type of the trace entry.  
 byte 2 One-byte trace sub function code.  
 byte 3-4 Two-byte trace sequence number assigned by the IMS trace component.

Word 1 -- byte 1 One-byte numeric Resource Recovery Service call code (see the RRS call table below).  
 byte 2 One-byte LCREGFLG.  
 byte 3-4 Two-byte RRS return code.

Word 2 -- byte 1-2 Two-byte PST number - PSTPSTNR.  
 byte 3-4 Not used.

Word 3 -- byte 1-4 Four-byte LCRERRSF.

Word 4 -- Not used.  
 Word 5 -- Not used.  
 Word 6 -- Not used.  
 Word 7 -- byte 1-4 Bytes 3 through 6 of the system clock (STCK) at the time the trace entry was created.

Table 41 shows the Resource Recovery Services calls associated with the subfunction codes (SC):

*Table 41. Resource Recovery Service Calls Associated with the Subfunction Codes*

Subfunction Code	Function
X'00'	ATRBACK
X'01'	ATRCMIT
X'02'	ATRDINT
X'03'	ATREINT
X'04'	ATREINT5
X'05'	ATRIERS
X'06'	ATRIERS
X'07'	ATRIRLN
X'08'	ATRIRNI
X'09'	ATRIRRI
X'0A'	ATRISLN
X'0B'	ATRPDUE
X'0C'	ATRREIC
X'0D'	ATTRURD
X'0E'	ATTRWID
X'0F'	ATRSROI
X'10'	ATRSIT
X'11'	ATRSPID
X'12'	ATRSUSI2
X'13'	CRGDRM
X'14'	CRGGRM
X'15'	CRGSEIF
X'16'	CRXSEIF
X'17'	CTXBEGC
X'18'	CTXEINT1
X'19'	CTXDINT
X'1A'	CTXENDC
X'1B'	CTXSWCH
X'1C'	CTXSCID
X'1D'	IEANTCR
X'1E'	IEANTRT

Table 42 on page 227 shows the Resource Recovery Services (RRS) function routines associated with the DFSRRSI function routine codes (FRC):

Table 42. Resource Recovery Services Function Routines Associated with DFSRRSI Function Routine Codes

Function Routine Codes	Function Routine
X'01'	Register
X'02'	Restart
X'03'	End_Restart
X'04'	Unregister
X'05'	Switch_Context
X'06'	Determine_Syncpt_Coord
X'07'	Initiate_Syncpt
X'08'	End_Context
X'09'	Retain_Interest
X'0A'	Post_Deferred_UR
X'0B'	Disassociate_Context
X'0C'	Coordinate_Backout
X'0D'	Perform_Syncpt
X'0E'	Identify_Context
X'0F'	Post_Deferred_Backout
X'10'	Unhook_for_Phase2
X'11'	RRS_Validation
X'12'	Delete_UR_Interest
X'13'	Retrieve_XID
X'14'	Determine_Batch_Coord
X'15'	Create_Context
X'16'	Set_Side_Information
X'17'	Create_Cascaded_UR
X'18'	Express_UR_Interest
X'19'	Commit_UR
X'1A'	Backout_UR
X'1B'	Associate_Context

The following diagrams show the format of the trace records. Each trace record has a trace function code of X'A5' and is X'20' bytes long.

Subfunction Code = X'00'  
 Description = Resource Recovery Services - ATRBACK

Word 1 - byte 1 - Numeric DFSRRSI function routine code  
 - byte 2 - LCREGFLG  
 - byte 3-4 - RRS return code

Word 2 - byte 1-2 - PSTPSTNR  
 - byte 3-4 - Not used

Word 3 - LCRERRSF

Word 4 - Not used

Words 5-6 - LCURIDNT

Word 7 - Low 4 byte time stamp (STCK)

Subfunction Code = X'01'  
 Description = Resource Recovery Services - ATRCMIT

Word 1 - byte 1 - Numeric DFSRRSI function routine code  
 - byte 2 - LCREGFLG  
 - byte 3-4 - RRS return code

Word 2 - byte 1-2 - PSTPSTNR  
 - byte 3-4 - Not used

Word 3 - LCRERRSF

Word 4 - Not used

Words 5-6 - LCURIDNT

Word 7 - Low 4 byte time stamp (STCK)

Subfunction Code = X'02'  
 Description = Resource Recovery Services - ATRDINT

Word 1 - byte 1 - Numeric DFSRRSI function routine code  
 - byte 2 - LCREGFL

Word 2 - byte 3-4 - RRS return code  
 - byte 1-2 - PSTPSTNR  
 - byte 3-4 - Not used

Word 3 - LCRERRSF

Word 4 - Not used

Words 5-6 - LCURIDNT

Word 7 - Low 4 byte time stamp (STCK)

Subfunction Code = X'03'  
 Description = Resource Recovery Services - ATREINT

Word 1 - byte 1 - Numeric DFSRRSI function routine code  
 - byte 2 - LCREGFLG  
 - byte 3-4 - RRS return code

Word 2 - byte 1-2 - PSTPSTNR  
 - byte 3-4 - Not used

Word 3 - LCRERRSF

Word 4 - Not used

Words 5-6 - LCURIDNT

Word 7 - Low 4 byte time stamp (STCK)

Subfunction Code = X'04'  
 Description = Resource Recovery Services - ATREINT5

Word 1 - byte 1 - Numeric DFSRRSI function routine code  
 - byte 2 - LCREGFLG  
 - byte 3-4 - RRS return code

Word 2 - byte 1-2 - PSTPSTNR  
 - byte 3-4 - Not used

Word 3 - LCRERRSF

Word 4 - Not used

Words 5-6 - LCURIDNT

Word 7 - Low 4 byte time stamp (STCK)

Subfunction Code = X'05'  
 Description = Resource Recovery Services - ATRIBRS

Word 1 - byte 1 - Numeric DFSRRSI function routine code  
 - byte 2 - LCREGFLG  
 - byte 3-4 - RRS return code

Word 2 - byte 1-2 - PSTPSTNR  
 - byte 3-4 - Not used

Word 3 - LCRERRSF

Word 4 - Not used

Word 5 - Not used

Word 6 - Not used

Word 7 - Low 4 byte time stamp (STCK)

Subfunction Code = X'06'  
 Description = Resource Recovery Services - ATRIERIS

Word 1 - byte 1 - Numeric DFSRRSI function routine code  
 - byte 2 - LCREGFLG  
 - byte 3-4 - RRS return code

Word 2 - byte 1-2 - PSTPSTNR  
 - byte 3-4 - Not used

Word 3 - LCRERRSF

Word 4 - Not used

Word 5 - Not used

Word 6 - Not used

Word 7 - Low 4 byte time stamp (STCK)

Subfunction Code = X'07'  
 Description = Resource Recovery Services - ATRIRLN

Word 1	- byte 1 - Numeric DFSRRSI function routine code
	- byte 2 - LCREGFLG
	- byte 3-4 - RRS return code
Word 2	- byte 1-2 - PSTPSTNR
	- byte 3-4 - Not used
Word 3	- LCRERRSF
Word 4	- Not used
Word 5	- Not used
Word 6	- Not used
Word 7	- Low 4 byte time stamp (STCK)
Subfunction Code	= X'08'
Description	= Resource Recovery Services - ATRIRNI
Word 1	- byte 1 - Numeric DFSRRSI function routine code
	- byte 2 - LCREGFLG
	- byte 3-4 - RRS return code
Word 2	- byte 1-2 - PSTPSTNR
	- byte 3-4 - Not used
Word 3	- LCRERRSF
Word 4	- Not used
Word 5	- Not used
Word 6	- Not used
Word 7	- Low 4 byte time stamp (STCK)
Subfunction Code	= X'09'
Description	= Resource Recovery Services - ATRIRRI
Word 1	- byte 1 - Numeric DFSRRSI function routine code
	- byte 2 - LCREGFLG
	- byte 3-4 - RRS return code
Word 2	- byte 1-2 - PSTPSTNR
	- byte 3-4 - Not used
Word 3	- LCRERRSF
Word 4	- Not used
Word 5	- Not used
Word 6	- Not used
Word 7	- Low 4 byte time stamp (STCK)
Subfunction Code	= X'0A'
Description	= Resource Recovery Services - ATRISLN
Word 1	- byte 1 - Numeric DFSRRSI function routine code
	- byte 2 - LCREGFLG
	- byte 3-4 - RRS return code
Word 2	- byte 1-2 - PSTPSTNR
	- byte 3-4 - Not used
Word 3	- LCRERRSF
Word 4	- Not used
Word 5	- Not used
Word 6	- Not used
Word 7	- Low 4 byte time stamp (STCK)
Subfunction Code	= X'0B'
Description	= Resource Recovery Services - ATRPDUE
Word 1	- byte 1 - Numeric DFSRRSI function routine code
	- byte 2 - LCREGFLG
	- byte 3-4 - RRS return code
Word 2	- byte 1-2 - PSTPSTNR
	- byte 3-4 - Not used
Word 3	- LCRERRSF
Word 4	- byte 1-2 - ATRPDUEEXITNUMBER
	- byte 3-4 - ATRPDUECOMPLETION
Word 5	- Not used
Word 6	- Not used
Word 7	- Low 4 byte time stamp (STCK)

Subfunction Code = X'0C'  
 Description = Resource Recovery Services - ATRREIC

Word 1 - byte 1 - Numeric DFSRRSI function routine code  
 - byte 2 - LCREGFLG  
 - byte 3-4 - RRS return code

Word 2 - byte 1-2 - PSTPSTNR  
 - byte 3-4 - Not used

Word 3 - LCRERRSF

Word 4 - Not used

Words 5-6 - LCURCNTX

Word 7 - Low 4 byte time stamp (STCK)

Subfunction Code = X'0D'  
 Description = Resource Recovery Services - ATRRURD

Word 1 - byte 1 - Numeric DFSRRSI function routine code  
 - byte 2 - LCREGFLG  
 - byte 3-4 - RRS return code

Word 2 - byte 1-2 - PSTPSTNR  
 - byte 3-4 - Not used

Word 3 - LCRERRSF

Word 4 - Not used

Words 5-6 - IMS\_PCTASK\_URI\_TOKEN

Word 7 - Low 4 byte time stamp (STCK)

Subfunction Code = X'0E'  
 Description = Resource Recovery Services - ATRRWID

Word 1 - byte 1 - Numeric DFSRRSI function routine code  
 - byte 2 - LCREGFLG  
 - byte 3-4 - RRS return code

Word 2 - byte 1-2 - PSTPSTNR  
 - byte 3-4 - Not used

Word 3 - LCRERRSF

Word 4 - Not used

Words 5-6 - URI\_Token

Word 7 - Low 4 byte time stamp (STCK)

Subfunction Code = X'0F'  
 Description = Resource Recovery Services - ATRSROI

Word 1 - byte 1 - Numeric DFSRRSI function routine code  
 - byte 2 - LCREGFLG  
 - byte 3-4 - RRS return code

Word 2 - byte 1-2 - PSTPSTNR  
 - byte 3-4 - Not used

Word 3 - LCRERRSF

Word 4 - Not used

Words 5-6 - URID

Word 7 - Low 4 byte time stamp (STCK)

Subfunction Code = X'10'  
 Description = Resource Recovery Services - ATRSIT

Word 1 - byte 1 - Numeric DFSRRSI function routine code  
 - byte 2 - LCREGFLG  
 - byte 3-4 - RRS return code

Word 2 - byte 1-2 - PSTPSTNR  
 - byte 3-4 - Not used

Word 3 - LCRERRSF

Word 4 - Not used

Word 5 - Not used

Word 6 - Not used

Word 7 - Low 4 byte time stamp (STCK)

Subfunction Code = X'11'  
 Description = Resource Recovery Services - ATRSPID

Word 1 - byte 1 - Numeric DFSRRSI function routine code



	- byte 2 - LCREGFLG
	- byte 3-4 - RRS return code
Word 2	- byte 1-2 - PSTPSTNR
	- byte 3-4 - Not used
Word 3	- LCRERRSF
Word 4	- Not used
Words 5-6	- IMS_PCTASK_URI_TOKEN
Word 7	- Low 4 byte time stamp (STCK)
Subfunction Code	= X'12'
Description	= Resource Recovery Services - ATRSUSI2
Word 1	- byte 1 - Numeric DFSRRSI function routine code
	- byte 2 0 - LCREGFLG
	- byte 3-4 - RRS return code
Word 2	- byte 1-2 - PSTPSTNR
	- byte 3-4 - Not used
Word 3	- LCRERRSF
Word 4	- Not used
Words 5-6	- IMS_PCTASK_URI_TOKEN
Word 7	- Low 4 byte time stamp (STCK)
Subfunction Code	= X'13'
Description	= RRMS Registration Services - CRGDRM
Word 1	- byte 1 - Numeric DFSRRSI function routine code
	- byte 2 - LCREGFLG
	- byte 3-4 - RRS return code
Word 2	- byte 1-2 - PSTPSTNR
	- byte 3-4 - Not used
Word 3	- LCRERRSF
Word 4	- Not used
Word 5	- Not used
Word 6	- Not used
Word 7	- Low 4 byte time stamp (STCK)
Subfunction Code	= X'14'
Description	= RRMS Registration Services - CRGGRM
Word 1	- byte 1 - Numeric DFSRRSI function routine code
	- byte 2 - LCREGFLG
	- byte 3-4 - RRS return code
Word 2	- byte 1-2 - PSTPSTNR
	- byte 3-4 - Not used
Word 3	- LCRERRSF
Word 4	- Not used
Word 5	- Not used
Word 6	- Not used
Word 7	- Low 4 byte time stamp (STCK)
Subfunction Code	= X'15'
Description	= RRMS Registration Services - CRGSEIF
Word 1	- byte 1 - Numeric DFSRRSI function routine code
	- byte 2 - LCREGFLG
	- byte 3-4 - RRS return code
Word 2	- byte 1-2 - PSTPSTNR
	- byte 3-4 - Not used
Word 3	- LCRERRSF
Word 4	- Not used
Word 5	- Not used
Word 6	- Not used
Word 7	- Low 4 byte time stamp (STCK)
Subfunction Code	= X'16'
Description	= RRMS Registration Services - CRXSEIF
Word 1	- byte 1 - Numeric DFSRRSI function routine code
	- byte 2 - LCREGFLG
	- byte 3-4 - RRS return code

Word 2	- byte 1-2 - PSTPSTNR - byte 3-4 - Not used
Word 3	- LCRERRSF
Word 4	- Not used
Word 5	- Not used
Word 6	- Not used
Word 7	- Low 4 byte time stamp (STCK)
Subfunction Code	= X'17'
Description	= RRMS Registration Services - CTXBEGC
Word 1	- byte 1 - Numeric DFSRRSI function routine code - byte 2 - LCREGFLG - byte 3-4 - RRS return code
Word 2	- byte 1-2 - PSTPSTNR - byte 3-4 - Not used
Word 3	- LCRERRSF
Word 4	- Not used
Words 5-6	- LCURCNTX
Word 7	- Low 4 byte time stamp (STCK)
Subfunction Code	= X'18'
Description	= RRMS Context Services - CTXEINT1
Word 1	- byte 1 - Numeric DFSRRSI function routine code - byte 2 - LCREGFLG - byte 3-4 - RRS return code
Word 2	- byte 1-2 - PSTPSTNR - byte 3-4 - Not used
Word 3	- LCRERRSF
Word 4	- Address(LCRE)
Word 5	- Not used
Word 6	- Not used
Word 7	- Low 4 byte time stamp (STCK)
Subfunction Code	= X'19'
Description	= RRMS Context Services - CTXDINT
Word 1	- byte 1 - Numeric DFSRRSI function routine code - byte 2 - LCREGFLG - byte 3-4 - RRS return code
Word 2	- byte 1-2 - PSTPSTNR - byte 3-4 - Not used
Word 3	- LCRERRSF
Word 4	- Address(IMS_PC_CI_Token)
Word 5	- Not used
Word 6	- Not used
Word 7	- Low 4 byte time stamp (STCK)
Subfunction Code	= X'1A'
Description	= RRMS Context Services - CTXENDC
Word 1	- byte 1 - Numeric DFSRRSI function routine code - byte 2 - LCREGFLG - byte 3-4 - RRS return code
Word 2	- byte 1-2 - PSTPSTNR - byte 3-4 - Not used
Word 3	- LCRERRSF
Word 4	- Not used
Words 5-6	- LCURCNTX
Word 7	- Low 4 byte time stamp (STCK)
Subfunction Code	= X'1B'
Description	= RRMS Context Services - CTXSWCH
Word 1	- byte 1 - Numeric DFSRRSI function routine code - byte 2 - LCREGFLG - byte 3-4 - RRS return code
Word 2	- byte 1-2 - PSTPSTNR - byte 3-4 - Not used

Word 3 - LCRERRSF  
 Word 4 - Not used  
 Words 5-6 - LCURCNTX  
 Word 7 - Low 4 byte time stamp (STCK)

Subfunction Code = X'1C'  
 Description = RRMS Context Services - CTXSCID

Word 1 - byte 1 - Numeric DFSRRSI function routine code  
 - byte 2 - LCREGFLG  
 - byte 3-4 - RRS return code

Word 2 - byte 1-2 - PSTPSTNR  
 - byte 3-4 - Not used

Word 3 - LCRERRSF  
 Word 4 - Address(LCRE)

Word 5 - Not used  
 Word 6 - Not used  
 Word 7 - Low 4 byte time stamp (STCK)

Subfunction Code = X'1D'  
 Description = MVS Name/Token Services - IEANTCR

Word 1 - byte 1 - Numeric DFSRRSI function routine code  
 - byte 2 - LCREGFLG  
 - byte 3-4 - RRS return code

Word 2 - byte 1-2 - PSTPSTNR  
 - byte 3-4 - Not used

Word 3 - LCRERRSF  
 Word 4 - Not used

Word 5 - Not used  
 Word 6 - Not used  
 Word 7 - Low 4 byte time stamp (STCK)

Subfunction Code = X'1E'  
 Description = MVS Name/Token Services - IEANTRT

Word 1 - byte 1 - Numeric DFSRRSI function routine code  
 - byte 2 - LCREGFLG  
 - byte 3-4 - RRS return code

Word 2 - byte 1-2 - PSTPSTNR  
 - byte 3-4 - Not used

Word 3 - LCRERRSF  
 Word 4 - Not used

Word 5 - Not used  
 Word 6 - Not used  
 Word 7 - Low 4 byte time stamp (STCK)

## RRST Trace Examples

The following topic shows examples of the RRST trace in an OTMA and an APPC environment. The traces were gathered with tracing volume set to HIGH.

### RRST trace in an OTMA environment

```
CONTROL CNTRL STOPAFT=EOF
*****
* INPUT LOG DATA SET NAME(S): *
* DARIO.IMS1.OLDSP0.OTMU01.DECKS2
*****
* SELECTION FOR INTERNAL TRACE LOG RECORD(S) *
*
OPTION PRINT 0=5,V=67FA,L=2,E=DFSERA60,C=E
1 FUNCTION WORD 0 WORD 1 WORD 2 WORD 3 WORD 4 WORD 5 WORD 6 WORD 7 PAGE 0001
0* RRI TRACE TABLE - DATE 2004173 TIME 224754462929 OFFSET 028D SKIP 0000 TOTAL SKIP 00000000 RECORD NUMBER 0000028F
-DFSSLUM0 Exit 7B027F86 01C90000 00000000 00000000 00000000 00000000 00000000 BB676D0D B53E27A0 SYNCHRONOUS OUTPUT LU MANAGER
DFSRLM10 Exit 7B0281E8 06400000 C9D4E2F2 40404040 00000000 00000000 00000000 BB676D7A 2B82CF89 RECEIVE LU MANAGER RECEIVER
DFSSLUM0 Exit 7B0282FD 01C90000 00000000 00000000 00000000 00000000 00000000 BB676DBE 97EF7323 SYNCHRONOUS OUTPUT LU MANAGER
DFSRLM10 Exit 7B02856E 06400000 C9D4E2F3 40404040 00000000 00000000 00000000 BB676E36 DD321202 RECEIVE LU MANAGER RECEIVER
DFSRGFS0 (RRS) A0A68972 480005A6 0CB252F8 0CB25060 00010000 00010000 00000000 BB676EB9 17C56A2A ENTER COMMIT
CTXEINT1 (RRS) A5188973 15000000 00010000 00000000 00000000 00000000 00000000 B917C5DD CREATE_CONTEXT NO LCRE FLAGS
ATREINT5 (RRS) A5048974 18000000 00010000 04000000 00000000 00000000 00000000 BB676EB9 7E71E000 B917C977 EXPRESS UR_INTRST NO LCRE FLAGS
DFSRGFS0 (RRS) A0A78975 480005A7 0CB252F8 0CB25060 00000001 00000000 00000000 BB676EB9 17ED934A EXIT COMMIT
```

DFSRGFSO (RRS)	A0AA8976	480005AA	0001E4D9	7E71E000	7E71E000	01000000	0100001E	17ED934A	TKN TRACE (A0A7)		
DFSYTIB0 (RRS)	5A00897D	28010084	0CB25060	7E71E000	7E71E000	01000000	0100001E	B917EEAD	INPT MSG ENQUEUED(UR TOKEN WORD3-6)		
CTXDINT (RRS)	A5198C04	05000000	00010000	00000000	00AFBA048	19070000	00000000	BEB54D12	SWITCH_CONTEXT	NO LCRE FLAGS	
ATREINT (RRS)	A5038C05	05000000	00010000	00000000	00000000	BB676EBE	7E71E374	BEB54EDC	SWITCH_CONTEXT	NO LCRE FLAGS	
ATRRURD (RRS)	A50D8C88	06400000	00010000	60000000	00000000	BB676EBE	B54E530B	BF5AF676	DETRMN_SYNC_COORD	OUTPUT SENT	
DFSRGFSO (RRS)	A0A68CF7	480003A6	0CB252F8	0AFF5060	00010000	00010000	BB676EBF	5BACC466	ENTER COMMIT		
ATRDINT (RRS)	A5028CF8	12000000	00010000	20000000	00000000	00000000	00000000	BF5BAD3B	DELETE UR INTRST	NO LCRE FLAGS	
DFSRGFSO (RRS)	A0A78CF9	480003A7	0CB252F8	0AFF5060	00000001	00000000	BB676EBF	5BB19926	EXIT COMMIT		
DFSRGFSO (RRS)	A0AA8CFA	480003AA	0001C3E7	BB676EB9	17C86FAA	01000000	022A4060	5BB19926	TKN TRACE (A0A7)		
DFSRGFSO (RRS)	A0A68D0A	480001A6	0CB252F8	0AFF5060	00010000	00010000	BB676EBF	5BB30B26	ENTER COMMIT		
CTXSCID (RRS)	A51C8D0B	19000000	00010000	00000000	00000000	BB676EB9	17C86FAA	BF5BB31D	COMMIT UR	NO LCRE FLAGS	
ATRCMIT (RRS)	A5018D0C	19000000	00010000	00080000	00000000	00000000	00000000	BF5BB422	COMMIT UR	NO LCRE FLAGS	
DFSRGFSO (RRS)	A0A78D0D	480001A7	0CB252F8	0AFF5060	00000001	00000000	BB676EBF	5BFA1C26	EXIT COMMIT		
DFSRGFSO (RRS)	A0AA8D0E	480001AA	0001C3E7	BB676EB9	17C86FAA	01000000	022A4060	5BFA1C26	TKN TRACE (A0A7)		
ATRDINT (RRS)	A5028D05E	08400000	00010000	62002001	00000000	BB676EBE	7E71E374	BF5C4396	END_CONTEXT	OUTPUT SENT	
CTXENDC (RRS)	A51A8D5F	08400000	00010000	62002001	00000000	BB676EBE	B54E9F6B	BF5C4602	END_CONTEXT	OUTPUT SENT	
DFSRGFSO (RRS)	A0A68F9F	480005A6	0CB252F8	0CB25060	00010000	00010000	00010003	BB676ED7	ENTER COMMIT		
CTXEINT1 (RRS)	A5188FA0	15000000	00010000	00000000	00000000	00000000	00000000	D72204AF	CREATE_CONTEXT	NO LCRE FLAGS	
ATREINT5 (RRS)	A5048FA1	18000000	00010000	04000000	00000000	BB676ED7	7E71E000	D72206A2	EXPRESS UR INTRST	NO LCRE FLAGS	
DFSRGFSO (RRS)	A0A78FA2	480005A7	0CB252F8	0CB25060	00000001	00000000	00000000	BB676ED7	EXIT COMMIT		
DFSRGFSO (RRS)	A0AA8FA3	480005AA	0001E4D9	7E71E000	7E71E000	01000002	0100001E	220DB20D	TKN TRACE (A0A7)		
DFSYTIB0 (RRS)	5A008FAA	28010084	0CB25060	7E71E000	7E71E000	01000002	0100001E	D7223231	INPT MSG ENQUEUED(UR TOKEN WORD3-6)		
DFSAOSW0 (RRS)	7B0890AB	47000000	0B27C060	40C1D6E2	0CB252F8	00000000	BB676ED8	D8D3326F	APPC/OTMA SMQ AWE server		
DFSAAPC0 Exit	7B0290AC	02400000	00000000	0CB25060	00000000	00000000	BB676ED8	D8D3630F	DFSAAPC MSG SWITCH PROCESSOR		
DFSSLUM0 Exit	7B0290AD	01800000	00000002	00000000	00000000	00000000	BB676ED8	D8DD6EAF	SYNCHRONOUS OUTPUT LU MANAGER		
DFSYTIB0 (RRS)	5A0090B5	2802A084	0CB25060	7E71E000	7E71E000	01000002	0100001E	D8D8E13F	BE RESPONSE RECVD(UR TOKEN WORD3-6)		
DFSYLUS0 (RRS)	5A0091F1	2A060002	0CB25060	7E71E000	7E71E000	01000002	0100001E	D8F4361F	OTMA SERVICE:UNK FUNC		
DFSRGFSO (RRS)	A0A691FA	480001A6	0CB252F8	0CB25060	00020000	00010000	BB676ED8	F4376B81	ENTER COMMIT		
CTXSCID (RRS)	A51C91FB	19000000	00020000	00000000	00000000	BB676ED7	22060A4D	D8F437D0	COMMIT UR	NO LCRE FLAGS	
ATRCMIT (RRS)	A50191FC	19000000	00020000	00080000	00000000	00000000	00000000	D8F439BA	COMMIT UR	NO LCRE FLAGS	
DFSRGFSO (RRS)	A0A791FD	480001A7	0CB252F8	0CB25060	00000001	00000000	BB676ED8	FC093A68	EXIT COMMIT		
DFSRGFSO (RRS)	A0AA91FE	480001AA	0002E4D9	7E71E000	7E71E000	01000002	0100001E	FC093A68	TKN TRACE (A0A7)		
DFSRGFSO (RRS)	A0A69401	480005A6	0CB252F8	0CB25060	00010000	00010000	00010004	BB676EFE	ENTER COMMIT		
CTXEINT1 (RRS)	A5189402	15000000	00010000	00000000	00000000	00000000	00000000	FEB72191	CREATE_CONTEXT	NO LCRE FLAGS	
ATREINT5 (RRS)	A5049403	18000000	00010000	04000000	00000000	BB676EFE	7E71E000	FEB7242F	EXPRESS UR INTRST	NO LCRE FLAGS	
DFSRGFSO (RRS)	A0A79404	480005A7	0CB252F8	0CB25060	00000001	00000000	BB676EFE	B72C3502	EXIT COMMIT		
DFSRGFSO (RRS)	A0AA9405	480005AA	0001E4D9	7E71E000	7E71E000	01000004	0100001E	B72C3502	TKN TRACE (A0A7)		
DFSYTIB0 (RRS)	5A00940C	28010084	0CB25060	7E71E000	7E71E000	01000004	0100001E	FEB7314C	INPT MSG ENQUEUED(UR TOKEN WORD3-6)		
DFSAOSW0 (RRS)	7B0895E1	47000000	0B27C060	40C1D6E2	0CB252F8	00000000	BB676F0A	3B00B4A3	APPC/OTMA SMQ AWE server		
DFSAAPC0 Exit	7B0295E2	02400000	00000000	0CB25060	00000000	00000000	BB676F0A	3B000E323	DFSAAPC MSG SWITCH PROCESSOR		
DFSSLUM0 Exit	7B0295E3	01800000	00000003	00000000	00000000	00000000	BB676F0A	3B0ABC23	SYNCHRONOUS OUTPUT LU MANAGER		
DFSYTIB0 (RRS)	5A0095EB	2802A084	0CB25060	7E71E000	7E71E000	01000004	0100001E	0A3B0E44	BE RESPONSE RECVD(UR TOKEN WORD3-6)		
DFSYLUS0 (RRS)	5A0096BC	2A060002	0CB25060	7E71E000	7E71E000	01000004	0100001E	0A7AD931	OTMA SERVICE:UNK FUNC		
DFSRGFSO (RRS)	A0A696C5	480001A6	0CB252F8	0CB25060	00020000	00010000	BB676F0A	7ADED120	ENTER COMMIT		
1 FUNCTION	WORD 0	WORD 1	WORD 2	WORD 3	WORD 4	WORD 5	WORD 6	WORD 7		PAGE 0002	
-CTXSCID (RRS)	A51C96C6	19000000	00020000	00000000	00000000	BB676EFE	B7236902	0A7ADF43	COMMIT UR	NO LCRE FLAGS	
ATRCMIT (RRS)	A50196C7	19000000	00020000	00080000	00000000	00000000	00000000	0A7AE4B4	COMMIT UR	NO LCRE FLAGS	
DFSRGFSO (RRS)	A0A79739	480001A7	0CB252F8	0CB25060	00000001	00000000	BB676F0A	D26D5C0E	EXIT COMMIT		
DFSRGFSO (RRS)	A0AA973A	480001AA	0002E4D9	7E71E000	7E71E000	01000004	0100001E	D26D5C0E	TKN TRACE (A0A7)		
DFSRGFSO (RRS)	A0A69881	480005A6	0CB252F8	0CB25060	00010000	00010000	00010005	BB676F22	71395349	ENTER COMMIT	
CTXEINT1 (RRS)	A5189882	15000000	00010000	00000000	00000000	00000000	00000000	227139DC	CREATE_CONTEXT	NO LCRE FLAGS	
ATREINT5 (RRS)	A5049883	18000000	00010000	04000000	00000000	BB676F22	7E71E000	22713C07	EXPRESS UR INTRST	NO LCRE FLAGS	
DFSRGFSO (RRS)	A0A79884	480005A7	0CB252F8	0CB25060	00000001	00000000	BB676F22	71439009	EXIT COMMIT		
DFSRGFSO (RRS)	A0AA9885	480005AA	0001E4D9	7E71E000	7E71E000	01000006	0100001E	71439009	TKN TRACE (A0A7)		
DFSYTIB0 (RRS)	5A00988C	28010080	0CB25060	7E71E000	7E71E000	01000006	0100001E	22714D7A	INPT MSG ENQUEUED(UR TOKEN WORD3-6)		
DFSAOSW0 (RRS)	7B089890	47000000	0B27C060	40C1D6E2	0CB252F8	00000000	BB676F22	F5CA6326	APPC/OTMA SMQ AWE server		
DFSAAPC0 Exit	7B02999D	02400000	00000000	0CB25060	00000000	00000000	BB676F22	F5CA8E46	DFSAAPC MSG SWITCH PROCESSOR		
DFSSLUM0 Exit	7B02999E	01800000	00000004	00000000	00000000	00000000	BB676F22	F5D4A386	SYNCHRONOUS OUTPUT LU MANAGER		
DFSYTIB0 (RRS)	5A0099A6	2802C080	0CB25060	7E71E000	7E71E000	01000006	0100001E	22F5D812	BE RESPONSE RECVD(UR TOKEN WORD3-6)		
DFSYLUS0 (RRS)	5A0099B9	2A060002	0CB25060	7E71E000	7E71E000	01000006	0100001E	22F5DEAF	OTMA SERVICE:UNK FUNC		
DFSRGFSO (RRS)	A0A699C5	480001A6	0CB252F8	0CB25060	00030000	00010000	BB676F22	F5F4C026	ENTER COMMIT		
CTXSCID (RRS)	A51C99C6	19000000	00030000	00000000	00000000	BB676F22	713B6A69	22F5F53D	COMMIT UR	NO LCRE FLAGS	
ATRCMIT (RRS)	A50199C7	19000000	00030000	00080000	00000000	00000000	00000000	22F5F72B	COMMIT UR	NO LCRE FLAGS	
DFSRGFSO (RRS)	A0A799CA	480001A7	0CB252F8	0CB25060	00000001	00000000	BB676F22	FBFA8107	EXIT COMMIT		
DFSRGFSO (RRS)	A0AA99CB	480001AA	0003E4D9	7E71E000	7E71E000	01000006	0100001E	FBFA8107	TKN TRACE (A0A7)		
DFSRGFSO (RRS)	A0A69C13	480005A6	0CB252F8	0CB25060	00010000	00010006	BB676F52	B618084F	ENTER COMMIT		
CTXEINT1 (RRS)	A5189C14	15000000	00010000	00000000	00000000	00000000	00000000	52B6187C	CREATE_CONTEXT	NO LCRE FLAGS	
ATREINT5 (RRS)	A5049C15	18000000	00010000	04000000	00000000	BB676F52	7E71E000	52B61B58	EXPRESS UR INTRST	NO LCRE FLAGS	
DFSRGFSO (RRS)	A0A79C16	480005A7	0CB252F8	0CB25060	00000001	00000000	BB676F52	B6230D2F	EXIT COMMIT		
DFSRGFSO (RRS)	A0AA9C17	480005AA	0001E4D9	7E71E000	7E71E000	01000008	0100001E	B6230D2F	TKN TRACE (A0A7)		
DFSYTIB0 (RRS)	5A009C1E	28010084	0CB25060	7E71E000	7E71E000	01000008	0100001E	52B62C94	INPT MSG ENQUEUED(UR TOKEN WORD3-6)		
DFSAOSW0 (RRS)	7B089E2A	47000000	0B27C060	40C1D6E2	0CB252F8	00000000	BB676F5C	AB73B56C	APPC/OTMA SMQ AWE server		
DFSAAPC0 Exit	7B029E2B	02400000	00000000	0CB25060	00000000	00000000	BB676F5C	AB73E68C	DFSAAPC MSG SWITCH PROCESSOR		
DFSSLUM0 Exit	7B029E2C	01800000	00000005	00000000	00000000	00000000	BB676F5C	AB7BB62C	SYNCHRONOUS OUTPUT LU MANAGER		
DFSYTIB0 (RRS)	5A009E34	2802A084	0CB25060	7E71E000	7E71E000	01000008	0100001E	5CAB7F1D	BE RESPONSE RECVD(UR TOKEN WORD3-6)		
DFSYLUS0 (RRS)	5A009F94	2A060002	0CB25060	7E71E000	7E71E000	01000008	0100001E	5CD77072	OTMA SERVICE:UNK FUNC		
DFSRGFSO (RRS)	A0A69F9D	480001A6	0CB252F8	0CB25060	00020000	00010000	BB676F5C	D771B08A	ENTER COMMIT		
CTXSCID (RRS)	A51C9F9E	19000000	00020000	00000000	00000000	BB676F52	B61A728F	5CD7720F	COMMIT UR	NO LCRE FLAGS	
ATRCMIT (RRS)	A5019F9F	19000000	00020000	00080000	00000000	00000000	00000000	5CD773D6	COMMIT UR	NO LCRE FLAGS	
DFSRGFSO (RRS)	A0A79FA0	480001A7	0CB252F8	0CB25060	00000001	00000000	BB676F5D	26DB750C	EXIT COMMIT		

```

DFS707I END OF FILE ON INPUT
DFS708I OPTION COMPLETE
DFS703I END OF JOB
    
```

**RRST trace in an APPC environment**

```

CONTROL  CNTL STOPAFT=EOF
*****
* INPUT LOG DATA SET NAME(S): *
* DARIO.IMS1.OLDSPO.ASQAO1.DECKS2
*****
*
* SELECTION FOR INTERNAL TRACE LOG RECORD(S) *
*
OPTION PRINT 0=5,V=67FA,L=2,E=DFSERA60,C=E
1 FUNCTION WORD 0 WORD 1 WORD 2 WORD 3 WORD 4 WORD 5 WORD 6 WORD 7 PAGE 0001
0* RRI TRACE TABLE - DATE 2004173 TIME 232543999620 OFFSET 028D SKIP 0000 TOTAL SKIP 00000000 RECORD NUMBER 0000025E
-DFSRLM0 Exit 7B027EEE 01C90000 00000000 00000000 00000000 00000000 BB677710 C2AE808 SYNCHRONOUS OUTPUT LU MANAGER
DFSRLM10 Exit 7B028133 06400000 C9D4E2F2 40404040 00000000 00000000 BB677781 0AE6512D RECEIVE LU MANAGER RECEIVER
DFS707I (RRS) A0A684E2 480005A6 0CB25998 0CB25700 00010000 00010000 BB6777BB CE2AFD09 ENTER COMMIT
CTXEINT1 (RRS) A51884E3 15000000 00010000 00000000 00000000 00000000 00000000 BBCE2B81 CREATE_CONTEXT NO LCRE FLAGS
ATREINT5 (RRS) A50484E4 18000000 00010000 04000000 00000000 00000000 BB6777BB 7E71E000 BBCE2F49 EXPRESS_UR_INTRST NO LCRE FLAGS
DFS707I (RRS) A0A784E5 480005A7 0CB25998 0CB25700 00000001 00000000 BB6777BB D2E7E905 EXIT COMMIT
DFS707I (RRS) A0A84E6 480005AA 0001E4D9 7E71E000 7E71E000 01000000 01000001 D2E7E905 TKN TRACE (A0A7)
DFSRLM10 (RRS) 7B0084ED 06000000 0CB25700 7E71E000 7E71E000 01000000 01000001 00000000 RECEIVE LU MANAGER RECEIVER
DFS707I (RRS) 7B0885AB 47000000 0BB80060 40C1D6E2 0CB25998 00000000 BB6777BD 4A2DBA20 APPC/OTMA SMQ AWE server
DFS707I (RRS) 7B0285AC 02400000 00000000 0CB25700 00000000 00000000 BB6777BD 4A2DEE60 DFSAPPC MSG SWITCH PROCESSOR
DFS707I (RRS) 7B0285AD 01800000 00000002 00000000 00000000 00000000 BB6777BD 4A37C900 SYNCHRONOUS OUTPUT LU MANAGER
DFS707I (RRS) 7B0985FD 0A000001 0CB25700 7E71E000 7E71E000 01000000 01000001 00000000 LU62 SERVICES INTERFACE 1
DFS707I (RRS) A0A68656 480001A6 0CB25998 0CB25700 00020000 00010000 BB6777BD 598FF94D ENTER COMMIT
CTXSCID (RRS) A51C8657 19000000 00020000 00000000 00000000 BB6777BB CE2E8E89 BD5990AC COMMIT_UR NO LCRE FLAGS
ATRCMIT (RRS) A5018658 19000000 00020000 00080000 00000000 00000000 00000000 BD599426 COMMIT_UR NO LCRE FLAGS
DFS707I (RRS) A0A78659 480001A7 0CB25998 0CB25700 00000001 00000000 BB6777BE 761A1648 EXIT COMMIT
DFS707I (RRS) A0A865A 480001AA 0002E4D9 7E71E000 7E71E000 01000000 01000001 761A1648 TKN TRACE (A0A7)
DFS707I (RRS) A0A686C2 480005A6 0CB25998 0CB25700 00010000 00010001 BB6777C3 E18B4D00 ENTER COMMIT
CTXEINT1 (RRS) A51886C3 15000000 00010000 00000000 00000000 00000000 00000000 C3E18BC9 CREATE_CONTEXT NO LCRE FLAGS
ATREINT5 (RRS) A50486C4 18000000 00010000 04000000 00000000 BB6777C3 7E71E000 C3E18BE8 EXPRESS_UR_INTRST NO LCRE FLAGS
DFS707I (RRS) A0A786C5 480005A7 0CB25998 0CB25700 00000001 00000000 BB6777C3 E1973CA0 EXIT COMMIT
DFS707I (RRS) A0A86C6 480005AA 0001E4D9 7E71E000 7E71E000 01000002 01000001 E1973CA0 TKN TRACE (A0A7)
DFS707I (RRS) 7B0086CD 06000000 0CB25700 7E71E000 7E71E000 01000002 01000001 00000000 RECEIVE LU MANAGER RECEIVER
DFS707I (RRS) 7B088777 47000000 0BB80060 40C1D6E2 0CB25998 00000000 BB6777C4 701B928E APPC/OTMA SMQ AWE server
DFS707I (RRS) 7B028778 02400000 00000000 0CB25700 00000000 00000000 BB6777C4 701BC82E DFSAPPC MSG SWITCH PROCESSOR
DFS707I (RRS) 7B028779 01800000 00000003 00000000 00000000 00000000 BB6777C4 7026386E SYNCHRONOUS OUTPUT LU MANAGER
DFS707I (RRS) 7B0987C9 0A000001 0CB25700 7E71E000 7E71E000 01000002 01000001 00000000 LU62 SERVICES INTERFACE 1
DFS707I (RRS) A0A687D2 480001A6 0CB25998 0CB25700 00020000 00010000 BB6777C4 711B1201 ENTER COMMIT
CTXSCID (RRS) A51C87D3 19000000 00020000 00000000 00000000 BB6777C3 E18DEC00 C4711B7B COMMIT_UR NO LCRE FLAGS
ATRCMIT (RRS) A50187D4 19000000 00020000 00080000 00000000 00000000 00000000 C4711D30 COMMIT_UR NO LCRE FLAGS
DFS707I (RRS) A0A787D5 480001A7 0CB25998 0CB25700 00000001 00000000 BB6777C4 869FA88D EXIT COMMIT
DFS707I (RRS) A0A87D6 480001AA 0002E4D9 7E71E000 7E71E000 01000002 01000001 869FA88D TKN TRACE (A0A7)
CTXDINT (RRS) A519888E 0E000000 00000000 00000000 0C1833D8 19070000 00000000 C9D80F9E IDENTIFY_CONTEXT NO LCRE FLAGS
DFS707I (RRS) A0A688D8 480004A6 0CB25998 0CB25700 00010000 00010002 BB6777C9 D867E74F ENTER COMMIT
ATREINT5 (RRS) A50488D9 18000000 00010000 04000000 00000000 BB6777C9 7E71E374 C9D86849 EXPRESS_UR_INTRST NO LCRE FLAGS
DFS707I (RRS) A0A788DA 480004A7 0CB25998 0CB25700 00000001 00000000 BB6777C9 D86CE76F EXIT COMMIT
DFS707I (RRS) A0A88DB 480004AA 0001E4D9 7E71E374 7E71E374 01000000 01000001 D86CE76F TKN TRACE (A0A7)
DFS707I (RRS) 7B0088E2 06000000 0CB25700 7E71E374 7E71E374 01000000 01000001 00000000 RECEIVE LU MANAGER RECEIVER
DFS707I (RRS) 7B088976 47000000 0BB80060 40C1D6E2 0CB25998 00000000 BB6777CA 80475262 APPC/OTMA SMQ AWE server
DFS707I (RRS) 7B028977 02400000 00000000 0CB25700 00000000 00000000 BB6777CA 80479342 DFSAPPC MSG SWITCH PROCESSOR
DFS707I (RRS) 7B028978 01800000 00000004 00000000 00000000 00000000 BB6777CA 8052CE22 SYNCHRONOUS OUTPUT LU MANAGER
DFS707I (RRS) 7B0989DB 0A000001 0CB25700 7E71E374 7E71E374 01000000 01000001 00000000 LU62 SERVICES INTERFACE 1
DFS707I (RRS) A0A689E4 480001A6 0CB25998 0CB25700 00020000 00010000 BB6777CA 85F8B84C ENTER COMMIT
CTXSCID (RRS) A51C89E5 19000000 00020000 00000000 00000000 BB6777C9 D6DD9446 C485F91D COMMIT_UR NO LCRE FLAGS
ATRCMIT (RRS) A50189E6 19000000 00020000 00080000 00000000 00000000 00000000 C4865434 COMMIT_UR NO LCRE FLAGS
DFS707I (RRS) A0A78A4B 480001A7 0CB25998 0CB25700 00000001 00000000 BB6777CA AA554045 EXIT COMMIT
DFS707I (RRS) A0A8A4C 480001AA 0002E4D9 7E71E374 7E71E374 01000000 01000001 AA554045 TKN TRACE (A0A7)
CTXSWCH (RRS) A5188A72 08000000 00000000 00002000 00000000 BB6777C9 D6DD9446 CABA1624 END_CONTEXT NO LCRE FLAGS
DFS707I (RRS) A0A68AD6 480005A6 0CB25998 0CB25700 00010000 00010003 BB6777D2 3170EE6A ENTER COMMIT
CTXEINT1 (RRS) A5188AD7 15000000 00010000 00000000 00000000 00000000 00000000 D231717F CREATE_CONTEXT NO LCRE FLAGS
ATREINT5 (RRS) A5048AD8 18000000 00010000 04000000 00000000 BB6777D2 7E71E000 D2317557 EXPRESS_UR_INTRST NO LCRE FLAGS
DFS707I (RRS) A0A78AD9 480005A7 0CB25998 0CB25700 00000001 00000000 BB6777D2 317D3C2A EXIT COMMIT
1 FUNCTION WORD 0 WORD 1 WORD 2 WORD 3 WORD 4 WORD 5 WORD 6 WORD 7 PAGE 0002
-DFS707I (RRS) A0A8AD9 480005AA 0001E4D9 7E71E000 7E71E000 01000006 01000001 317D3C2A TKN TRACE (A0A7)
DFS707I (RRS) 7B008AE1 06000000 0CB25700 7E71E000 7E71E000 01000006 01000001 00000000 RECEIVE LU MANAGER RECEIVER
DFS707I (RRS) 7B088C59 47000000 0BB80060 40C1D6E2 0CB25998 00000000 BB6777D2 E44C1303 APPC/OTMA SMQ AWE server
DFS707I (RRS) 7B028C5A 02400000 00000000 0CB25700 00000000 00000000 BB6777D2 E44C6D43 DFSAPPC MSG SWITCH PROCESSOR
DFS707I (RRS) 7B028C5B 01800000 00000005 00000000 00000000 00000000 BB6777D2 E4567063 SYNCHRONOUS OUTPUT LU MANAGER
DFS707I (RRS) 7B098C8D 0A000001 0CB25700 7E71E000 7E71E000 01000006 01000001 00000000 LU62 SERVICES INTERFACE 1
DFS707I (RRS) A0A68C96 480001A6 0CB25998 0CB25700 00030000 00010000 BB6777D2 E4A89B69 ENTER COMMIT
CTXSCID (RRS) A51C8C97 19000000 00030000 00000000 00000000 BB6777D2 31746F2A D2E4A90C COMMIT_UR NO LCRE FLAGS
ATRCMIT (RRS) A5018C98 19000000 00030000 00080000 00000000 00000000 00000000 D2E4BD92 COMMIT_UR NO LCRE FLAGS
    
```

DFS	RRS	A0A78C99	480001A7	0CB25998	0CB25700	00000001	00000000	BB6777D2	EF0C5785	EXIT COMMIT		
DFS	RRS	A0AA8C9A	480001AA	0003E4D9	7E71E000	7E71E000	01000006	0100001E	EF0C5785	TKN TRACE (A0A7)		
DFS	RRS	A0A68D66	480005A6	0CB25998	0CB25700	00010000	00010004	BB6777D7	0AB06B68	ENTER COMMIT		
CTX	RRS	A5188D67	15000000	00010000	00000000	00000000	00000000	00000000	D70AB0E6	CREATE_CONTEXT	NO LCRE FLAGS	
ATRE	RRS	A5048D68	18000000	00010000	04000000	00000000	00000000	BB6777D7	7E71E000	D70AB49F	EXPRESS_UR_INTRST	NO LCRE FLAGS
DFS	RRS	A0A78D69	480005A7	0CB25998	0CB25700	00000001	00000000	BB6777D7	0ABC5A28	EXIT COMMIT		
DFS	RRS	A0AA8D6A	480005AA	0001E4D9	7E71E000	7E71E000	01000008	0100001E	0ABC5A28	TKN TRACE (A0A7)		
DFS	RRS	7B008D71	06000000	0CB25700	7E71E000	7E71E000	01000008	0100001E	00000000	RECEIVE LU MANAGER RECEIVER		
DFS	RRS	7B088EE9	47000000	0BB80060	40C1D6E2	0CB25998	00000000	BB6777D7	A738EA6F	APPC/OTMA SMQ AWE server		
DFS	Exit	7B028EEA	02400000	00000000	0CB25700	00000000	00000000	00000000	BB6777D7	A739192F	DFSAPPC MSG SWITCH PROCESSOR	
DFS	Exit	7B028EEB	01800000	00000006	00000000	00000000	00000000	00000000	BB6777D7	A744892F	SYNCHRONOUS OUTPUT LU MANAGER	
DFS	RRS	7B098F1D	0A000001	0CB25700	7E71E000	7E71E000	01000008	0100001E	00000000	LU62 SERVICES INTERFACE 1		
DFS	RRS	A0A68F26	480001A6	0CB25998	0CB25700	00020000	00010000	BB6777D7	A7D2356F	ENTER COMMIT		
CTX	RRS	A51C8F27	19000000	00020000	00000000	00000000	00000000	BB6777D7	0AB3A848	D7A7D2B2	COMMIT_UR	NO LCRE FLAGS
ATRC	RRS	A5018F28	19000000	00020000	00080000	00000000	00000000	00000000	D7A7D4D7	COMMIT_UR	NO LCRE FLAGS	
DFS	RRS	A0A78F29	480001A7	0CB25998	0CB25700	00000001	00000000	BB6777D7	B160BD29	EXIT COMMIT		
DFS	RRS	A0AA8F2A	480001AA	0002E4D9	7E71E000	7E71E000	01000008	0100001E	B160BD29	TKN TRACE (A0A7)		
CTX	RRS	A5198FD5	0E000000	00000000	00000000	0C1833D8	19070000	00000000	DD59F8EF	IDENTIFY_CONTEXT	NO LCRE FLAGS	
DFS	RRS	A0A6901F	480004A6	0CB25998	0CB25700	00010000	00010005	BB6777DD	5B634549	ENTER COMMIT		
ATRE	RRS	A5049020	18000000	00010000	04000000	00000000	00000000	BB6777DD	7E71E374	D5B63AD	EXPRESS_UR_INTRST	NO LCRE FLAGS
DFS	RRS	A0A79021	480004A7	0CB25998	0CB25700	00000001	00000000	BB6777DD	5B699BA9	EXIT COMMIT		
DFS	RRS	A0AA9022	480004AA	0001E4D9	7E71E374	7E71E374	01000002	0100001E	5B699BA9	TKN TRACE (A0A7)		
DFS	RRS	7B009029	06000000	0CB25700	7E71E374	7E71E374	01000002	0100001E	00000000	RECEIVE LU MANAGER RECEIVER		
DFS	RRS	7B0891A1	47000000	0BB80060	40C1D6E2	0CB25998	00000000	BB6777DD	E519D12A	APPC/OTMA SMQ AWE server		
DFS	Exit	7B0291A2	02400000	00000000	0CB25700	00000000	00000000	00000000	BB6777DD	E519FD2A	DFSAPPC MSG SWITCH PROCESSOR	
DFS	Exit	7B0291A3	01800000	00000007	00000000	00000000	00000000	00000000	BB6777DD	E5241B2A	SYNCHRONOUS OUTPUT LU MANAGER	
DFS	RRS	7B0991E8	0A000001	0CB25700	7E71E374	7E71E374	01000002	0100001E	00000000	LU62 SERVICES INTERFACE 1		
DFS	RRS	A0A691F1	480001A6	0CB25998	0CB25700	00050000	00010000	BB6777DD	E935F58C	ENTER COMMIT		
CTX	RRS	A51C91F2	19000000	00050000	00000000	00000000	00000000	BB6777DD	591A1C0A	DDE93669	COMMIT_UR	NO LCRE FLAGS
ATRC	RRS	A50191F3	19000000	00050000	00080000	00000000	00000000	00000000	DDE93885	COMMIT_UR	NO LCRE FLAGS	
DFS	RRS	A0A79272	480001A7	0CB25998	0CB25700	00000001	00000000	BB6777DD	FC906981	EXIT COMMIT		
DFS	RRS	A0AA9273	480001AA	0005E4D9	7E71E374	7E71E374	01000002	0100001E	FC906981	TKN TRACE (A0A7)		
CTX	RRS	A51B9299	08000000	00000000	00002000	00000000	00000000	BB6777DD	591A1C0A	DE02A604	END_CONTEXT	NO LCRE FLAGS
DFS		707I	END OF FILE ON INPUT									
DFS		708I	OPTION COMPLETE									
DFS		703I	END OF JOB									

## RRST Entries for OTMA Modules

The following diagrams show the format of OTMA trace entries for RRS related events:

TRACE ID = X'5A00'	
Word 1	- byte 0 - 2A, module number for DFSYLUS0
	- byte 1 - 01, OTMA GU was invoked
	- byte 2 - DLAFLAG1
	- byte 3 - DLAFLAG4
Word 2	- Back-end YTIB CLB address
Words 3-6	- RRS parent UR token
Word 7	- Time stamp (short)
TRACE ID = X'5A00'	
Word 1	- byte 0 - 2A, module number for DFSYLUS0
	- byte 1 - 02, Fastpaht GU was invoked
	- byte 2 - DLAFLAG1
	- byte 3 - DLAFLAG4
Word 2	- Front-end IMS YTIB CNT address
Words 3-6	- RRS parent UR token
Word 7	- Time stamp (short)
TRACE ID = X'5A00'	
Word 1	- byte 0 - 2A, module number for DFSYLUS0
	- byte 1 - 03, Back-end IMS issued a DFS2224 message
	- byte 2 - DLAFLAG1
	- byte 3 - DLAFLAG4
Word 2	- Front-end IMS YTIB CLB address (CNT address for fastpath transaction)
Words 3-6	- RRS parent UR token
Word 7	- Time stamp (short)
TRACE ID = X'5A00'	
Word 1	- byte 0 - 2D, module number for DFSYSLM0
	- byte 1 - 01, back-end XCF send succeed
	- byte 2 - AOS_FLAGS

```

|
|   Word 2           - byte 3 - 0
|   Words 3-6       - Back-end YTIB address
|   Word 7          - RRS parent UR token
|   Word 7          - Time stamp (short)
|
| TRACE ID = X'5A00'
|   Word 1          - byte 0 - 2D, module number for DFSYSLM0
|                   - byte 1 - 02, back-end XCF send failed
|                   - byte 2 - AOS_FLAGS
|                   - byte 3 - 0
|   Word 2          - Back-end YTIB address
|   Words 3-6       - RRS parent UR token
|   Word 7          - Time stamp (short)
|
| TRACE ID = X'5A00'
|   Word 1          - byte 0 - 25, module number for DFSYPSI0
|                   - byte 1 - 01, OTMA Protected Trans was submitted
|                   - byte 2 - 0
|                   - byte 3 - 0
|   Word 2          - Front-end YTIB CLB address
|   Words 3-6       - Context token
|   Word 7          - Time stamp (short)
|
| TRACE ID = X'5A00'
|   Word 1          - byte 0 - 28, module number for DFSYTIB0
|                   - byte 1 - 01, OTMA input message is about to enqueue
|                   - byte 2 - 0
|                   - byte 3 - YTIB_MSG_TYPE
|   Word 2          - Front-end YTIB CLB address
|   Words 3-6       - RRS parent UR token
|   Word 7          - Time stamp (short)
|
| TRACE ID = X'5A00'
|   Word 1          - byte 0 - 28, module number for DFSYTIB0
|                   - byte 1 - 02, response from back-end is received
|                   - byte 2 - YTIB_MSG_STATUS_3
|                   - byte 3 - YTIB_MSG_TYPE
|   Word 2          - Front-end YTIB CLB address
|   Words 3-6       - RRS parent UR token
|   Word 7          - Time stamp (short)

```

### RRS Entries Logged by OTMA Modules

```

| TRACE ID = X'5A00'
|   Word 1          - byte 0 - 2A, module number for DFSYLUS0
|                   - byte 1 - 06, XCF_good_Send or - 05, XCF_bad_Send
|   Word 2          - A(ECB)
|   Words 3-6       - LUP_UR_TOKEN (AWRRURTK)
|   Word 7          - Time stamp (short)

```

### Trace Entries Logged by RRS Related Modules

```

| TRACE ID = X'A0A6' DFSRGSF0 Entry record
|   Word 1          - byte 0 - 72, module number for DFSRGSF0
|                   - bytes 1-2 - Function, SEE AWRRFUNC
|                   - byte 3 - A6 Module entry
|   Word 2          - A(TIB)
|   Word 3          - A(ECB) - from AWRRECB
|   Word 4          - bytes 0-2 - PST number
|   Word 5          - Number of queued AWEs
|   Word 7          - Time stamp (STCK)
|
| TRACE ID = X'A0A7' DFSRGSF0 Exit record
|   Word 1          - byte 0 - 72, module number for DFSRGSF0
|                   - bytes 1-2 - Function, SEE AWRRFUNC
|                   - byte 3 - A7 Module exit
|   Word 2          - A(TIB)
|   Word 3          - A(ECB) - from AWRRECB

```

```

|   Word 4      - bytes 0-1 - Highest number of AWEs
|   - bytes 2-3 - Number greater of AWEs than TCBS
|   Word 5      - Trace return code (AWRRETCD)
|   Word 7      - Time stamp (STCK)
| TRACE ID = X'A0A8' DFSRGSF0 Error Occurred
| TRACE ID = X'A0A9' DFSRGSF0 ABEND Occurred
| TRACE ID = X'A0AA' TOKEN Tracing record
|   Word 1      - byte 0 - 72, module number for DFSRGSF0
|               - bytes 1-2 - Function, SEE AWRRFUNC
|               - byte 3 - AA - Token trace record
|   Word 2      - bytes 0-1 - PST Number
|               - bytes 2-3 - Token id (CX=Context,IN=Interest,
|               UR=Unit of recovery)
|   Words 4-7   - TOKEN

```

## 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 73 through Figure 80 on page 240 show the formats of the scheduler trace records for function codes X'41' through X'48' that are listed in Table 36 on page 194.

```

| TRACE ID = X'41'
|   word 0 - byte 1 - X'41' Scheduling starts, traced by DFSSBMP0
|           byte 2 - PST number
|           bytes 3-4 - Trace sequence number
|   word 1 - SCHD must be addressable by caller
|   word 2 - Reserved
|   word 3 - SAPCNTRL
|   words 4-5 - Reserved
|   word 6 - Module identifier
|   word 7 - Store clock value

```

Figure 73. Scheduler Trace Record Format for Function Code X'41'

```

| TRACE ID = X'42'
|   word 0 - byte 1 - X'42' Block mover, traced by DFSSBMP0,
|           DFSSBMP0, DFSSMSC0
|           byte 2 - PST number
|           bytes 3-4 - Trace sequence number
|   word 1 - byte 1 - PDIRCODE
|           byte 2 - PDIROPTC
|           byte 3 - SMBSTATS
|           byte 4 - Reserved
|   word 2 - PSTPSB
|   word 3 - PSTSMB
|   words 4-5 - Reserved
|   word 6 - Module identifier
|   word 7 - Store clock value

```

Figure 74. Scheduler Trace Record Format for Function Code X'42'



```

| TRACE ID = X'43'
|   word 0 – byte 1 - X'43' Scheduling ends
|             byte 2 - PST number
|             bytes 3-4 - Trace sequence number
|   word 1 – PSTABTRM
|   word 2 – PSTPSB
|   word 3 – SAPCNTRL
|   words 4-5 – Reserved
|   word 6 – Module identifier
|   word 7 – Store clock value

```

*Figure 75. Scheduler Trace Record Format for Function Code X'43'*

```

| TRACE ID = X'44'
|   word 0 – byte 1 - X'44' IRC started
|             byte 2 - PST number
|             bytes 3-4 - Trace sequence number
|   word 1 – SSIMCOMP
|   word 2 – Reserved
|   word 3 – SAPCNTRL
|   words 4-5 – Reserved
|   word 6 – Module identifier
|   word 7 – Store clock value

```

*Figure 76. Scheduler Trace Record Format for Function Code X'44'*

```

| TRACE ID = X'45'
|   word 0 – byte 1 - X'45' TMS00 started
|             byte 2 - PST number
|             bytes 3-4 - Trace sequence number
|   word 1 – A(PST)
|   word 2 – Sync point function code (COMMIT/P1/P2/BACKOUT)
|   word 3 – Caller of TMS00
|   word 4 – TPI (first four bytes)
|   word 5 – TPI (last four bytes)
|   word 6 – Module identifier
|   word 7 – Store clock value

```

*Figure 77. Scheduler Trace Record Format for Function Code X'45'*

```

| TRACE ID = X'46'
|   word 0 – byte 1 - X'46' TMS00 finished
|             byte 2 - PST number
|             bytes 3-4 - Trace sequence number
|   word 1 – A(PST)
|   word 2 – Sync point function code (COMMIT/P1/P2/BACKOUT)
|   word 3 – Return code
|   word 4 – TPI (first four bytes)
|   word 5 – TPI (last four bytes)
|   word 6 – Module identifier
|   word 7 – Store clock value

```

*Figure 78. Scheduler Trace Record Format for Function Code X'46'*

```

| TRACE ID   = X'47'
|   word 0 – byte 1 - X'47' APPC extract call made
|             byte 2 - PST number
|             bytes 3-4 - Trace sequence number
|   word 1 – Function code (FPRETRY/PUSER)
|   word 2 – Abend code (PSTABTRM)
|   word 3 – Return code from DFSTMR00
|   word 4 – Return code from APPC extract call
|   word 5 – Reserved
|   word 6 – Module identifier
|   word 7 – Store clock value

```

| *Figure 79. Scheduler Trace Record Format for Function Code X'47'*

```

| TRACE ID   = X'48'
|   word 0 – byte 1 - X'48' Scheduling failed
|             byte 2 - PST number
|             bytes 3-4 - Trace sequence number
|   word 1 – byte 1 - PDIRCODE
|             byte 2 - PDIROPTC
|             byte 3 - PSTSCHDF
|             byte 4 - PSTCODE1
|   word 2 – PSTPSB
|   word 3 – PSTSMB
|   words 4-5 - Reserved
|   word 6 – Module identifier
|   word 7 – Store clock value

```

| *Figure 80. Scheduler Trace Record Format for Function Code X'48'*

| Figure 81 on page 241 shows an example of a scheduler 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
|  BLOCK MOVER      4207E98A  44060000  16F90598  00800041
|                  00000000  00000000  00000000  F89569D5
|  SCHED END        4307E994  00000000  16F90598  00800001
|                  00000000  00000000  00000000  F8956BD3
|  SCHED START      4156F4D0  E2C3C8C4  16920060  00800001
|                  00000000  00000000  00000000  F89973E5
|  BLOCK MOVER      4256F4DE  44060000  170305E8  00800041
|                  00000000  00000000  00000000  F89979DA
|  SCHED END        4356F4E8  00000000  170305E8  00800001
|                  00000000  00000000  00000000  F8997B43
|  IRC START        44560737  00000000  00000000  00800001
|                  00000000  00000000  16CAF7A0  F8A95716
|  IRC START        4407077F  00000000  00000000  00800001
|                  00000000  00000000  16CAF7A0  F8A9CA44
|  SCHED START      4107078C  E2C3C8C4  15AB5060  00800001
|                  00000000  00000000  00000000  F8A9D45F
|  BLOCK MOVER      4207079A  44060000  16F90598  00800041
|                  00000000  00000000  00000000  F8A9DF19
|  SCHED END        430707A4  00000000  16F90598  00800001
|                  00000000  00000000  00000000  F8A9E0C4
|  SCHED START      417007B5  E2C3C8C4  15A48060  00800001
|                  00000000  00000000  00000000  F8AA4B87
|  BLOCK MOVER      42700804  44060000  16F91740  00800041
|                  00000000  00000000  00000000  F8AB0631
|  SCHED END        4370080E  00000000  16F91740  00800001
|                  00000000  00000000  00000000  F8AB07C2
|  IRC START        447008CE  00000000  00000000  00800001
|                  00000000  00000000  16CAF7A0  F8ABC593
|  SCHED START      417008DB  E2C3C8C4  15A48060  00800001
|                  00000000  00000000  00000000  F8ABCD0C
|  BLOCK MOVER      427008E9  44060000  16F91740  00800041
|                  00000000  00000000  00000000  F8ABD209
|  SCHED END        437008F3  00000000  16F91740  00800001

```

Figure 81. 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 9: 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 “Formatting IMS Dumps Offline” on page 154 or in *IMS Version 9: 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.

Table 43. TRACE ID = X'5F03' (Get Trace Record)

WORD 0	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	0	Caller's Return Address	Return Code	0

Table 44. TRACE ID = X'5F04' (Get Trace Record)

WORD 0	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

Table 45. TRACE ID = X'5F05' (Release Trace Record)

WORD 0	WORD 1	WORD 2	WORD 3	WORD 4	WORD 5	WORD 6	WORD 7
Control Information	Pool Name	0	Buffer Address	Address of Caller's ECB	Caller's Return Address	Return Code	Current Pool Size

## 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. Table 46 shows the general format of a latch trace entry.

Table 46. 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						

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. This field is used for latch manager trace entries to denote the latch function being traced. It is not currently used for Use Manager trace entries.

### SEQ NUM

Two-byte trace sequence number assigned by the IMS trace component.

### ENTRY TYPE

For Use Manager trace entries only: 4-byte printable character string, indicating the type of entry.

- | Words 2 through 6 contain data specific to each trace entry, as described in "Latch Manager Trace Entries" on page 243, "Use Manager Trace Entries" on page 243, and "System Locate Control Function Entries" on page 245.

## 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:*

**Entry Type:** USE

**Description:** Inuse request trace entry

```
word 1 -- 'USE'
word 2 -- Block type
word 3 -- Call ID
word 4 -- Work ID
word 5 -- Block address
word 6 -- SAP address
word 7 -- Caller's return address
```

**Entry Type:** LOK

**Description:** Lock request trace entry

```
word 1 -- 'LOK'
word 2 -- Block type
word 3 -- Call ID
word 4 -- Work ID
word 5 -- Block address
word 6 -- SAP address
word 7 -- Caller's return address
```

Entry Type: CON

Description: Connect request trace entry

```
word 1 -- 'CON'
word 2 -- Block type
word 3 -- Call ID
word 4 -- Work ID
word 5 -- Block address
word 6 -- SAP address
word 7 -- Caller's return address
```

Entry Type: MRG

Description: Merge request trace entry

```
word 1 -- 'MRG'
word 2 -- Block type
word 3 -- Call ID
word 4 -- Work ID
word 5 -- Block address
word 6 -- SAP address
word 7 -- Caller's return address
```

Entry Type: INQ

Description: Inquiry request trace entry

```
word 1 -- 'INQ'
word 2 -- Block type
word 3 -- Call ID
word 4 -- Work ID
word 5 -- Block address
word 6 -- SAP address
word 7 -- Caller's return address
```

Entry Type: NUSE

Description: Nouse request trace entry

```
word 1 -- 'NUSE'
word 2 -- Block type
word 3 -- Call ID
word 4 -- Work ID
word 5 -- Block address
word 6 -- SAP address
word 7 -- Caller's return address
```

Entry Type: NLOK

Description: Unlock request trace entry

```
word 1 -- 'NLOK'
word 2 -- Block type
word 3 -- Call ID
word 4 -- Work ID
word 5 -- Block address
word 6 -- SAP address
word 7 -- Caller's return address
```

Entry Type: NCON

Description: Disconnect request trace entry

```
word 1 -- 'NCON'
word 2 -- Block type
word 3 -- Call ID
word 4 -- Work ID
word 5 -- Block address
word 6 -- SAP address
word 7 -- Caller's return address
```

Entry Type: RCOV (SAP level)

Description: Use recovery performed at the SAP (ITASK) level trace entry

```
word 1 -- 'RCOV'
word 2 -- 'SAP'
```

```
word 3 -- Block Type
word 4 -- SAP address
word 5 -- Ø
word 6 -- Ø
word 7 -- Caller's return address
```

Entry Type: RCOV (TCB level)

Description: Use recovery performed at the TCB level trace entry

```
word 1 -- 'RCOV'
word 2 -- 'TCB'
word 3 -- Block Type
word 4 -- Ø
word 5 -- TCB address
word 6 -- Ø
word 7 -- Caller's return address
```

Entry Type: RCOV (address space level)

Description: Use recovery performed at the address space level trace entry

```
word 1 -- 'RCOV'
word 2 -- 'MEM'
word 3 -- Block Type
word 4 -- Ø
word 5 -- ASCB address
word 6 -- Ø
word 7 -- Caller's return address
```

## System Locate Control Function Entries

Entry Type: SLC0

Description: Locate a block and issue a use manager inuse call against it

```
word 1 -- 'SLC0'
word 2 -- Block Type
word 3 -- Work ID
word 4 -- Call ID
word 5 -- ''
word 6 -- SAP address
word 7 -- Caller's return address
```

Entry Type: SLC1

Description: Locate a block and issue a use manager nouse call against it

```
word 1 -- 'SLC1'
word 2 -- Block Type
word 3 -- Work ID
word 4 -- Call ID
word 5 -- ''
word 6 -- SAP address
word 7 -- Caller's return address
```

Figure 82 on page 246 shows an example of a Latch trace.

```

**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  065975F0  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 82. 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 9: Installation Volume 2: System Definition and Tailoring*.
- The /TRACE command, see *IMS Version 9: Command Reference*.
- The common trace table interface, see “Common Trace Table Interface” on page 190.
- The Offline Dump Formatter, see “Formatting IMS Dumps Offline” on page 154.
- The File Select and Formatting Print utility, see *IMS Version 9: 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.

Figure 83 on page 248 depicts the trace (low level) record format of the following functions with these subfunction codes (SC):

<u>SC</u>	<u>FUNCTION</u>
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

**FUNCTION: See above listing**

Subfunction Code: See above listing

```

word 0 -- Control information
word 1 -- A(ECB)
word 2 -- A(QTPPCB)
word 3 -- byte 1 - Current call type
           byte 2 - Prior call type
           byte 3 - (unused)
           byte 4 - (unused)
word 4 -- Caller's ID (WORD 1)
word 5 -- Caller's ID (WORD 2)
word 6 -- Unused (zero)
word 7 -- Time stamp

```

*Figure 83. Low Level Trace Record Format*

Figure 84 depicts the trace (medium level) record format of the following function with subfunction code X'21':

**FUNCTION: EXIT FROM QUEUE MANAGER**

Subfunction Code: X'21'

```

word 0 -- Control information
word 1 -- PCB Contents (WORD 1)
word 2 -- A(QTPPCB)
word 3 -- Return code
word 4 -- PCB contents (WORD 4)
word 5 -- PCB contents (WORD 5)
word 6 -- PCB contents (WORD 6)
word 7 -- Time stamp

```

*Figure 84. Medium Level Trace Record Format - X'21'*

Figure 85 depicts the trace (medium level) record format of the following function with subfunction code X'20':

**FUNCTION: ENTRY TO QUEUE MANAGER**

Subfunction Code: X'20'

```

word 0 -- Control information
word 1 -- PCB Contents (WORD 1)
word 2 -- A(QTPPCB)
word 3 -- PCB contents (WORD 3)
word 4 -- PCB contents (WORD 4)
word 5 -- PCB contents (WORD 5)
word 6 -- PCB contents (WORD 6)
word 7 -- Time stamp

```

*Figure 85. Medium Level Trace Record Format - X'20'*

This figure depicts the trace (medium level) record format of the following function with subfunction code X'22':

**FUNCTION: Special- Not Applicable**

Subfunction Code: X'22'

```
word 0 -- Control information
word 1 -- Varies by use
word 2 -- Varies by use
word 3 -- Varies by use
word 4 -- Varies by use
word 5 -- Varies by use
word 6 -- Varies by use
word 7 -- Time stamp
```

*Figure 86. Medium Level Trace Record Format - X'22'*

Figure 87 depicts the trace (low level) record format of the following functions with these subfunction codes (SC):

**SC      FUNCTION**

**X'08'**

**X'15'**    MESSAGE REROUTE

**X'1B'**    INSERT MOVE SPANNABLE

**FUNCTION:** See above list

Subfunction Code: See above list

```
word 0 -- Control information
word 1 -- A(ECB)
word 2 -- A(QTPPCB)
word 3 -- byte 1 - Current call type
           byte 2 - Prior call type
           byte 3 - (unused)
           byte 4 - (unused)
word 4 -- Caller's ID (WORD 1)
word 5 -- Caller's ID (WORD 2)
word 6 -- byte 1 - Length of user segment
           byte 2 - Length of user segment
           byte 3 - (unused)
           byte 4 - (unused)
word 7 -- Time stamp
```

*Figure 87. Low Level Trace Record Format - X'08', X'15', X'1B'*

Figure 88 on page 250 depicts the trace (low level) record format of the following function with subfunction code X'0A':

**FUNCTION:** INSERT LOCATE  
Subfunction Code: X'0A'

```

word 0 -- Control information
word 1 -- A(ECB)
word 2 -- A(QTPPCB)
word 3 -- byte 1 - Current call type
           byte 2 - Prior call type
           byte 3 - (unused)
           byte 4 - (unused)
word 4 -- Caller's ID (WORD 4)
word 5 -- Caller's ID (WORD 2)
word 6 -- Length of requested message area
word 7 -- Time stamp

```

Figure 88. Low Level Trace Record Format - X'0A'

Figure 89 depicts the trace (low level) record format of the following function with subfunction code X'16':

**FUNCTION:** RELEASE  
Subfunction Code: X'16'

```

word 0 -- Control information
word 1 -- A(ECB)
word 2 -- A(QTPPCB)
word 3 -- byte 1 - Current call type
           byte 2 - Prior call type
           byte 3 - (unused)
           byte 4 - (unused)
word 4 -- Caller's ID (WORD 1)
word 5 -- Caller's ID (WORD 2)
word 6 -- Contents of DECArea
word 7 -- Time stamp

```

Figure 89. Low Level Trace Record Format - 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 SQT 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 SQT 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 eye catcher \*\*SQTT.

To display the status of the trace, use the /DISPLAY TRACE command

**Related Reading:** For information about:

- The SQTT parameter, see *IMS Version 9: Installation Volume 2: System Definition and Tailoring*.
- The /TRACE command, see *IMS Version 9: Command Reference*.
- The common trace table interface, see “Common Trace Table Interface” on page 190.
- The Offline Dump Formatter, see “Formatting IMS Dumps Offline” on page 154.
- The File Select and Formatting Print utility, see *IMS Version 9: Utilities Reference: System*.

## Fast Path Trace

When you use the /TRACE SET ON TABLE FPTT command, IMS enables the Fast Path trace. The Fast Path trace will reside in the internal IMS trace tables, with the OPTION LOG causing the trace to also be written to the IMS logs. If the OPTION LOG parameter is not specified (or the OPTION NOLOG is specified), the trace will only reside in the IMS internal trace tables and is formatted through the IMS dump formatter. If the OPTION LOG parameter is specified, the trace will also reside on the logs and can be formatted with DFSERA60 for log type X'67FA' or through the IMS dump formatter.

## Trace Formats

Fast Path reserves X'9C' and X'9D' trace entries. X'9C' is reserved for tracing notifies and X'9D' is reserved for all other Fast Path traces:

- “X'9C' Trace Format”
- “X'9D' Trace Format” on page 252

**X'9C' Trace Format:** Table 47 shows the format of the X'9C' trace entry and below the table are the trace IDs and descriptions of content of trace entry.

Table 47. Format of the Fast Path X'9C' Trace Entry

Word 0	Word 1	Word 2	Word 3	Word 4	Word 5	Word 6	Word 7
aabbcccc	dddddddd	dddddddd	dddddddd	dddddddd	dddddddd	dddddddd	dddddddd

## Trace ID

## Description of Content of Trace Entry

<b>aa</b>	The FP Notify trace code, X'9C'
<b>bb</b>	The trace subcode, used so that each IMSFP trace entry has a unique code or subcode, avoiding duplication of the same trace function.
<b>X'01'</b>	DBFNOTM0 Entry
<b>X'02'</b>	NCB contents at entry to DBFNOTM0
<b>X'03'</b>	DBFNOTM0 NOTEXC (DFSLM->IRLM)
<b>X'04'</b>	DBFNOTM0 IWAIT
<b>X'05'</b>	DBFNOTM0 after IWAIT

- | X'06' DBFICLI0 Entry
- | X'07' NCB contents at entry to DBFICLI0
- | X'08' DBFICLI0 Response decrement EPSTNCTR
- | X'09' DBFICLI0 IPOST
- | X'0A' DBFCSTS2 EPST Timeout Candidate
- | X'0B' DBFCSTS2 EPST Timeout IPOST
- | cccc The Trace Sequence Number
- | dddddddd Data, specific for each trace entry.

| **X'9D' Trace Format:** Table 48 shows the format of the X'9D' trace entry and below the table are the trace IDs and descriptions of content of trace entry.

| *Table 48. Format of the Fast Path X'9D' Trace Entry*

Word 0	Word 1	Word 2	Word 3	Word 4	Word 5	Word 6	Word 7
aabbcccc	dddddddd	dddddddd	dddddddd	dddddddd	dddddddd	dddddddd	dddddddd

- | Trace ID | Description of Content of Trace Entry |
|----------|---------------------------------------|
| aa       | The FP General trace code, X'9D'      |
| bb       | The FP latch trace subcodes           |
| X'01'    | DBFELOCK DMAC                         |
| X'02'    | DBFELOCK DMCB                         |
| X'03'    | DBFELOCK DSM                          |
| X'04'    | DBFELOCK FLD                          |
| X'05'    | DBFELOCK FNCB                         |
| X'06'    | DBFELOCK MSDB                         |
| X'07'    | DBFELOCK TRAT                         |
| X'08'    | DBFELOCK VSO                          |
| X'09'    | DBFELOCK VSTR                         |
| X'0A'    | DBFELOCK XCRB                         |
| X'10'    | Resource Latch                        |
| X'11'    | DBFSYNL Latch                         |
| X'12'    | DBFBUFL Latch                         |
| X'13'    | DBFEMHBL Latch                        |
| X'14'    | DBFLATCH Latch                        |
| X'15'    | DBFALOCK Latch                        |
| X'16'    | DBFHLOCK Latch                        |
| X'17'    | DBFPLOCK Latch                        |
| cccc     | The Trace Sequence Number             |
| dddddddd | Data, specific for each trace entry   |

## Chapter 8. DB—Database Service Aids

The information contained in this section addresses service aids and diagnostic techniques used to analyze IMS database problems. Specifically, this section addresses the following items:

- | • “The Job Control Block (JCB) Trace” traces the last few DL/I calls and related status codes for a specific logical database.
- | • “Data Language/I Test Program—DFSDDL0” on page 255 is used to test DL/I calls against a given database.<sup>a</sup>
- | • “COMPARE Statement SNAPs” on page 255 discusses the COMPARE statement SNAP and the output from SNAP calls.<sup>a</sup>
- | • “SNAPs on Exceptional Conditions” on page 256 discusses SNAPs on exceptional conditions.<sup>a</sup>
- | • “DL/I Call Image Capture” on page 257 discusses 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.
- | • “DL/I Analysis” on page 258 discusses a technique for approaching DL/I analysis in a batch environment.
- | • “Locating Database-Related Traces” on page 262 discusses locating database related traces.
- | • “DL/I Trace” on page 263 provides a description of the DL/I record formats.
- | • “Retrieve Trace” on page 301 discusses the Retrieve trace that records the control flow between the retrieve module and other database routines.
- | • “Program Isolation-Related Problem Analysis” on page 311 discusses program isolation-related problem analysis.
- | • “Log Analysis (Database Related)” on page 312 discusses IMS log record analysis.
- | • “Sequential Buffering Service Aids” on page 316 discusses diagnostic tools that are of use when you receive a message or abend that indicates a problem with Sequential Buffering.
- | • “GSAM Control Block Dump—DFSZD510” on page 319 discusses GSAM control blocks dump.<sup>a</sup>

**Note:** <sup>a</sup> 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 the offset of JCBTRACE 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 49 on page 254). 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 of the JCB, and the status code of the prior call is in field JCBPREVR.

**Related Reading:** The DL/I status codes and return codes are defined in the topic titled “DL/I Codes” in *IMS Version 9: Messages and Codes, Volume 1*.

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'.

The following topics provide additional information:

- “Sample JCB Trace”
- “JCB Trace Call Function Codes”

## Sample JCB Trace

A sample JCB dump is shown in Figure 90.

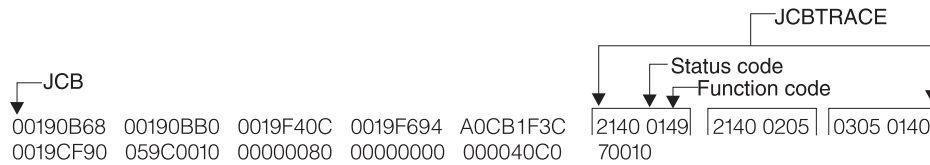


Figure 90. Example of a Job Control Block (JCB) Dump

## JCB Trace Call Function Codes

The DL/I user call encoded functions are contained in DFSDLA00, at label FUNCSTRT. They are listed in Table 49.

Table 49. 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



Table 49. DL/I User Call Encoded Functions (continued)

Code	Call	Code	Call
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 9: Application Programming: Database Manager*.

---

## Data Language/I Test Program—DFSDDLTO

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 section on testing an application program in *IMS Version 9: 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 9: 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 9: Application Programming: Database Manager*.

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 and buffer prefix and subpools
  - 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, and level table
  - 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 (DFSERA10) with exit routine, DFSERA30. For information about this utility, see *IMS Version 9: 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 9: Utilities Reference: System*.

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 152).

---

## 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 DFSDDLTO.

**Related Reading:** For information about DFSDDLTO, see *IMS Version 9: 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 DB/DC environment.

| The following topics provide additional information:

- | • “Batch Environment” on page 258
- | • “Online Environment” on page 258
- | • “How to Retrieve DL/I Call Image Capture Data from the Log Data Set” on page 258

## 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 9: Installation Volume 2: System Definition and Tailoring*.
- Using a call image capture statement to trace DL/I calls, see *IMS Version 9: 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 9: 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 9.
- Allocating the external trace data sets (DFSTRA01 and DFSTRA02) used by the IMS online systems, see *IMS Version 9: Installation Volume 1: Installation 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 9: 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:

- | **Note:** In a pseudoabend SVC dump generated by DFSERA20, you can find the failing PST by searching  
| the save areas for the caller of DFSERA20. In the save area flow, DFSERA20 is called INTERA20  
| and register 1 contains the failing PST address.
- The first request block (RB) on the RB chain represents the IMS batch region controller (DFSRRRC00); 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 z/OS.
- 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 20-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.

**Note:** When pseudoabends are detected by some modules, the registers 14 to 12 at error, are stored at PSTSAVL+12. The high order byte in PSTSAVL+12 will contain a one-byte code for the module detecting the error. Here are the modules which will save registers and their corresponding codes in PSTSAVL+12:

	<b>X'AA'</b>	DFSDLR00
	<b>X'BB'</b>	DFSDDLE0
	<b>X'CC'</b>	DFSDLD00
	<b>X'DD'</b>	DFSDXMT0
	<b>X'EE'</b>	DFSURGU0
	<b>X'FF'</b>	DFSRCHB0

Here is an example from the formatted PST of an abend U0853:

```
WD1 00000000 HSA 202C6BC8 LSA 2CD73B08 RET AA049128 EPA 30B02F40 R0 30000355
R1 212AD040 R2 2CD78790 R3 2FB6F5B4 R4 8004911E R5 2FB6FA8C R6 01410254
R7 21748060 R8 2FB6F82C R9 00000002 R10 30B053C0 R11 000401E0 R12 00047DC0
```

Since "RET" (PSTSAVL+12) contains 'AA'x, module DFSDLR00 detected this pseudoabend.

## 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 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 260 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 at label JCBTRACE in the JCB are six 2-byte trace entries for the last six calls issued against this PCB. The oldest entry is at the beginning and the newest entry is at the end of JCBTRACE. 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 90 on page 254.)
5. Look at the contents of JCBLEV1C. 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, JDBLEV1C 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, LEVSEGOFF 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 LEVSEGOFF. 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. This field is a pointer to the parent SDB (the root SDB points at the PCB). (See Figure 41 on page 116 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. 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 129 on page 300.) 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. 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 41 on page 116 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 258. Figure 91 on page 263 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.



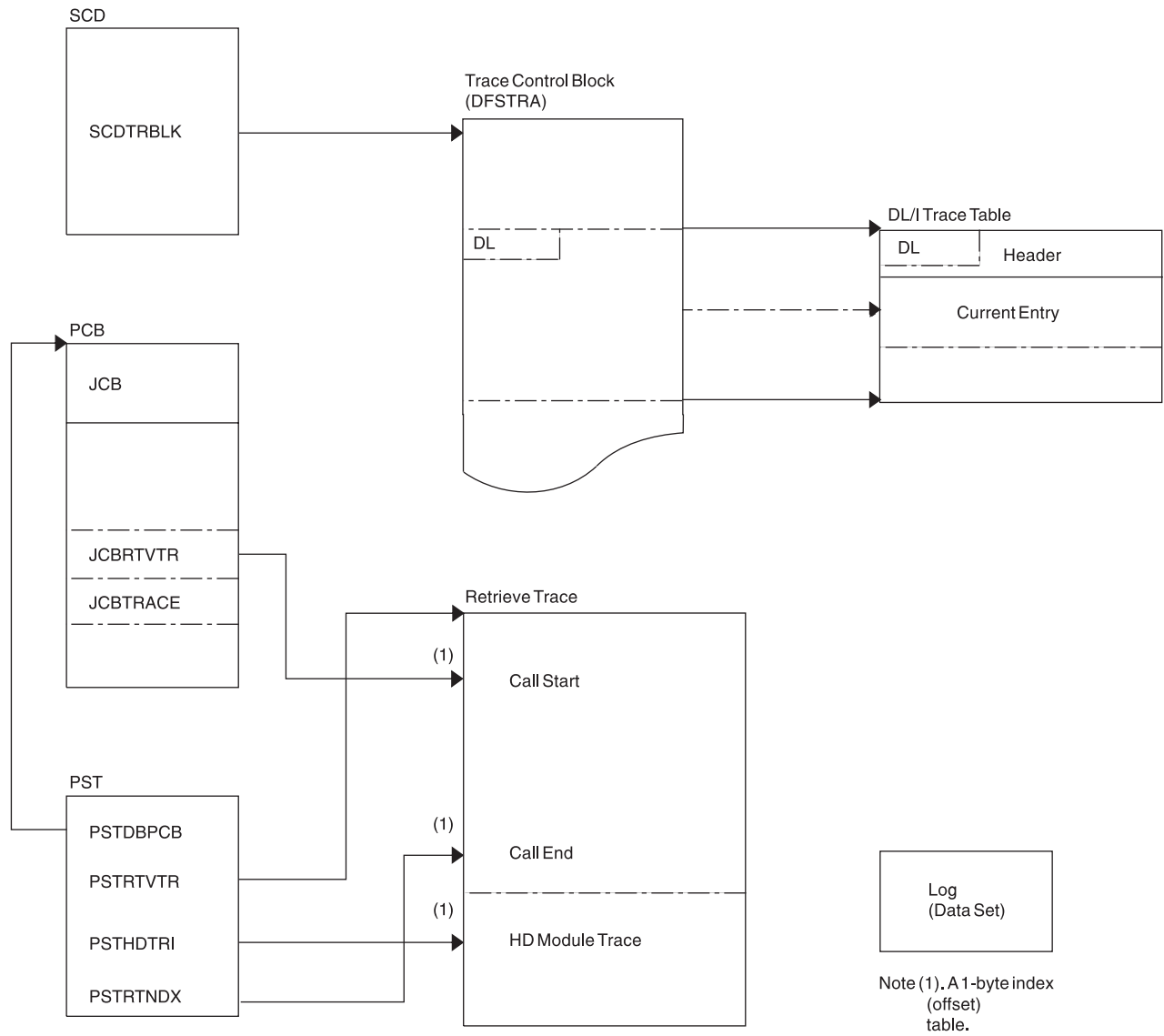


Figure 91. How to Locate the Database Traces

## 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, HALDB OLR trace, 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 190. This section also lists the function codes for the DL/I and lock traces.

**Note:** IMS always turns on the DL/I and lock trace at initialization. The trace level is set to high and it is written in memory.

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 can use either of the following for formatting and processing of the DL/I trace:

- The Knowledge-Based Log Analysis (KBLA) panel interface. For more information, see the section titled “Knowledge-Based Log Analysis” in the *IMS Version 9: Utilities Reference: System*.
- The File Select and Formatting Print utility (DFSERA10) with an exit routine (DFSERA40 or DFSERA60).

The Database Tracking trace entries are described in “X'D4': Database Tracker Trace Entries (D4)” on page 489.

- | The following topics provide additional information:
  - | • “Using the DL/I Trace”
  - | • “DL/I Trace Formats”
  - | • “DELETE/REPLACE—DL/I Trace Information” on page 300

## 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 9: 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 9: 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 113 on page 281). 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 (“X'0C' Trace Entry” on page 266).

<b>X'31'</b>	HD space management: Get space for the segment (“X'31’, X'32’, X'34’, X'B1’, and X'B2’ Trace Entries” on page 266).
<b>X'32'</b>	HD space management: Free space for the segment (“X'31’, X'32’, X'34’, X'B1’, and X'B2’ Trace Entries” on page 266).
<b>X'34'</b>	HD space management: Get space close to root anchor point (“X'31’, X'32’, X'34’, X'B1’, and X'B2’ Trace Entries” on page 266).
<b>X'60'</b>	OSAM I/O initiated trace entry (“X'60’ and X'61’ Trace Entries” on page 267).
<b>X'61'</b>	OSAM I/O posted trace entry (“X'60’ and X'61’ Trace Entries” on page 267).
<b>X'62'</b>	OSAM trace entry for OPEN/CLOSE/EOV trace entries (“X'62’ Trace Entry” on page 268).
<b>X'69'</b>	Sequential Buffering buffer invalidation trace entry (Figure 96 on page 269).
<b>X'6A'</b>	Sequential Buffering buffer evaluation trace entry (Figure 97 on page 270).
<b>X'6B'</b>	Indicates Sequential Buffering at program termination. (Figure 98 on page 270).
<b>X'6C'</b>	Indicates Sequential Buffering refreshing of buffers. (Figure 99 on page 271).
<b>X'6F'</b>	Sequential Buffering search by RBA issued by OSAM BH trace entry (Figure 100 on page 272).
<b>X'80'</b>	Database authorization request (Figure 101 on page 272).
<b>X'81'</b>	Database change authorization request (Figure 101 on page 272).
<b>X'82'</b>	Database re-authorization request (Figure 101 on page 272).
<b>X'AA'</b>	Analyzer entry (“X'AA’ Trace Entry” on page 272).
<b>X'AB'</b>	ABEND U0427 trace entry (Figure 103 on page 273).
<b>X'AC'</b>	Database call analyzer entry (DBCTL only) (“X'AC’ Trace Entry” on page 275).
<b>X'B1'</b>	HD space management: Get space set by backout or DELETE/REPLACE (“X'31’, X'32’, X'34’, X'B1’, and X'B2’ Trace Entries” on page 266).
<b>X'B2'</b>	HD space management: Free space set by backout (“X'31’, X'32’, X'34’, X'B1’, and X'B2’ Trace Entries” on page 266).
<b>X'C4'</b>	DELETE/REPLACE (“X'C4’ Trace Entry” on page 275).
<b>X'C6'</b>	Special promote lock trace entry. Descriptions available by assembling IDLIVSAM TRACENT.
<b>X'C7'</b>	Exclusive control deadlock detection trace entry (with and without IRLM in “X'C7’ Trace Entry Using IRLM” on page 277).
<b>X'C8'</b>	Lock request manager entry (DFSLMGR0) (“X'C8’ Trace Entry” on page 278).
<b>X'C9'</b>	Lock request manager exit (DFSLMGR0) (“X'C9’ Trace Entry” on page 278).
<b>X'CA'</b>	Exclusive control ENQ/DEQ (program isolation) entry (for non-Fast Path and Fast Path, see “X'CA’ Trace Entry” on page 279).
<b>X'CA'- X'08'</b>	PI DL/I call trace entry (“X'CA’ through X'08’ Trace Entry” on page 281).
<b>X'CB'</b>	PI trace lock elapsed time (“X'CB’ Trace Entry” on page 281).
<b>X'CC'</b>	Lock request handler (DFSLRH00) entry (“X'CC’ Trace Entry” on page 282).
<b>X'CF'</b>	I/O toleration (DFSTOPR0) entry. Descriptions available by assembling IDLIVSAM TRACENT.
<b>X'D0'</b>	IRLM notify sent trace entry (Figure 117 on page 286).
<b>X'D1'</b>	IRLM notify received trace entry (Figure 118 on page 287).

- I **X'D4'** RSR DTT trace entry. See Table 140 on page 489.
- I **X'D9'** HALDB Online Reorganization (OLR) trace entry ("X'D9' Trace Entry" on page 287).
- X'DA'** VSAM JRNAD or UPAD exit ("X'DA' Trace Entry" on page 294).
- X'DB'- X'FA'** Buffer handler trace ("X'DB' through X'FA' Trace Entry" on page 295).

### **X'0C' Trace Entry**

Figure 92 shows the X'0C' trace entry.

**TRACE ID = X'0C'**

```

word 0 – byte 1 - X'0C' - DL/I OPEN/CLOSE trace entry for each
           data set. This entry shows a successful OPEN/CLOSE.
           For an error during OPEN/CLOSE, the data in ENTRY6 and
           ENTRY7, X'18' and X'1C' respectively, is shown in the
           "error condition"
           byte 2 - PST number
           bytes 3-4 - Trace sequence number description.
word 1 – byte 1 - PSTFNCTN (See note below)
           bytes 2-3 - DMB number
           byte 4 - DCB number
word 2 – DCB address
word 3 – DD name
word 4 – DD name
word 5 – PSTDBPCB - database PCB address
word 6 – DMB address (Error condition) - Offset in DFSDLOC0 where error was
           detected.
word 7 – bytes 1-3 - PSTPSB-PSB address - database PCB address (Error
           condition) - Word "LKER" or reason codes described in
           message DFS07301
           byte 4 - Not used

```

*Figure 92. X'0C' Trace Entry*

**Note to Figure 92:** Use the "Open/Close Function Codes" section of Table 63 on page 296.

### **X'31', X'32', X'34', X'B1', and X'B2' Trace Entries**

Figure 93 on page 267 shows the X'31', X'32', X'34', X'B1', and X'B2' trace entries.

**TRACE ID = X'31', X'32', X'34', X'B1', X'B2'**

word 0 – byte 1 - X'31', X'32', X'34', X'B1', or X'B2' - Function code for HD space management (see note 1 below)  
 byte 2 - PST number  
 bytes 3-4 - Trace sequence number  
 word 1 – bytes 1-2 - Length of request (see note 3 below)  
 bytes 3-4 - Offset (requested or returned)  
 word 2 – byte 1 - PSTTRNID (ID of module calling space management)  
 byte 2 - PSTTRMSC (subcode of module calling the buffer handler - see note 4 below)  
 byte 3 - Not used  
 byte 4 - PSTRTCDE (return code from space management)  
 word 3 – byte 1 - Flag byte (X'80' - entry already in use)  
 bytes 2-4 - PSTDATA (core address - see note 5 below)  
 word 4 – PSTBYTNM (RBA or RRN - see note 6 below)  
 word 5 – RBA of space given to caller  
 word 6 – bytes 1-2 - DMB number  
 byte 3 - DCB number  
 byte 4 - Reserved  
 word 7 – MSG/ABEND feedback

Figure 93. X'31', X'32', X'34', X'B1', and X'B2' Trace Entries

**Notes to Figure 93:**

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 9: Administration Guide: Database Manager* for an explanation of FSEs.)
4. See Table 65 on page 299 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.

**X'60' and X'61' Trace Entries**

Figure 94 on page 268 shows the X'60' and X'61' trace entries.

**TRACE ID = X'60' or X'61'**

word 0 – byte 1 - X'60' or X'61'- OSAM I/O initiated trace entry  
           byte 2 - Zero (no PST number)  
           bytes 3-4 - Trace sequence number

word 1 – IOSB address

word 2 – DCB address

word 3 – DECB address

word 4 – byte 1 - For POST this is the completion code (X'7F',  
                   X'41', and so on)  
           byte 2 - For POST this is the count of I/O initiated  
           bytes 3-4 - For POST (61) this is the trace sequence  
                   number of I/O INIT (60)

word 5 – RBN or EXTENT number

word 6 – Not used

word 7 – byte 1 - Operation function code  
           byte 2-4 - Not used

*Figure 94. X'60' and X'61' Trace Entries***X'62' Trace Entry**

Figure 95 shows the X'62' trace entry.

**TRACE ID = X'62'**

word 0 – byte 1 - X'62' - OSAM trace entry for OPEN/CLOSE/EOV  
           trace entries  
           byte 2 - Zero (no PST number)  
           bytes 3-4 - Trace sequence number

word 1 – Not used

word 2 – DCB address

word 3 – DCBRELAD

word 4 – byte 1 - Not used  
           byte 2 - R15 return code  
           bytes 3-4 - Not used

word 5 – OPEN/CLOSE/EOV error code (the same as in message DFS07301)

word 6 – Not used

word 7 – byte 1 - Caller's function (see below)  
           byte 2-4 - Not used

*Figure 95. X'62' Trace Entry***X'69' Trace Entry**

Figure 96 on page 269 shows the X'69' trace entry.



**TRACE ID = X'6A'**

word 0 – byte 1 - X'6A' - Sequential Buffering buffer evaluation trace entry  
           byte 2 - PST number  
           bytes 3-4 - Trace sequence number  
 word 1 – bytes 1-2 - DMB number  
           - byte 3 - DCB number  
           - byte 4 - Not used  
 word 2 – byte 1 - Type of evaluation (see note 1)  
           - byte 2 - Not used  
           - byte 3 - Result of evaluation of sequentially (see note 2)  
           - byte 4 - Result of evaluation of I/O rate (see note 2)  
 word 3 – DSG address  
 word 4 - SBPSTCNB (=SBH CALL NUMBER THIS PST)  
 word 5 - byte 1 - Not used  
           - bytes 2-4 - Threshold cost for SB logic  
 word 6     - byte 1 - Not used  
           - bytes 2-4 - Current cost of SB logic  
 word 7 - bytes 1-2 - Threshold value for I/O rate  
           - bytes 3-4 - Current value of I/O rate

Figure 97. X'6A Trace Entry

| **Note:**

- |       1.
- |             |                       |
|-------------|-----------------------|
| <b>C'P'</b> | Periodical evaluation |
| <b>C'E'</b> | Early evaluation      |
- |       2.
- |             |                        |
|-------------|------------------------|
| <b>C'P'</b> | Evaluation is positive |
| <b>C'N'</b> | Evaluation is negative |

| **X'6B' Trace Entry**

| Figure 98 shows the X'6B' trace entry.

**TRACE ID = X'6B'**

word 0 – byte 1 - X'6B' - Indicates why SB was or was not used  
           byte 2 - PST number  
           bytes 3-4 - Trace sequence number  
 word 1 – C'TERM'  
 word 2 – byte 1 - SCDSBFL - Sequential buffering flag (see note 1)  
           - byte 2 - Resource allocation failure (see note 2)  
           - byte 3 - Info from user exit routine (see note 3)  
           - byte 4 - SBPSTITR - Termination trace flag (see note 4)  
 word 3 – Not used  
 words 4-5 – Job name  
 words 6-7 – PSB name

Figure 98. X'6B Trace Entry

| **Note:**

- |       1.
- |              |  |
|--------------|--|
| <b>X'80'</b> | ...SCDSBNSB: DON'T LOAD SB MODULES             |
| <b>X'20'</b> | ...SCDSBLER: ERROR WHILE LOADING SB MODULES SB |
| <b>X'10'</b> | ...SCDSBOER: OTHER SB ERRORS                   |



	2.	
	<b>X'80'</b>	...SBPSTGM1: GM ERROR FOR CB OR WORKAREA
	<b>X'40'</b>	...SBPSTGM2: GM ERROR FOR SBH BUFFERS
	<b>X'20'</b>	...SBPSTGM3: MAXSB= LIMIT EXHAUSTED
	<b>X'10'</b>	...SBPSTGM4: MAX NBR OF IOSB EXHAUSTED
	<b>X'08'</b>	...SBPSTGM5: GETIOSB FAILURE
	<b>X'04'</b>	...SBPSTGM6: PAGE-FIX ERROR
	<b>X'02'</b>	...SBPSTGM7: I/O-ITASK INIT FAILURE
	<b>X'01'</b>	...SBPSTGM8: GM ERROR FOR CB OR WORKAREA
	3.	
	<b>X'80'</b>	...SBPRMPDI: DISALLOW USAGE OF SB
	<b>X'40'</b>	...SBPRMPAD: CONDITIONAL ACTIV BY DEFAULT
	4.	
	<b>X'80'</b>	...SBPSTITP: USER PROVIDED SB= KEYW IN PSBGEN
	<b>X'40'</b>	...SBPSTITC: SBPARM CARD PROCESSED
	<b>X'01'</b>	...SBPSTITS: /STOP SB ISSUED BY MTO

**X'6C' Trace Entry**

Figure 99 shows the X'6C' trace entry.

**TRACE ID = X'6C'**

- word 0 – byte 1 - X'6C' - Indicates if Sequential buffering was used trace entry
  - byte 2 - PST number
  - bytes 3-4 - Trace sequence number
- word 1 – bytes 1-2 - DMB number
  - byte 3 - DCB number
  - byte 4 - Not used
- word 2 – bytes 1-2 - Number of refreshed SBH buffers
  - bytes 3-4 - Number of invalidated SBH buffers
- word 3 – DSG address of owner of the last touched SBH buffer or zero
- word 4 – byte 1 - SBPSTGS – Global serialization trace (see note below)
  - bytes 2-4 - Not used
- word 5 – OSAM BH prefix address
- word 6 – SBH buffer CB address of last touched buffer or zero
- word 7 – Block number

*Figure 99. X'6C' Trace Entry*

**Note:**

	<b>X'80'</b>	Global serialization entered (SBH search started).
	<b>X'40'</b>	Waiting PST was posted.

**X'6F' Trace Entry**

Figure 100 on page 272 shows the X'6F' trace entry.

**TRACE ID = X'6F'**

word 0 – byte 1 - X'6F' - Sequential Buffering search by RBA issued  
by OSAM BH trace entry  
byte 2 - PST number  
bytes 3-4 - Trace sequence number  
word 1 – bytes 1-2 - DMB number  
- byte 3 - DCB number  
- byte 4 - Last byte of return code from OSAM BH  
word 2 – First trace word within SDSG  
word 3 – DSG address  
word 4 – Second trace word within SDSG  
word 5 – OSAM BH prefix address  
word 6 – SBH buffer CB address  
word 7 – Block number

*Figure 100. X'6F' Trace Entry***X'80', X'81', X'82' Trace Entries**

Figure 101 shows the X'80', X'81', and X'82' trace entries.

**TRACE ID = X'80', X'81', X'82'**

word 0 – byte 1 - X'80', X'81', X'82' - Database authorization,  
change-authorization, and re-authorization request  
byte 2 - PST number  
bytes 3-4 - Trace sequence number  
words 1-2 – AURDBDNM - database name  
word 3 – byte 1 - AURACC – database access  
- byte 2 - AURECD – authorized encoded state  
- byte 3 - AURSLV – database share level  
- byte 4 - AURWRKC – authorization work field  
word 4 - bytes 1-2 - AURDMBNO – Global DMB number  
- bytes 3-4 - AURERRCD – DBRC error reason code  
word 5 - AURSYSID - IMS online subsystem id  
word 6 - AURDDIRA – DDIR address  
word 7 - AURDSGCH – DSG address of last in DSG chain  
or  
word 7 - TCB number for restart authorization by DFSRDA00

*Figure 101. X'80', X'81', X'82' Trace Entry***X'AA' Trace Entry**

Figure 102 on page 273 shows the X'AA' trace entry.

**TRACE ID = X'AA'**

word 0 – byte 1 - X'AA' - Analyzer entry - This entry is created for each call passed to DFSDLA00. All entries are the internal activities in IMS that take place as a result of the user call. Be sure to use only the entries with the same PST number as the one identified as the failing PST.  
 byte 2 - PST number (see note 1 below)  
 bytes 3-4 - Trace sequence number

word 1 – Address of user parameter list (this list consists of all entries up to and including the entry with a X'80' in the high-order byte of a word.)

word 2 – Call function for current call (GU, GN and so on - see note 2 below)

word 3 – PCB address for current call

words 4-5 – If DB PBC, LEVLEV thru LEVSEGOF (first 10 bytes of level table for level of segment returned on prior call) IF TP PCB, character string is TP CALL

word 6 – bytes 1-2 - If DB PBC, LEVLEV thru LEVSEGOF (first 10 bytes of level table for level of segment returned on prior call) IF TP PCB, character string is TP CALL  
 bytes 3-4 - Status code in PCB from prior call (see note 3 below)

word 7 – LEVSDB - SDB address for level of segment returned on prior call

*Figure 102. X'AA' Trace Entry*

**Notes to Figure 102:**

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.

**X'AB' Trace Entry**

Figure 103 shows the X'AB' trace entry.

**TRACE ID = X'AB'**

word 0 – byte 1 - X'AB' - ABEND U0427 trace entry  
 byte 2 - PST number  
 bytes 3-4 - X'0427'

word 1 – byte 1 - PSTFNCTN – Buffer handler function code  
 - byte 2 - RPLREQ  
 - bytes 3-4 – Trace sequence number

word 2 – bytes 1-2 - Offset to abend within DFSDVSM0  
 - byte 3 - DSGINDA - data set information (see note 1)  
 - byte 4 - DSGINDB – caller information (see note 2)

word 3 - RPLI address (Register 8)

word 4 - RPLARG - VSAM argument

word 5 - RPLAREA - VSAM area pointer

word 6 - byte 1 - RPLERREG - VSAM return code  
 - byte 2 - RPLERRCD - VSAM error code  
 - byte 3 - RPLOPT1 - VSAM request option (see note 3)  
 - byte 4 - RPLOPT2 - VSAM request option (see note 4)

word 7 – AMP address

*Figure 103. X'AB' Trace Entry*

**Notes to Figure 103:**

1. See Table 50 for data set information:

Table 50. Data Set Information

Code (Hex)	DSGINDA	Definition
X'80'	DSGDSOLS	This is the last DSG in JCB.
X'44'	DSGDSORI	Data set group is root in index.
X'20'	DSGDSOHD	Data set group is HDAM.
X'10'	DSGDSOHI	Data set group is HDAM.
X'08'	DSGDSOH2	Data set group is HISAM case 2.
X'04'	DSGDSOH1	Data set group is HISAM.
X'02'	DSGDSOHS	Data set group is HSAM or SSAM.
X'01'	DSGVSAM	Data set group is VSAM.

2. See Table 51 for caller information:

Table 51. Caller Information

Code (Hex)	DSGINDB	Description
X'80'	DSGSETLR	From SETL routine for SYNAD routine.
X'40'	DSGGETR	From GET routine for SYNAD routine.
X'20'	DSGBATIS	Record returned is batch, DSGIRECA is actual address.
X'10'	DSGNXTIS	Next sequential root is current keyed record.
X'08'	DSGSETL2 DSGSETK2	Second SETL has been issued. Move key to DSG high key area.
X'04'	DSGGETGT	A GET in BISAM being done using a SETL GT.
X'02'	DSGKEYSR	Buffer pool has been searched for key.
X'01'	DSGSTLIS	This is STL for INSERT.

3. See Table 52 for VSAM request option 1:

Table 52. VSAM Request Option

Code (Hex)	RPLOPT1	Description
X'80'	RPLLOC	Locate mode.
X'40'	RPLDIR	Direct processing.
X'20'	XRPLSEQ	Sequential.
X'10'	RPLSKP	Skip SEQ access.
X'08'	RPLASY	Asynchronous.
X'04'	RPLKGE	Search key GT/EQ.
X'02'	RPLGEN	Generic key request.
X'01'	RPLECBSW	External ECB.

4. See Table 53 for VSAM request option 2:

Table 53. VSAM Request Option

Code (Hex)	RPLOPT2	Description
X'80'	RPLKEY	Keyed access.
X'40'	RPLADR	Addressed access.
	RPLADD	Addressed access.
X'20'	RPLCNV	CINV access (by RBA).
X'10'	RPLBWD	FWD=0/BWD=1
X'08'	RPLLRD	ARD=0/LRD=1
X'04'	RPLWAITX	SYN processing wait exit.
X'02'	RPLUPD	Update.
X'01'	RPLNSP	Note string position.

### X'AC' Trace Entry

Figure 104 shows the X'AC' trace entry.

**TRACE ID = X'AC'**

word 0 – byte 1 - X'AC' - Database call analyzer entry (only present in a DBCTL environment)  
 byte 2 - PST number  
 bytes 3-4 - Trace sequence number  
 word 1 – Eye -catcher RTKN  
 word 2 – Not used  
 word 3 – Not used  
 words 4-7 – This 16-byte CCTL recovery token is used to correlate DL/I activity on other subsystems

Figure 104. X'AC' Trace Entry

### X'C4' Trace Entry

Figure 105 on page 276 shows the X'C4' trace entry.

**TRACE ID = X'C4'**

word 0 – byte 1 - X'C4' - DELETE/REPLACE used to provide diagnosis information for error conditions. This entry is written when an error is detected.  
 byte 2 - PST number  
 bytes 3-4 - Trace sequence number

word 1 – byte 1 - ID invoking subroutine (see note 2 below; see note 3 below)  
 byte 2 - ID of originating subroutine (see note 3 below)  
 byte 3 - Subcode (set by originating subroutine - see note 3 below)  
 byte 4 - Internal code for status code or pseudoabend (see note 3 below)

word 2 – SDB for replace operation. DLTWS for delete operation.  
 Register value 7.

word 3 – Level table for replace operation. DLTWA address for delete operation.  
 Register value 8.

word 4 – Usually the PSDB address for segment. Register value 6.

word 5 – byte 1 - DELETE/REPLACE return code  
 bytes 2-4 - Return offset from caller's CSECT

word 6 – PSTDSGA - DSG address

word 7 – Information local to the subroutine that might be useful in problem resolution

Figure 105. X'C4' Trace Entry

**Notes to Figure 105:**

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.
3. These 4 bytes, in word 2, in a DELETE/RELEASE error are documented in the *IMS Version 9: Failure Analysis Structure Tables (FAST) for Dump Analysis* for the various abends. This is ENTRY1 field referred to in the DELETE/REPLACE module.

**X'C6' Trace Entry**

Figure 106 shows the X'C6' trace entry.

**TRACE ID = X'C6'**

word 0 – byte 1 - X'C6' - Special promote lock trace entry  
 byte 2 - PST number (see note 1 below)  
 bytes 3-4 - Trace sequence number

word 1 – byte 1-2 - Not used  
 - byte 3 - Special lock/unlock call (see note 1)  
 - byte 4 - Level of this lock

words 2-3 - C'PROMOTE '

word 4 - REQ address

word 5 - QCB address

words 6-7 - Resource id (see note 2)

Figure 106. X'C6' Trace Entry

**Notes to Figure 106:**

1. See Table 54 for special lock or unlock call:

Table 54. Special Lock or Unlock Call

Code (Hex)	Special Lock or Unlock Call	Description
X'08'	PROENQ	Lock call.
X'10'	PRODEQ	Unlock call.

2. Resource id is an 8-byte field:

- bytes 1-4 – Complement of original RBA
- bytes 5-6 – DMB number
- byte 7 – DCB number
- byte 8 – C'Z' id suffix

### X'C7' Trace Entry Not Using IRLM

Figure 107 shows the X'C7' trace entry when not using the IRLM.

**TRACE ID = X'C7'**

- word 0 – byte 1 - X'C7' - Exclusive control deadlock detection trace entry (Written only when a conflict causes an abend).
- byte 2 - PST number (see note 1 below)
- bytes 3-4 - Trace sequence number
- word 1 – byte 1 - PST number (see note 1 below)
- bytes 2-4 - Address of PST to be backed out (gets ABENDU0777 - see note 3 below)
- words 2-3 (see note 2 below) - byte 1 - PST number
- bytes 2-4 - Conflicting PST address
- words 4-5 (see note 4 below) - PSB name
- words 6-7 (see note 4 below) - DMB name

Figure 107. X'C7' Trace Entry (When Not Using the IRLM)

#### Notes to Figure 107:

- The entry for the PST number that got the U0777.
- The addresses of the two conflicting PSTs.
- The address of the PST that got the U0777.
- The PSB and DMB name of the cause for the contention.

### X'C7' Trace Entry Using IRLM

Figure 108 shows the X'C7' trace entry when using the IRLM.

**TRACE ID = X'C7'**

- word 0 – byte 1 - X'C7'
- byte 2 - 00
- bytes 3-4 - Trace sequence number
- word 1 – Not used
- words 2-5 (see note 1 below) - byte 1 - PST number
- bytes 2-4 - PST address
- words 6-7 (see note 2 below) - Resource ID

Figure 108. X'C7' Trace Entry (When Using the IRLM)

#### Notes to Figure 108:

- PST number and address of PSTs in deadlock net. If number of PSTs in deadlock net is greater than 4, only 4 are shown.
- Resource ID that is the cause of the deadlock.

## X'C8' Trace Entry

Figure 109 shows the X'C8' trace entry.

**TRACE ID = X'C8'**

word 0 – byte 1 - X'C8' - Lock request manager entry (DFSLMGR0)  
 byte 2 - PST number  
 bytes 3-4 - Trace sequence number  
 word 1 – byte 1 - Function - See macro DFLMD for mapping of  
 each byte in this word  
 byte 2 - State (see note below)  
 byte 3 - Class - the class is the relative PST number  
 byte 4 - Flags  
 word 2 - byte 1 - Return code from IRLM  
 bytes 2-4 - Can be PST, CLB, or SRB address  
 word 3 - Can be resource name address, token, or altered  
 buffer mask  
 word 4 - bytes 1-2 - 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 Messages and Codes*,  
*Volumes 1 and 2*.)  
 bytes 3-4 - This is a feedback area from the RLPL and is used  
 primarily by the IBM Support Center, if needed.  
 words 5-7 - This is a feedback area from the RLPL and is used primarily  
 by the IBM Support Center, if needed.

Figure 109. X'C8' Trace Entry

### Note to Figure 109:

The possible state settings and their meaning:

**X'00'** Unconditional release  
**X'02'** Read  
**X'04'** Share  
**X'06'** Update  
**X'08'** Exclusive

## X'C9' Trace Entry

Figure 110 on page 279 shows the X'C9' trace entry.



**TRACE ID = X'C9'**

word 0 – byte 1 - X'C9' - Lock request manager entry (DFSLMGR0) exit  
byte 2 - PST number  
bytes 3-4 - Trace sequence number

word 1 – byte 1 - Function - See macro DFLMD for mapping of  
each byte in this word.  
byte 2 - State (see note below )  
byte 3 - Class - the class is the relative PST number  
byte 4 - Flags

word 2 - byte 1 - Return code from IRLM  
bytes 2-4 - Can be PST, CLB, or SRB address

word 3 - Can be resource name address, token, or altered  
buffer mask

word 4 - bytes 1-2 - 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 Messages and Codes*,  
*Volumes 1 and 2*.)  
bytes 3-4 - This is a feedback area from the RLPL and is used  
primarily by the IBM Support Center, if needed.

words 5-7 - This is a feedback area from the RLPL and is used primarily  
by the IBM Support Center, if needed.

*Figure 110. X'C9' Trace Entry*

**Note to Figure 110:**

The possible state settings and their meaning:

**X'00'** Unconditional release

**X'02'** Read

**X'04'** Share

**X'06'** Update

**X'08'** Exclusive

**X'CA' Trace Entry**

Figure 111 on page 280 shows the X'CA' trace entry.

**TRACE ID = X'CA'**

word 0 – byte 1 - X'CA' - Exclusive control ENQ/DEQ (PI - Program Isolation) trace entry  
 byte 2 - PST number (see note 1 below)  
 bytes 3-4 - Trace sequence number

word 1 – byte 1 - Record type (see note 8 below)  
 byte 2 - Class for Q command operation  
 byte 3 - Requested function (Use PRM DSECT (PRMFUNCTN) - see note 2 below)  
 byte 4 - PRMLEVEL - Level of control requested  
 (1 =Read only, 2=Share, 3=Update, 4=Exclusive - see note 3 below)

word 2 - bytes 1-2 - Wait count (how many times this task had to wait - see note 7 below)  
 bytes 2-4 - Waited for count (number of tasks waiting for this resource)

word 3 - PITIME relative to 00:00:00 on PIDATE (SCDPITIME)

word 4 - bytes 1-2 - Feedback from DFSFXC10 (Use PRM DSECT, PRMFBK field. See note 5 below)  
 byte 3 - Return code from DFSFXC10 (see note 6 below)  
 byte 4 - PSFUNCT (function codes DSECT)

word 5 - Token from DFSFXC10 (pointer to control block enqueued resource)

word 6 - RBA or RBN (see note 4 below)

word 7 - bytes 1-2 - DMB number  
 byte 3 - DCB number  
 byte 4 - Not used

*Figure 111. X'CA' Trace Entry***Notes to Figure 111:**

- 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.
- The requested PI function.
- The level at which the resource was requested.
- The RBA or RBN of the resource requested by PI (relates to X'04' in the X'CC' trace entry).
- The 2 bytes of feedback from DFSFXC10 (X'0C' and X'0D' in PRM DSECT).
- The return code.  
 DFSFXC10 RETURN CODES:  
 0 - Successful  
 4 - Wait required - usually has CB trace related to it  
 8 - Pseudoabend, either lost deadlock (U0777) or out of ENQ/DEQ space (U0775)  
 C - Invalid call
- 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.)
- This shows the type of X'CA' record this is. (X'CA-08' trace entry follows.)  
**X'00'** Standard trace PI record  
**X'01'** Timing ACT/ENQ wait - may have CB trace entry associated with it  
**X'04'** Lock MGR trace record  
**X'08'** DL/I call record - see X'CA' - X'08' trace entry

## **X'CA' through X'08' Trace Entry**

Figure 112 shows the X'CA' through X'08' trace entry.

```
TRACE ID   = X'CA'-X'08'
```

word 0 – byte 1 - X'CA'-X'08' - PI-DL/I call trace entry  
          byte 2 - PST number  
          bytes 3-4 - Trace sequence number

word 1 – byte 1 - X'08' = DL/I call record  
          bytes 2-4 - Not used

word 2 – bytes 1-2 - Wait count (how many times this task  
                    had to wait)  
          bytes 2-4 - Waited for count (number of tasks waiting  
                    for this resource)

word 3 – PI time

word 4 – PST account field for function (count of  
          the time of calls)

word 5 – DL/I call (GNP, ISRT, etc.)

words 6-7 - Not used

*Figure 112. X'CA'—X'08' Trace Entry*

## **X'C4' Trace Entry**

Figure 113 shows the X'CA' trace entry for Fast Path calls.

```
TRACE ID   = X'CA'
```

word 0 – byte 1 - X'CA' - Exclusive control ENQ/DEQ (PI -  
                    Program Isolation) trace entry  
          byte 2 - PST number (1)  
          bytes 3-4 - Trace sequence number

word 1 – IRC1, indicating a Fast Path call

word 2 – Call Function (GU, GN, and so on)

word 3 – PROCOPT

word 4 – PI time (also in reg 5 in Fast Path trace,  
          if active)

word 5 – A(PBC)

word 6 – A(EPCB)

word 7 – Not used

*Figure 113. X'CA' Trace Entry for Fast Path Calls*

## **X'CB' Trace Entry**

Figure 114 on page 282 shows the X'CB' trace entry.

**TRACE ID = X'CB'**

word 0 – byte 1 - X'CB' - PI - (Program Isolation) trace lock elapsed time  
 byte 2 - PST number  
 bytes 3-4 - Trace sequence number  
 words 1-2 – DMB Name for which the wait was performed  
 word 3 - Same as PITIME IN X'CA' record  
 word 4 - byte 1 - First byte of feedback from enqueue request  
 byte 2 - PST owning resource at the time of wait  
 bytes 3-4 - Trace sequence number on X'CA' record  
 word 5 - Elapsed time for enqueue wait  
 word 6 bytes 1-4 - word 8 bytes 1-3 - 7 bytes of resource ID  
 word 7 - byte 4 - Post code

*Figure 114. X'CB' Trace Entry***X'CC' Trace Entry**

Figure 115 shows the X'CC' trace entry.

**TRACE ID = X'CC'**

word 0 – byte 1 - X'CC' - Lock request handler (DFSLRH00) entry  
 byte 2 - PST number (see note 1 below)  
 bytes 3-4 - Trace sequence number  
 word 1 - Block number on RBA (see note 2 below)  
 word 2 - PSTTOKEN - The object of the request  
 word 3 - PSTLRPRM - These bytes are described in the PSTLRPRM chart below. The first byte equates to byte 0, the second to byte 1, and so on (see note 3 below).  
 word 4 - bytes 1-2 - Subcode from lock manager (IRLM) or PRMFBK feedback for DFSFXC10. (For a description of IRLM codes, see *IMS Messages and Codes, Volumes 1 and 2.*)  
 byte 3 - Register 15 return code  
 byte 4 - Return code from lock manager or DFSFXC10 (Use DFSFXC10 return codes from the X'CA' trace entry, note 6) (See note 5 below)  
 word 5 - byte 1 - PSTLRSUB-DFSLRH00 abend subcode (see note 7 below)  
 bytes 2-4 - PSTABTRM - System abend code (see note 6 below)  
 word 6 - PSTDSGA - Address of the DSG used by this PST  
 word 7 - byte 1 - Return register  
 bytes 2-4 - Address within module where DFSLRH00 was called

*Figure 115. X'CC' Trace Entry***Notes to Figure 115:**

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 9: 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 9: 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 55 shows the PSTLRPRM chart (bytes 0 through 3).

*Table 55. 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
75	Test feedback for local and global locks
80	LRHGIRD 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

Table 55. PSTLRPRM Chart (Bytes 0 through 3) (continued)

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
<b>Byte 1(Hex) Meaning</b>	
80	MODE=COND
40	MODE=UNCOND
10	Owning WU given on RRIDX
00	Mode not applicable
<b>Byte 2(Hex) Meaning</b>	
01	STATE=READ
02	STATE=SHARE
03	STATE=UPDATE
04	STATE=EXCL
F0	STATE PRESET (Fast Path)
00	STATE not applicable
<b>Byte 3(Hex) Meaning</b>	
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

### | X'CF' Trace Entry

| Figure 116 on page 285 shows the X'CF' trace entry.

|

**TRACE ID = X'CF'**

```

word 0 - byte 1 - x'CF' - I/O toleration (DFSTOPR0)
                                     trace entry
        byte 2 - PST number
        bytes 3-4 - Trace sequence number
word 1 - byte 1 - I/O toleration return code
        - byte 2 - TORFUNC - I/O toleration function
          code (see note 1)
        - byte 3 - TORFLG1 - I/O toleration flag 1 (see
          note 2)
        - byte 4 - TORFLG2 - I/O toleration flag 2 (see
          note 3)
words 2-3 - EEQEFLCS - EEQE flags
word 4 - DDIR or DMAC address
word 5 - RBA or RBN
word 6 - bytes 1-2 - DMB number
        - byte 3 - DCB number
        - byte 4 - TORWORK+2 when DBRC change of EEQE
word 7 - EEQE address
    
```

Figure 116. X'CF' Trace Entry

**Note:**

1. See Table 56 for I/O toleration function code:

Table 56. I/O Toleration Function Code

Code (Hex)	TORFUNC	Description
X'01'	TORCEQE	Create EEQE.
X'02'	TORDEQE	Delete EEQE.
X'04'	TORFEQE	Find I/O toleration EEQE.
X'08'	TORMEQE	Copy/Move to I/O toleration buffer.
X'10'	TORNEQE	Send notifies on I/O toleration EEQE's.
X'20'	TORPURG	Close I/O toleration mode.
X'40'	TORDUI	Process DBRC DUI call EEQE list.
X'80'	TORDBCL	DB close I/O error write retry.
X'C0'	TORCHKPT	Do system checkpoint logging.

2. See Table 57 for I/O toleration flag 1:

Table 57. I/O Toleration Flag 1

Code (Hex)	TORFLG1	Description
X'80'	TOR1FP	<ul style="list-style-type: none"> <li>• If on, Fast Path.</li> <li>• If off, DL/I.</li> </ul>
X'40'	TOR1NOT	Creator is notify.
X'20'	TOR1PST	<ul style="list-style-type: none"> <li>• If on, then R0 has PST address.</li> <li>• If off, then R0 has SCD address.</li> </ul>
X'10'	TOR1RST	Caller is restart log read.
X'01'	TOR1FPIR	DBFMIOS0: FP IDT resolution.
X'90'	TOR1FPRS	Caller is FP restart log read.

3. See Table 58 for I/O toleration flag 2:

Table 58. I/O Toleration Flag 2

Code (Hex)	TORFLG2	Description
X'80'	TOR2IOT	Creator is I/O toleration.
X'40'	TOR2RD	Creator is read error.
X'20'	TOR2WRT	Creator is write error.
X'10'	TOR2USER	Creator is DBRC command.
X'08'	TOR2PERM	Creator is permanent.
X'04'	TOR2IDT	Creator is indoubt process.
X'01'	TOR2NDX	EEQEFLG2:EEQENDX KSDS INDEX CI

### X'D0' Trace Entry

Figure 117 shows the X'D0' trace entry.

**TRACE ID = X'D0'**

- word 0 – byte 1 - x'D0' - IRLM notify sent trace entry
  - byte 2 - PST number
  - bytes 3-4 - Trace sequence number
- word 1 - byte 1 - Sub-route code
  - bytes 2-3 - DMB number or PID for partition
  - byte 4 - DCB number

Format for buffer invalidation or write error notify:

- word 2 - RBN/RBA OF BUFFER
- word 3 - Not used
- word 4 - Not used
- word 5 - Not used
- word 6 - bytes 1-3 - Not used
  - byte 4 - NCBFLAG
- word 7 - Not used

Format for OSAM data set extend:

- word 2 - DCBHIBLK
- word 3 - DCBRLBLK
- word 4 - DCBRBASN
- word 5 - bytes 1-4 - Volume serial number
- word 6 - bytes 1-2 - Volume serial number
  - byte 3 - Not used
  - byte 4 - NCBFLAG
- word 7 - Not used

Format for VSAM data set extend:

- word 2 - VSILVL – Current VSI level number
- word 3 - VSIHRBA – Current high used RBA
- word 4 - VSIERBA – Current high allocated RBA
- word 5 - VSILVL – Extent VSI level number
- word 6 - VSIHRBA – Extent high used RBA
- word 7 - VSIERBA – Extent high allocated RBA

Figure 117. X'D0' Trace Entry

### X'D1' Trace Entry

Figure 118 on page 287 shows the X'D1' trace entry.



**TRACE ID = X'D1'**

- word 0 – byte 1 - x'D1' - IRLM notify received trace entry
  - byte 2 - Not used, no PST number
  - bytes 3-4 - Trace sequence number
- word 1 - byte 1 - Sub-route code
  - bytes 2-3 - DMB number or PID for partition
  - byte 4 - DCB number

**Format for buffer invalidation or write error notify:**

- word 2 - RBN/RBA of buffer
- word 3 - Buffer prefix address
- word 4 - byte 1 - SB Global serialization trace field (see note below)
  - byte 2 - Not used
  - bytes 3-4 - Number of invalidated buffers
- word 5 - Last invalidated buffer address
- word 6 - bytes 1-3 - Not used
  - byte 4 - NCBFLAG
- word 7 - Subsystem id

**Format for OSAM data set extend:**

- word 2 - DCBHIBLK
- word 3 - DCBRLBLK
- word 4 - DCBRBASN
- word 5 - bytes 1-4 - Volume serial number
- word 6 - bytes 1-2 - Volume serial number
  - bytes 3 - Not used
  - byte 4 - NCBFLAG
- word 7 - Subsystem id

**Format for VSAM data set extend:**

- word 2 - VSILVL – Current VSI level number
- word 3 - VSIHRBA – Current high used RBA
- word 4 - VSILVL – Extent VSI level number
- word 5 - VSIHRBA – Extent high used RBA
- word 6 - AMP address
- word 7 - Subsystem id

*Figure 118. X'D1' Trace Entry*

**Note:**

- | **X'80'** Global serialization entered (SBH search started)
- | **X'40'** Waiting PST was posted.

**X'D9' Trace Entry**

| Most X'D9' trace entries have the following information in the first three words, except for the OLR Command processing (see "X'D9' Trace Entry: OLR Command Processing" on page 293). Figure 119 on page 288 shows words 0-2 of the X'D9' trace entry:

**TRACE ID = X'D9'**

word 0 – byte 1 - x'D9' - Online Reorganization (OLR) trace entry.  
 byte 2 - PST number.  
 bytes 3-4 - Trace sequence number.  
 word 1 – byte 1 - Module ID.  
 byte 2 - Module subcode.  
 bytes 3-4 - Local DMB number.  
 word 2 - bytes 1-2 - Global DMB number.  
 bytes 3-4 - Partition ID.

Figure 119. X'D9' Trace Entry - Words 0 through 2

**X'D9' Trace Entry: OLR Output Data Set Validation or Creation and Inactive Data Set Deletion:**

Figure 120 shows words that are specific to the OLR output data set validation or creation and inactive data set deletion:

**TRACE ID = X'D9'**

word 3 - bytes 1-2 - Error message number as four packed decimal digits or as binary 0 if there is no error.  
 byte 3 - Reserved, 0.  
 byte 4 - DCB number for the data set involved.  
 The x'80' bit is on if the data set is one of the M through V and Y data sets (see notes 1 and 2 below).  
 word 4 - bytes 1-4 - DDIR address.

Figure 120. X'D9' Trace Entry - Words Specific to the Validation or Creation of the OLR Output Data Set and the Deletion of the Inactive Data Set

**Notes to Figure 120:**

1. When no error has occurred, the error message number in word 3 has a value of binary zero, and there is no further information in the trace entry beyond word 4.
2. For the following error message numbers, there is information that is specific to the particular error:

**2990 - Unexpected error from system macro instruction:**

words 5-6 - Macro name.  
 word 7 - bytes 1-2 - Return code.  
 bytes 3-4 - Reason Code.

**2991 - Output data set validation error:**

word 5 - Reason code from DFS2991I message text.

**2992 - Unexpected error from CSI or catalog management, form 1:**

word 5 - Reason area from CSI or catalog management  
 word 6 - byte 1 - Reason area type:  
 'C' catalog error  
 'D' data set error  
 'I' CSI call

**2992 - Unexpected return code from CSI, form 2:**

word 5 - Return code from CSI call.  
 word 6 - byte 1 - X'00'  
 bytes 2-4 - Reason code from CSI call.

**2993 - Unexpected device class:**

word 5 - byte 1 - UCB device class.

**2994 - Unexpected IDCAMS return code creating a data set:**

word 5 - Return code from IDCAMS.

**2995 - Unexpected IDCAMS return code deleting a data set:**

word 5 - Return code from IDCAMS.

**2996 - Insufficient DASD space to create a data set:**

word 5 - bytes 1-2 - SVC 99 error reason  
 code.  
 bytes 3-4 - Reserved.  
 word 6 - SMS error reason code.  
 word 7 - Number of blocks wanted.

**2998 - Miscellaneous SVC 99 errors creating a data set:**

word 5 - SVC 99 error reason  
 code.  
 bytes 3-4 - Reserved.  
 word 6 - SMS error reason code.  
 word 7 - Number of blocks wanted.

Table 59 shows the module ID and module subcode values for the X'D9' trace entries that represent the OLR output data set validation or creation process and the inactive data set deletion process:

*Table 59. Module and Subcode ID for X'D9': Validation or Creation of the OLR Output Data Set and the Deletion of the Inactive Data Set*

Module ID	Module	Subcode	Meaning
A	DFSORA00	X'10'	Data set creation successful
A	DFSORA00	X'11'	Data set creation successful
A	DFSORA00	X'12'	Data set creation successful
A	DFSORA00	X'13'	Data set creation successful
A	DFSORA00	X'14'	Data set validation successful
A	DFSORA00	X'15'	Data set validation successful
A	DFSORA00	X'16'	Data set validation successful
A	DFSORA00	X'20'	Primary index was not a VSAM KSDS
A	DFSORA00	X'21'	VSAM data set did not have REUSE attribute
A	DFSORA00	X'22'	VSAM record length did not match input
A	DFSORA00	X'23'	VSAM control interval size did not match input
A	DFSORA00	X'24'	KSDS key offset or length size did not match input

Table 59. Module and Subcode ID for X'D9': Validation or Creation of the OLR Output Data Set and the Deletion of the Inactive Data Set (continued)

Module ID	Module	Subcode	Meaning
A	DFSORA00	X'25'	Miscellaneous errors; another trace entry precedes this one
A	DFSORA00	X'C1'	Internal error: invalid DFSORA00 call
A	DFSORA00	X'C2'	Internal error: No data set in X'2930' log record
A	DFSORA00	X'C3'	Invalid input data set
A	DFSORA00	X'C4'	Multi-volume input, but no output data set
A	DFSORA00	X'C5'	Non-DASD data set
A	DFSORA00	X'C6'	Multi-volume data set to be recovered
A	DFSORA00	X'C7'	Non-DASD data set
A	DFSORA00	X'C8'	Data set not usable for OSAM
A	DFSORA00	X'C9'	Data set is a PDS or PDSE
A	DFSORA00	X'D1'	Data set is not VSAM
A	DFSORA00	X'D2'	Data set is not a VSAM KSDS
A	DFSORA00	X'D3'	VSAM data set did not have REUSE attribute
A	DFSORA00	X'D4'	VSAM record length did not match input
A	DFSORA00	X'D5'	VSAM control interval size did not match input
A	DFSORA00	X'D6'	KSDS key offset or length size did not match input
A	DFSORA00	X'D7'	Data set not usable for OSAM
A	DFSORA00	X'D8'	Data set is a PDS or PDSE
A	DFSORA00	X'D9'	Data set is not VSAM
A	DFSORA00	X'E2'	Data set is not a VSAM KSDS
A	DFSORA00	X'E3'	VSAM data set did not have REUSE attribute
A	DFSORA00	X'E4'	VSAM record length did not match input
A	DFSORA00	X'E5'	VSAM control interval size did not match input
A	DFSORA00	X'E6'	KSDS key offset or length size did not match input
A	DFSORA00	X'E7'	Data set not usable for OSAM
A	DFSORA00	X'E8'	Data set is a PDS or PDSE
A	DFSORA00	X'E9'	Data set is not VSAM
B	DFSORA10	X'C1'	Data set error reported by CSI
B	DFSORA10	X'C2'	No error information available from CSI
B	DFSORA10	X'C3'	Catalog error reported by CSI
B	DFSORA10	X'C4'	Unexpected return code 4 from CSI
B	DFSORA10	X'C5'	Unexpected return code 4 from CSI
B	DFSORA10	X'C6'	Unexpected return code from CSI

Table 59. Module and Subcode ID for X'D9': Validation or Creation of the OLR Output Data Set and the Deletion of the Inactive Data Set (continued)

Module ID	Module	Subcode	Meaning
B	DFSORA10	X'C7'	Unexpected return code from DEVTYPE
B	DFSORA10	X'C8'	Data set not on volume
B	DFSORA10	X'C9'	Unexpected return code from OBTAIN
B	DFSORA10	X'D1'	Unexpected return code from OBTAIN
B	DFSORA10	X'D2'	Unexpected return code from TRKCALC
B	DFSORA10	X'D3'	Unexpected return code 12 from GETDSAB
B	DFSORA10	X'D4'	Unexpected return code from GETDSAB
B	DFSORA10	X'D5'	Unexpected return code from SWAREQ
B	DFSORA10	X'D6'	Invalid data set name
D	DFSORA20	X'C1'	SVC 99 information reason returned
D	DFSORA20	X'C2'	Insufficient space on volume
D	DFSORA20	X'C3'	Data set in use
D	DFSORA20	X'C4'	Insufficient space, SMS
D	DFSORA20	X'C5'	SVC 99 error and SMS reason returned
D	DFSORA20	X'C6'	SVC 99 error code returned
D	DFSORA20	X'C7'	SVC 99 error code returned
D	DFSORA20	X'C8'	Unexpected return code from SVC 99
D	DFSORA20	X'C9'	SVC 99 information reason returned
D	DFSORA20	X'D1'	SVC 99 error code
D	DFSORA20	X'D2'	Unexpected return code from SVC 99
D	DFSORA20	X'D3'	Unexpected return code from IDCAMS
E	DFSORA30	X'C1'	Unexpected return code from IDCAMS
E	DFSORA40	X'C1'	GETMAIN failure

**X'D9' Trace Entry: Fence Value Before an OLR IPOST/IWAIT:** Figure 121 shows the remaining words of the X'D9' trace entries that are specific to the fence value before an OLR IPOST/IWAIT:

TRACE ID = X'D9'

- word 3 - Can contain the address of the PST to be posted.
- words 4-5 - Contains DMBORFEN.
- words 6-7 - Contains DMBAMFEN.

Figure 121. X'D9' Trace Entry - Words Specific to the Fence Value Before an OLR IPOST/IWAIT

Table 60 shows the module ID and module subcode values for the X'D9' trace entries that represent the fence value before an OLR IPOST/IWAIT.

Table 60. Module and Subcode ID for X'D9': Fence Value Before an OLR IPOST/IWAIT

Module ID	Module	Subcode	Meaning
J	DFSORP70	X'01'	IPOST for the OLR I/O fence

Table 60. Module and Subcode ID for X'D9': Fence Value Before an OLR IPOST/WAIT (continued)

Module ID	Module	Subcode	Meaning
J	DFSORP70	X'02'	IWAIT for the OLR action module fence
J	DFSORP70	X'03'	IPOST for the OLR I/O fence
J	DFSORP70	X'04'	IWAIT for the OLR action module fence
L	DFSORP40	X'01'	IWAIT for the OLR action module fence
L	DFSORP40	X'02'	IPOST for the OLR action module fence
M	DFSPCSH0	X'01'	IWAIT for the OLR action module fence
M	DFSPCSH0	X'02'	IPOST for the OLR action module fence
M	DFSPCSH0	X'03'	IWAIT for the OLR action module fence
M	DFSPCSH0	X'04'	IPOST for the OLR action module fence
O	DFSDLOC0	X'01'	IPOST for the OLR I/O fence
O	DFSDLOC0	X'02'	IWAIT for the OLR I/O fence
R	DFSDLR00	X'01'	IPOST for the OLR action module fence
R	DFSDLR00	X'02'	IPOST for the OLR action module fence
R	DFSDLR00	X'03'	IWAIT for the OLR action module fence
R	DFSDLR00	X'04'	IPOST for the OLR action module fence
R	DFSDLR00	X'05'	IPOST for the OLR action module fence
R	DFSDLR00	X'06'	IWAIT for the OLR action module fence
R	DFSDLR00	X'07'	IWAIT for the OLR action module fence
R	DFSDLR00	X'08'	IWAIT for the OLR action module fence
S	DFSDVBH0	X'01'	IWAIT for the OLR I/O fence
S	DFSDVBH0	X'02'	IPOST for the OLR I/O fence
V	DFSDVSM0	X'01'	IPOST for the OLR I/O fence
V	DFSDVSM0	X'02'	IPOST for the OLR I/O fence

**X'D9' Trace Entry : Next UOR Determination:** Figure 122 on page 293 shows the remaining words of the X'D9' trace entries that are specific to the next UOR determination.

TRACE ID = X'D9'

- word 3 - The total number of UORs performed.
- word 4 - The execution span for this UOR.
- word 5 - The proposed size for the next UOR.
- word 6 - The total bytes moved during this UOR.
- word 7 - The total locks held during this UOR.

Figure 122. X'9D' Trace Entry - Next UOR Determination

| **X'D9' Trace Entry: OLR Command Processing:** Figure 123 shows the X'9D' trace entry definitions used by the Online Reorganization (OLR) command processing:

TRACE ID = X'D9'

- word 0 - byte 1 - X'D9' Online Reorganization (OLR) trace entry.
- byte 2 - Zero - not used.
- bytes 3-4 - Trace sequence number.
- word 1 - byte 1 - Module ID.
- byte 2 - Module subcode.
- byte 3 - Module function.
- byte 4 - FREESTOR error return code.
- word 2 - Last 4 bytes of the IMS ID (SCDIMSNM+4) processing the command.
- words 3-4 - Command VERB (INIT, UPD, TERM, and QRY if an type-2 command)
- words 5-6 - Operation Manager name ('NONOMCMD' if OLR type-1 command).
- word 7 - Address of storage not freed if FREESTOR failure.

Figure 123. X'9D' Trace Entry - Words Specific to OLR Command Processing

| **Note:** For all X'D9' trace entries, the module ID, and usually the module subcode as well, indicate both the meaning of the trace entry and the format of the rest of the trace entry.

| Table 61 shows the module ID values in X'D9' trace entries that represent OLR command processing:

| Table 61. Module and Subcode ID for X'D9': OLR Command Processing

Module ID	Module	Subcode	Meaning
C	DFSORC00	X'00'	OLR type-2 command issued
C	DFSORC00	X'01'	FREESTOR error during INIT error processing
C	DFSORC00	X'02'	FREESTOR error after sending command response
P	DFSORC10	X'00'	OLR type-1 command issued
P	DFSORC10	X'01'	FREESTOR error during INIT processing cleanup

| **X'D9' Trace Entry: OLR Start:** Figure 124 on page 294 shows the remaining words of the X'D9' trace entries that are specific to the OLR start.

```
TRACE ID   = X'D9'

word 3 - The RBA of the cursor in the second CI
         or block.
word 4 - Unused.
word 5 - Unused.
word 6 - Unused.
word 7 - Unused.
```

Figure 124. X'9D' Trace Entry - Words Specific to OLR Start

| **X'D9' Trace Entry: Start of a UOR:** Figure Figure 125 shows the remaining words of the X'D9' trace  
| entries that are specific to the start of a UOR.  
|

```
TRACE ID   = X'D9'

word 3 - The first four bytes of the last committed
         cursor.
word 4 - The start time of this UOR.
word 5 - The execution span for this UOR.
word 6 - The time that was waited before this
         UOR started.
word 7 - Unused.
```

Figure 125. X'9D' Trace Entry - Words Specific to Start of UOR

| **X'D9' Trace Entry: UOR Wait for Timer:** Figure Figure 126 shows the X'D9' trace entries that are  
| specific to the UOR wait for timer.  
|

```
TRACE ID   = X'D9'

word 3 - Unused
word 4 - The start time of this UOR.
word 5 - The execution span for this UOR.
word 6 - The time that will be waited before
         the next UOR starts.
word 7 - Unused.
```

Figure 126. X'9D' Trace Entry - Words Specific to UOR Wait for Timer

| **X'DA' Trace Entry**  
| Figure 127 on page 295 shows the X'DA' trace entry.  
|



**TRACE ID = X'DA'**

word 0 – byte 1 - X'DA' - VSAM JRNAD or UPAD exit  
 byte 2 - PST number  
 bytes 3-4 - Trace sequence number  
 word 1 - Word 3 of JRNAD or UPAD parameter list  
 word 2 - Word 4 of JRNAD or UPAD parameter list  
 word 3 - Word 5 of JRNAD or UPAD parameter list  
 word 4 - byte 1 - JRNAD or UPAD code (For an explanation of these codes, see note 5 below)  
 word 4 bytes 2-4 - AMB address  
 word 5 bytes 1-3 - Register 14 from PLH stack  
 (see notes 1-4 below)  
 word 5 byte 4 and word 6 bytes 1-2 - Register 14 from PLH stack  
 (see notes 1-4 below)  
 word 6 bytes 3-4 and word 7 byte 1 - Register 14 from PLH stack  
 (see notes 1-4 below)  
 word 7 - bytes 2-4 - Register 14 from PLH stack

*Figure 127. X'DA' Trace Entry*

**Notes to Figure 127:**

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 are probably not in the dump.
4. An AMBLIST of VSAM module IDA019L1, with OUTPUT=BOTH specified, is needed to determine which CSECTS had control.
5. For an explanation of these codes, see Table 62.

*Table 62. 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
UPAD	04	Post ECB (XMEM only)

**X'DB' through X'FA' Trace Entry**

Figure 128 on page 296 shows the X'DB' through X'FA' trace entries.

**TRACE ID = X'DB' - X'FA'**

- word 0 - byte 1 - X'DB' through X'FA'PSTFNCTN - Buffer handler trace - See the table entitled "Buffer Handler Function Codes Chart" below. This is the function from X'DB' thru X'FA' for which the trace was written (see note 1 below).
- byte 2 - PST number (see note 2 below)
- bytes 3-4 - Trace sequence number
- word 1 - bytes 1-2 - PSTDMBNM - DMB number. This field indicates which DMB is being used. The DMB directory (DDIR) gives the first DMB.
- byte 3 - PSTDCBNM - DCB number
- byte 4 - PSTRTCDE - See the table entitled "Buffer Handler Function Codes Chart" below. Usually indicates an error if nonzero. If an error, PSTDATA may contain residual data from the last call (see note 3 below)
- word 2 - byte 1 - PSTTRMID - ID of the module calling the buffer handler (see note 4 below)
- byte 2 - PSTTRMSC - Subcode of the module calling the buffer handler (see note 4 below)
- byte 3 - PSTBHFLG - DL/I buffer handler flags
- byte 4 - PSTSUBCD - Buffer handler internal work byte
- word 3 - PSTDSGA - Address of the DSG
- word 4 - PSTDATA - Address in real storage of the requested data. May point to the last retrieved data address in a call (failed abend) (see note 5 below).
- word 5 - PSTBUFFA - Address of buffer header. OSAM uses IBFTPRF DSECT. VSAM uses IDABUFC DSECT.
- word 6 - PSTISAMW - Work area
- word 7 - PSTBYTNM - Relative byte number of data or block number (see note 6 below).

Figure 128. X'DB' through X'FA' Trace Entries

**Notes to Figure 128:**

- 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 65 on page 299 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 63 shows the buffer handler function codes.

Table 63. 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	PSTBKLCCT	Block Locate

Table 63. Buffer Handler Function Codes Chart (continued)

Code (Hex)	PSTFNCTN	Caller's Request Function
E2	PSTBYLCT	Byte Locate
E3	PSTISRCH	Not used
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)
F5	PSTBXFER	RSR DTT BQEL transfer
F6	PSTBPURG	RSR DTT Purge/Release BQEL
F7	PSTRSIAB	Reset invalidate all buffers trigger
F9	PSTCPYGU	Position by key for Image Copy
FA	PSTCPYGN	Get next record for Image Copy
<b>Space Management Function Codes</b>		
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.
<b>Open/Close Function Codes</b>		
00	PSTOCCLS	This is a close call. This is the absence of PSTOCOPN (X'08') or PSTOCOPN is reset.
01	PSTOCDMB	The DDIR address is in register 2
02	PSTOCPCB	The PCB address is loaded from PSTDBPCB to registers 1
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
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	The PSTOCBAD (X'80') is set to indicate to the caller that the requested function failed

---



---



---

**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 64 is a chart of the buffer handler return codes.

*Table 64. Buffer Handler Return Codes Chart*

Return Code		Definition
PSTCLOK	X'00'	Everything correct
PSTGTDS	X'04'	RBN beyond data set
PSTRDERR	X'08'	Permanent read error
PSTNOSPC	X'0C'	No more space in data set
PSTBDCAL	X'10'	Illegal call
PSTENDDA	X'14'	End of data set encountered — no record returned
PSTNDTFD	X'18'	Requested record cannot be found
PSTNWBLK	X'1C'	New block created in buffer pool
PSTNPLSP	X'20'	Insufficient space in pool.
PSTTRMNT	X'24'	User must terminate, no space in pool.
PSTDUPLR	X'28'	Logical record already in KSDS.
PSTWRERR	X'2C'	Permanent write error.
PSTBUFIN	X'30'	Buffer invalidate.
PSTBIDIN	X'34'	Unable to acquire BID lock.
PSTPDERR	X'38'	Unable to locate DDIR/PDIR entry.
PSTNOSTO	X'3C'	Storage not available.
PSTRRERR	X'40'	CF read and register error.
PSTCURER	X'44'	Space management OLR cursor error.
PSTCLSDS	X'48'	Attempt to access a closed data set.

---

## Space Management and Buffer Handler Module Trace IDs

In space management and DL/I buffer handler trace entries, a 1-byte module ID identifies the calling module. A 1-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 65 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 65. 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
TIDORA00	A	DFSORA00	OLR data set creation/deletion
TIDZDC00	A	DFSZDC00	GSAM Controller
TIDORA10	B	DFSORA10	OLR data set information
TIDZDI00	B	DFSZDI00	GSAM Initialization
TIDORC00	C	DFSORC00	OLR OM command processor
TIDZDI20	C	DFSZDI20	GSAM Initialize GB
TIDDLDC0	D	DFSDLDC0	DELETE/REPLACE
TIDORA20	D	DFSORA20	Create data sets for OLR
TIDZDI30	D	DFSZDI30	GSAM Buffering Initialization
TIDFLST0	E	DFSFLST0	Batch STAE exit
TIDORA30	E	DFSORA30	Delete data sets for OLR
TIDZD110	E	DFSZD110	GSAM BSAM OPEN / CLOSE
TIDLRH00	F	DFSLRH00	LOCK request handler
TIDZD150	F	DFSZD150	GSAM VSAM OPEN / CLOSE
TIDORA40	G	DFSORA40	Performs OLR IDCAMS
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
TIDORP60	I	DFSORP60	OLR interfaces to DBRC
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
TIDORP40	L	DFSORP40	OLR termination and cleanup
TIDZSR10	L	DFSZSR10	GSAM Restart positioned
TIDPCSH0	M	DFSPCSH0	Partitioning Common Services Handler
TIDORP20	N	DFSORP20	OLR cursor and commit manager
TIDDL0C0	O	DFSDLOC0	OPEN/CLOSE
TIDDL0V0	O	DFSDLOV0	LOGICAL/VIRTUAL OPEN
TIDDCAP0	P	DFSDCAP0	Full-Function Data capture
TIDORC10	P	DFSORC10	OLR type-1 command processor
TIDDDUI0	Q	DFSDDUI0	DUI processor
TIDDLR00	R	DFSDLR00	RETRIEVE function
TIDDHD00	S	DFSDHD00	Space Manager (INIT procedure)
TIDDVBH0	S	DFSDVBH0	Buffer handler router
TIDFRSP0	S	DFSFRSP0	Space Manager (free space)
TIDGGSP0	S	DFSGGSP0	Space Manager (GET space)
TIDMMUD0	S	DFSMUD0	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

Table 65. Space Management and Buffer Handler Module Trace IDs (continued)

ID Label	Module ID	Calling Module	Module Function
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
TIDDVSM0	V	DFSDVSM0	VSAM interface
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 129 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
* DLI 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 34895982 B96E24B9 BCE6BA6E 50B9AE68 .....-.....B.>...W.>&...
PSTBYLCT        E201008C 00040100 D2014400 000A101C 0273720C 0272FA60 0274E45E 0000260C S.....K.....-...U;...
VSAM EXIT        DA01008D 0272FAB0 06000000 00004800 34895982 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 34895982 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 34895982 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 34895982 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 .....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 34895982 B96E24B9 BCE6BA6E 50B9AE68 .....-.....B.>...W.>&...
PSTBYLCT        E201009B 00040100 D2014400 000A101C 0273720C 0272FA60 0274E45E 0000260C S.....K.....-...U;...
VSAM EXIT        DA01009C 0272FAB0 06000000 00004800 34895982 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 34895982 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 34895982 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 34895982 B96E24B9 BCE6BA6E 50B9AE68 ...W...&.....B.>...W.>&...
PSTBYLCT        E20100A7 00030100 D2014400 000A205C 0273D354 0272FB50 0274E45E 00002754 S..X....K.....*...L.....&..U;...

```

Figure 129. 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 the following lines of code:

```
| DSECTS CSECT
|         DFSDLDC FUNC=DSECTS
|         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 1-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 (DFSDLW0)
```

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, IMS always turns the Retrieve trace on. The Retrieve trace is turned on automatically. (See *IMS Version 9: 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 9: Command Reference*.)

| **Note:** The Retrieve trace cannot be turned on if the DL/I trace is not active.

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 91 on page 263.) The byte at PSTRTNDX contains the offset to the next entry in the table. (See Figure 130 on page 305.)

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 66 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 130 on page 305.

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 66), which you found in the Retrieve trace table.

*Table 66. 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.



Table 66. The Subroutines of the Retrieve Module (DFSDLR00) (continued)

Hex ID	Subroutine Title	Subroutine Description
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.
1B	RDLPCONK	Locates logical parent using 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.

Table 66. The Subroutines of the Retrieve Module (DFSDLR00) (continued)

Hex ID	Subroutine Title	Subroutine Description
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.
I 37	HDTARGET	PHDAM/HDAM get a key equal or greater.
I 38	HDNEXT	PHDAM/HDAM get next.
I 39	HDTARGET	PHDAM/HDAM get a first.
I 3A	OLRTRACE	Trace IWAIT/IPOST for OLR fence.
F1	INIT	Initialization.



## Online Recovery Manager Trace

The Online Recovery Manager trace (ORTT) records the control flow that is related to /RECOVER command processing.

The following topics provide additional information:

- “Starting the Online Recovery Manager Trace”
- “Format of the Online Recovery Manager Trace”
- “Online Recovery Manager Trace Example” on page 310

## Starting the Online Recovery Manager Trace

The /TRACE SET ON TABLE ORTT 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 154.

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.

## Format of the Online Recovery Manager Trace

### Trace Entry: Online Recovery Service Request

#### A001

Table 67. Trace Record 3702 - Create Data Set Routine Invoke DYA

**Module:** DFSRWM00 - Database Recovery Manager Master ITASK

**Explanation:** Record cut when AWE request received by DFSRWM00

**Trace Subcode** RWM00 Request

Offset	Type	Length	Description
4	Fixed	2	Requested function(awrwfunc)
6	Bit	2	RWGB_Flags
	1... ..		rwgb_startup_complete
	.1.. ..		rwgb_startup_failure
	..1. ..		rwgb_dlisas
	...1 ..		rwgb_fp_allowed
	... 1..		*
	.... .1..		rwgb_terminating
	.... ..1.		rwgb_record_pipe_alloc
	.... ...1		rwgb_drm_init_complete
	1... ..		rwgb_ORS_installed
	.1.. ..		rwgb_DRF_installed
	..11 1111		*
8	Address	4	Request AWE address
12	Address	4	Next AWE address
16	Address	4	awrwcecb
20	Fixed	4	Awrwcecb->c_ecb

Table 67. Trace Record 3702 - Create Data Set Routine Invoke DYA (continued)

Offset	Type	Length	Description
24	Bit	4	Rwgb_init_flags
	1... ..		rwgb_init_load_1
	.1.. ..		rwgb_init_rwsp
	..1. ..		rwgb_init_load_2
	...1 ..		*
	.... 1..		rwgb_init_ascre
	.... .1..		rwgb_init_route
	.... ..1.		rwgb_init_write
	.... ...1		rwgb_init_read
	1... ..		rwgb_init_cmd
	.1.. ..		rwgb_init_rtb
	..1. ..		rwgb_init_fp
	...1 ..		rwgb_init_dli
	.... 1111		*
	1111 1111		*
	1111 1111		*

**Trace Entry: Online Recovery Service Request Processed**

**A002**

Table 68. Trace Record 3702 - Create Data Set Routine Invoke DYA

**Module:** DFSRWM00 - Database Recovery Manager Master ITASK

**Explanation:** Record cut when DFSRWM00 completes processing of request

**Trace Subcode** RWM00 Return

Offset	Type	Length	Description
4	Fixed	2	Requested function(awrwfunc)
6	Bit	2	RWGB_Flags
	1... ..		rwgb_startup_complete
	.1.. ..		rwgb_startup_failure
	..1. ..		rwgb_disas
	...1 ..		rwgb_fp_allowed
	.... 1..		*
	.... .1..		rwgb_terminating
	.... ..1.		rwgb_record_pipe_alloc
	.... ...1		rwgb_drm_init_complete
	1... ..		rwgb_ORS_installed
	.1.. ..		rwgb_DRF_installed
	..11 1111		*
8	Fixed	4	Request feedback (awrwfdbk)
12	Address	4	Rwgb_hold_queue
16	Address	4	awrwcecb
20	Fixed	4	Awrwcecb->c_ecb
24	Bit	4	Rwgb_init_flags
	1... ..		rwgb_init_load_1
	.1.. ..		rwgb_init_rwsp
	..1. ..		rwgb_init_load_2
	...1 ..		*
	.... 1..		rwgb_init_ascre
	.... .1..		rwgb_init_route
	.... ..1.		rwgb_init_write
	.... ...1		rwgb_init_read
	1... ..		rwgb_init_cmd
	.1.. ..		rwgb_init_rtb
	..1. ..		rwgb_init_fp
	...1 ..		rwgb_init_dli
	.... 1111		*
	1111 1111		*
	1111 1111		*

**Trace Entry: Online Recovery Service Request Processor Termination**

**A003**

Table 69. Trace Record 3702 - Create Data Set Routine Invoke DYA

Module: DFSRWM00 - Database Recovery Manager Master ITASK

Explanation: Record cut when DFSRWM00 is terminating

Trace Subcode RWM00 Exit

Offset	Type	Length	Description
4	Fixed	2	Requested function(awrwfunc)
6	Bit	2	RWGB_Flags
	1... ..		rwgb_startup_complete
	.1.. ..		rwgb_startup_failure
	..1. ..		rwgb_dlisas
	...1 ..		rwgb_fp_allowed
	.... 1..		*
	.... .1..		rwgb_terminating
	.... ..1.		rwgb_record_pipe_alloc
	.... ...1		rwgb_drm_init_complete
	1... ..		rwgb_ORs_installed
	.1.. ..		rwgb_DRF_installed
	..11 1111		*
8	Fixed	4	Request feedback (awrwfdbk)
12	Address	4	Rwgb_hold_queue
16	Address	4	awrwcecb
20	Fixed	4	Awrwcecb->c_ecb
24	Bit	4	Rwgb_init_flags
	1... ..		rwgb_init_load_1
	.1.. ..		rwgb_init_rwsp
	..1. ....		rwgb_init_load_2
	...1 ..		*
	.... 1..		rwgb_init_ascre
	.... .1..		rwgb_init_route
	.... ..1.		rwgb_init_write
	.... ...1		rwgb_init_read
	1... ..		rwgb_init_cmd
	.1.. ..		rwgb_init_rtB
	..1. ....		rwgb_init_fp
	...1 ..		rwgb_init_dli
	.... 1111		*
	1111 1111		*
	1111 1111		*

Trace Entry: Online Recovery Pipe Receive Entry

A040

Table 70. Trace Record 3702 - Create Data Set Routine Invoke DYA

Module: DFSRWPR0 - Database Recovery Manager Record Receive Processor

Explanation: Record cut when DFSRWPR0 is entered

Trace Subcode RWPR0 Entry

Offset	Type	Length	Description
4	Fixed	2	Requested function(awrwfunc)
6	Bit	2	RWGB_Flags
	1... ..		rwgb_startup_complete
	.1.. ..		rwgb_startup_failure
	..1. ....		rwgb_dlisas
	...1 ..		rwgb_fp_allowed
	.... 1..		*
	.... .1..		rwgb_terminating
	.... ..1.		rwgb_record_pipe_alloc
	.... ...1		rwgb_drm_init_complete
	1... ..		rwgb_ORs_installed
	.1.. ..		rwgb_DRF_installed
	..11 1111		*

Table 70. Trace Record 3702 - Create Data Set Routine Invoke DYA (continued)

8	Bit	4	Rwgb_init_flags
	1... ..		rwgb_init_load_1
	.1.. ..		rwgb_init_rwsp
	..1. ..		rwgb_init_load_2
	...1 ..		*
	.... 1..		rwgb_init_ascre
	.... .1..		rwgb_init_route
	.... ..1.		rwgb_init_write
	.... ...1		rwgb_init_read
	1... ..		rwgb_init_cmd
	.1.. ..		rwgb_init_rtb
	..1. ..		rwgb_init_fp
	...1 ..		rwgb_init_dli
	.... 1111		*
	1111 1111		*
	1111 1111		*
12	Bit	4	Rwgb_read_flags
	1... ..		rwgb_read_terminating
	.1.. ..		rwgb_read_abend
	..1. ..		rwg_read_EODAD
	...1 ..		rwgb_read_open
	.... 1111		*
	1111 1111		*
	1111 1111		*
	1111 1111		*
16	Address	4	Rwgb_read_pipe
20	Address	4	Rwgb_read_buffer

**Trace Entry: Online Recovery Pipe Received Record**

**A041**

Table 71. Trace Record 3702 - Create Data Set Routine Invoke DYA

**Module:** DFSRWPR0 - Database Recovery Manager Record Receive Processor

**Explanation:** Record cut when DFSRWPR0 receives record from recovery product

**Trace Subcode** RWPR0 Record

Offset	Type	Length	Description
4	Fixed	2	Requested function(awrwfunc)
6	Bit	2	RWGB_Flags
	1... ..		rwgb_startup_complete
	.1.. ..		rwgb_startup_failure
	..1. ..		rwgb_dlisas
	...1 ..		rwgb_fp_allowed
	.... 1..		*
	.... .1..		rwgb_terminating
	.... ..1.		rwgb_record_pipe_alloc
	.... ...1		rwgb_drm_init_complete
	1... ..		rwgb_ORS_installed
	.1.. ..		rwgb_DRF_installed
	..11 1111		*

Table 71. Trace Record 3702 - Create Data Set Routine Invoke DYA (continued)

8	Bit	4	Rwgb_init_flags
	1... ..		rwgb_init_load_1
	.1.. ..		rwgb_init_rwsp
	..1. ..		rwgb_init_load_2
	...1 ..		*
	.... 1..		rwgb_init_ascre
	.... .1..		rwgb_init_route
	.... ..1.		rwgb_init_write
	.... ...1		rwgb_init_read
	1... ..		rwgb_init_cmd
	.1.. ..		rwgb_init_rt
	..1. ..		rwgb_init_fp
	...1 ..		rwgb_init_dli
	.... 1111		*
	1111 1111		*
	1111 1111		*
12	Bit	4	Rwgb_read_flags
	1... ..		rwgb_read_terminating
	.1.. ..		rwgb_read_abend
	..1. ..		rwg_read_EODAD
	...1 ..		rwgb_read_open
	.... 1111		*
	1111 1111		*
	1111 1111		*
	1111 1111		*
16	Address	4	Address of record received
20	Fixed	2	Record type (logrc_type)
22	Fixed	2	Record subtype (logrc_subtype)

## Online Recovery Manager Trace Example

The following example shows trace output for the Online Recovery Manager trace.

```

OPTION PRINT 0=5,V=67FA,EXITR=DFSERA60
END
FUNCTION      WORD 0      WORD 1      WORD 2      WORD 3      WORD 4      WORD 5      WORD 6      WORD 7
* ORI TRACE TABLE - DATE 2004209 TIME 212317790537 OFFSET 028D SKIP 0000 TOTAL SKIP 00000000 RECORD NUMBER 000022B9
RWM00 Request  A0019700  00A63040  0B7143F8  00000000  00000000  040C0000  00000000  09B18BAA
RWPR0 Entry    A040978A  00A13040  E7000000  00000000  00000000  00000000  40404040  09FB93AC
RWM00 Return  A00297A8  00A6B040  00000000  0B7143F8  00000000  00000000  EF000000  0A081EA6
RWPR0 Entry    A04097AD  00BDB040  EF000000  00000000  0C666B78  0CC74FD8  40404040  0A082039
RWPR0 Record  A04197AE  00BDB040  EF000000  00000000  0C666B78  0CC74FD8  40404040  0DD65C41
RWM00 Request  A00197B3  00A6B040  0B7143F8  0B7143B0  00000000  040C0000  EF000000  0DD65E3C
RWM00 Return  A00297B4  00A6B040  00000000  0B7143F8  00000000  00000000  EF000000  0DD65E63
RWM00 Request  A00197B5  00B4B040  0B7143B0  00000000  00000000  040C0000  EF000000  0DD65E6E
RWM00 Return  A00297B9  00B4B040  00000000  0B7143F8  00000000  040C0000  EF000000  0DD65EDB
RWM00 Request  A00197BA  00A6B040  0B7143F8  0B714518  00000000  040C0000  EF000000  0DD65EE7
RWM00 Return  A00297BB  00A6B040  00000000  00000000  00000000  040C0000  EF000000  0DD65F28
RWM00 Request  A00197BC  00BEB040  0B714518  00000000  00000000  040C0000  EF000000  0DD65F34
RWM00 Return  A00297BD  00BEB040  00000000  00000000  00000000  040C0000  EF000000  0DD65F40
RWPR0 Record  A04197C5  00BDB040  EF000000  00000000  0C666B78  0CC74FD8  40404040  0EAF4E7B
RWPR0 Record  A04197C9  00BDB040  EF000000  00000000  0C666B78  0CC74FD8  40404040  0EAF549C
RWPR0 Record  A0419843  00BDB040  EF000000  00000000  0C666B78  0CC74FD8  40404040  0EAF96C7
RWPR0 Record  A0419844  00BDB040  EF000000  00000000  0C666B78  0CC74FD8  40404040  0EAF9AE2
RWM00 Request  A0019849  00C4B040  0B7143B0  00000000  00000000  040C0000  EF000000  0EAF1AA
RWM00 Return  A002984A  00C4B040  00000000  00000000  00000000  040C0000  EF000000  0EAF1AE2
RWM00 Request  A001984B  00BEB040  0B7144D0  00000000  00000000  040C0000  EF000000  0EAF1F0
RWM00 Return  A002984C  00BEB040  00000000  00000000  00000000  040C0000  EF000000  0EAF201
RWM00 Request  A0019AE6  00A8B040  0B7144D0  00000000  00000000  040C0000  EF000000  20A9CABE
RWM00 Return  A0029AEA  00A8B040  00000000  00000000  00000000  040C0000  EF000000  20A9CDC0
RWPR0 Record  A0419B84  00BDB040  EF000000  00000000  0C666B78  0CC74FD8  40404040  210B71FC
RWPR0 Record  A0419B88  00BDB040  EF000000  00000000  0C666B78  0CC74FD8  40404040  210B7947

```



## 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.

The following topics provide additional information:

- “Limiting Locking Resources Used by an Application Program”
- “Program Isolation (PI) Trace”
- “DL/I Call Image Capture Program” on page 312

## 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 8 on page 127.

### 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 9: 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 264.

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 9: Utilities Reference: Database and Transaction Manager* for a description of the File Select and Formatting Print utility (DFSERA10).

"Format of X'67' Log Record" on page 150 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 DFSDDLTO. 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 257 or *IMS Version 9: 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 72 on page 313 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 131 on page 313.) 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 131 shows the general areas of database analysis: Application, Buffer, Database, Image Copy, and Log.

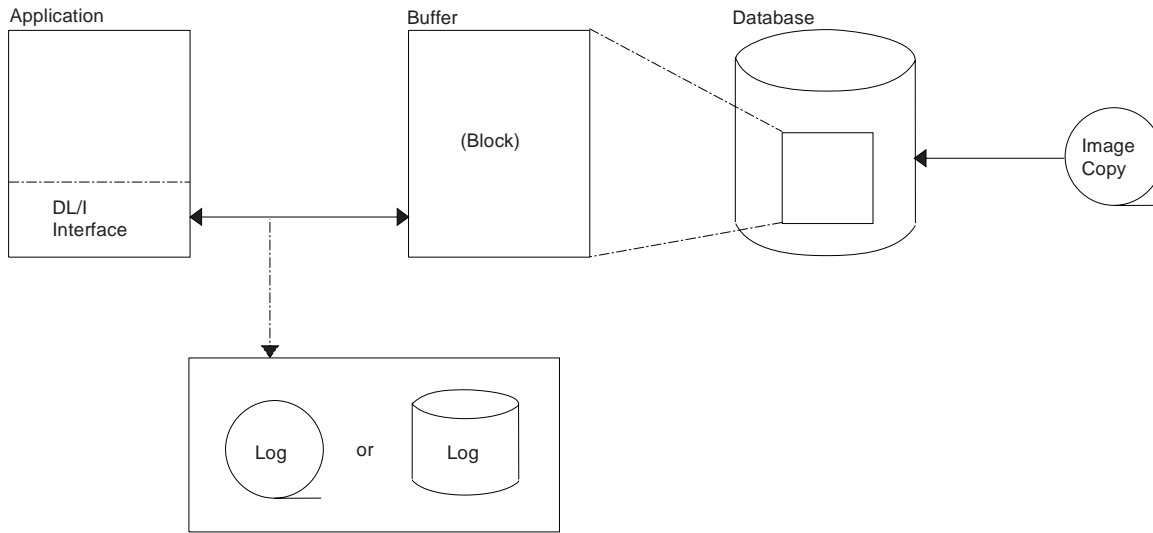


Figure 131. General Areas of Database (DB) Analysis

You can use Table 72 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 72. Database Change Log Record DSECT

Offset	Field	Length	Description
DSECT			
<b>DLOGB</b>			
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'81' Version 6 or 7 X'82' Version 8 or 9
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 72. Database Change 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 tracking X'10' Database is covered by RSR X'08' PHIDAM primary index; no REDO X'04' DLOGSEQ has update sequence number X'02' OLR non-backoutable; cursor not active yet X'01' OLR ITASK
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)  X'80' <ul style="list-style-type: none"> <li>• When this high order bit is on, then this DCB number represents one of the M-through-V or Y data sets.</li> <li>• When this high order bit is off, then this DCB number represents one of the A-through-J or X data sets.</li> </ul>
3D	DDSID2	1	For ARID
3E	DLOGSLVL	1	Database share level (for DBRC-registered databases)
3F	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	DLOGBLK0	2	Offset of RBA within block
48	DLOGSEQ	4	Update the sequence number when X'04' flag is on in DLOGDBF2
4C	DLOGXTOF	2	Database extension section offset (not used) <sup>1</sup>
4E	DLOGDSOF	2	Data sharing section offset <sup>1</sup>
50	DLOGIDOF	2	RACF userid offset <sup>1</sup>
52	DLOGTKOF	2	Tracking (XRF) section offset <sup>1</sup>
54	DLOGDLOF	2	DL/I call section offset (not used) <sup>1</sup>
56	DLOGKYOF	2	Key data section offset <sup>1</sup>
58	DLOGSPOF	2	Space management section offset <sup>1</sup>

Table 72. Database Change Log Record DSECT (continued)

Offset	Field	Length	Description
5A	DLOGUNOF	2	UNDO data offset <sup>1</sup>
5C	DLOGREOF	2	REDO data offset <sup>1</sup>
60	DDATE	4	Date in the format YYYYDDDF
64	DTIME	6	Time in the format HHMMSSTHMIJU
6A	DZONE	2	Offset to local time
<b>Data Sharing Section (DLOGDSHUR DSECT)</b>			
00	DLOGDSSN	4	Data set sequence number (DSSN)
04	DLOGLSN	6	Lock sequence number (LSN)
0A	DLOGUSID	4	Update Set ID (USID)
<b>RACF/SIGNON Userid (DLOGID DSECT)</b>			
00	DLOGUSER	8	RACF userid
<b>Buffer and Lock Tracking for DL/I in XRF-capable Systems (DLOGTRCK DSECT)</b>			
00	DLOGPOOL	2	Pool size for buffer tracking
02	DLOGBUFF	2	Buffer number for buffer tracking
04	DLOGHASH	4	Root hash value
08	DLOGLOCK	4	Lock value
0C	DLOGFL1	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
0D	DLOGFL2	1	Reserved
0E	DLOGDBDN	8	DBD name
16	DLOGSKID	4	Task ID
<b>KSDS Key Data Section (DLOGKEY DSECT)</b>			
00	DLOGKYF1	1	X'40' KSDS key X'20' Key is being erased
02	DLOGKLEN	2	Length of key
04	DLOGKDAT	variable	Key data
<b>Space Management Section for HD Inserts and Deletes (DLOGSPCE DSECT)</b>			
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
<b>UNDO/REDO Data Section (DLOGDATA DSECT)</b>			

Table 72. Database Change Log Record DSECT (continued)

Offset	Field	Length	Description
00	DLOGDFLG	1	X'80' Last data element in this section X'40' Data is compressed using z/OS 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 <sup>2</sup>
	DBLGSEG	8	Logical logger sequence number <sup>2</sup>

**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:

- DL/I trace table entries: “DL/I Trace” on page 263
- Dump formatting of IMS control blocks: “Using Interactive Dump Formatter Menus” on page 179
- SNAPs of IMS control blocks during pseudoabends: “SNAPs on Exceptional Conditions” on page 256

The //DFSSTAT statistics report is also a useful tool for evaluating a potential Sequential Buffering problem. For information about //DFSSTAT, see *IMS Version 9: 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.

| The following topics provide additional information:

- | • “SBSNAP Option”
- | • “SBESNAP Option” on page 318
- | • “SB IMAGE CAPTURE Option and SB Test Program (DFSSBHD0 Utility)” on page 318
- | • “SB COMPARE Option” on page 318

## 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 9: 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 (DFSERA10) with exit routine DFSERA30. For information about this utility, see *IMS Version 9: 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*th call to the SB buffer handler; specifying STOP=*m* deactivates the SBSNAP option during the *m*th 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 9: 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 (DFSERA10) with exit DFSERA30. For information about this utility, see *IMS Version 9: 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 9: Installation Volume 2: System Definition and Tailoring* for more information). The SB Test program (DFSSBHD0 utility) is described in the *IMS Version 9: 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 9: 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 (DFSERA10) with exit DFSERA30. For information about this utility, see *IMS Version 9: 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 132 on page 320 shows an example of a formatted GSAM control block dump, and Figure 133 on page 321 shows an example of an unformatted GSAM control block dump.

| The following topics provide additional information:

- | • “Example of a Formatted GSAM Control Block Dump”
- | • “Example of an Unformatted GSAM Control Block Dump” on page 321
- | • “Recovering from Out-of-Space Sx37 Abends on GSAM Data Sets” on page 322

## Example of a Formatted GSAM Control Block Dump

In Figure 132 on page 320, 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
GPTCNTRL 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 GLTCBDM 8007CCB0 GLTCNTRL 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 GBUFNO 01
GBCDISP 0000 GBCHAIN 0007A220 GBCRTNCD 0028 GBCSEVCD 08 GBCTRS 0000
GBDCBPTR 8007A178 GBDDNAME GS378770 GBDECB 0007A1D4 GBDEVYTP 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 00091888
GBSERA 0000 GBSERR 0600 GBSUPVR 00 GBTRCALC BB60 GBTRECL 0096
GBURTNC D AM GBVLSQ 0001

07A178 DATA CONTROL BLOCK (DCB)
DCBBFTEK 06 DCBBLKCT 04FDBEBC DCBBLKSI 01C2 DCBBUFCB 01064C98 DCBBUFL 01C2
DCBBUFNO 01 DCBBUFUF 00 DCBCECHK 00C89480 DCBCIND1 00 DCBCIND2 00
DCBCNTRL 00D57F48 DCBDDNAM DCBDEBAD 009D1554 DCBDEN AD DCBDEVT 2E
DCBDSORG 4000 DCBDVTBA FDBEBC DCBE0BR 01D57650 DCBE0BW 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 DCBOPCTD 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 8007A178 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 08 GBUFNO 01
GBCDISP 0000 GBCHAIN 0007A0C0 GBCRTNCD 0000 GBCSEVCD 08 GBCTRS 0000
GBDCBPTR 8007A2D8 GBDDNAME GS378770 GBDECB 0007A334 GBDEVYTP 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
GBURTNC D GBVLSQ 0000

```

Figure 132. 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 DCBBUF0F 00 DCBCECHK 00000001 DCBCIND1 00 DCBCIND2 00
DCBCNTRL 00000001 DCBDDNAM GS378770 DCBDEBAD F8F7F7D6 DCBDEN 00 DCBDEV 00
DCBDSORG 4000 DCBDVTBA 000000 DCBE0BR 01000001 DCBE0BW 00000001 DCBE0DA 0000001
DCBE0DAD 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
DCBREFCM 00 DCBREL 000000 DCBRELAD 00000000 DCBRELB 00000000 DCBSTACK 00
DCBSVXL 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 132. 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 00000000 0000FFFF 00000000 00000000 00000000 *.... *
0007A0C0 0007A220 00000401 00010000 00010096 00000096 01C20000 0000208E 0001FFE0 *.....B..... *
0007A0E0 40400000 00289081 06000000 02830283 12020000 5008D101 00000000 00000000 * .....B.....J..... *
0007A100 0007A010 0007A090 B007A178 0007A1D4 8607BEA2 00093300 00091B88 00000000 *.....M..... *
0007A120 00064CA0 00064D36 BB60BB60 00000000 C7E2F3F7 F8F7F7D6 00000000 00000000 *.....GS378770..... *
0007A140 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 00000000 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 *..... *
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 00093300 0004DD54 00000000 00000000 00029E50 *..... *
0007A7E0 00000000 00000000 00000000 00000000 C4C4D3E3 F0F1F340 D3D6C1C4 40404040 *.....DDL013 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 00000000 000300CC 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 00000000 00000000 00010C00 0004D350 *.....L.. *
0007A980 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 *..... *

```

Figure 133. Unformatted GSAM Control Block Dump (Part 1 of 2)

```

LINE 0007A9A0 SAME AS ABOVE
0007A9C0 84000000 00000000 00000000 00000000 00000000 00000000 00000000 *.....*
0007A9E0 00000000 00000000 00000000 00000000 00000000 0088266F 11173205 02000000 *.....*
0007AA00 0004D050 00000000 00000000 00000000 00000000 00000000 00000000 *.....*
0007AA20 00000000 00000000 00000000 00000000 00000000 00000000 00000000 *.....*
0007AA40 00060040 00000000 00000000 00000000 00000000 00000000 00000000 *.....*
0007AA60 00000000 00000000 00000000 00000000 00000000 00000000 00000000 *.....*
0007AA80 00000000 00000000 00000000 00000000 00000000 00000000 0002CEE6 *.....W*
0007AAA0 00000000 00000000 00000000 0005713F 00057040 00000000 07FC040 40404040 *.....*
0007AAC0 00000000 00000000 00000000 00057340 00000000 00057140 00000000 00000000 *.....*
0007AAE0 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 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 *..P..P.....*
0007AE20 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 *.....*
0007AE80 00000000 00000000 00000000 00000000 0004D794 0004D824 00000000 00000000 *.....P..Q.....*
0007AEA0 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 *.....*

LINE 0007AF00 SAME AS ABOVE
0007AF20 0004D824 0008BD98 00000000 00000000 00000000 00000000 00000000 *..Q.....*
0007AF40 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 *.....*
0007AFA0 00000000 00000000 00000000 00000000 0004D8B4 00000000 00000000 00000000 *.....Q.....*
0007AFC0 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..DFSFLG0.130.LOGICA*

```

Figure 133. 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 IMSs method of checkpointing the position in the data set. For information about repositioning GSAM data sets, see the “XRST Call” section in *IMS Version 9: 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 section describes diagnostic aids and techniques used during data communication problem analysis. It does not apply to a Database Control (DBCTL) environment. Included are:

- | • “Terminal Communication Task Trace” discusses the terminal communication task trace, which shows the last few communications analyzer and device-dependent module interactions.
- | • “DC Trace” on page 327 discusses the data communication (DC) trace, which accumulates a history of device and line activity on the IMS log data set.
- | • “Diagnosing Problems in the Queue Control Facility/Message Requeuer” on page 341 discusses the problem diagnosis in the Queue Control Facility/Message Requeuer.
- | • “Diagnosing Message Routing Problems” on page 348 discusses diagnosis of message routing problems.
- | • “IMS Transaction Trace” on page 358 discusses the IMS Transaction trace, which is useful in analyzing problems associated with IMS and the application program.
- | • “Receive-Any Buffer Analysis” on page 361 discusses a procedure to help you determine if any receive-any buffers are left.
- | • “Finding the Active Save Set” on page 363 discusses a procedure to help you find the active save set.
- | • “IMS-VTAM Interface” on page 363 provides a description of the IMS-VTAM interface.
- | • “IBM 3270 Error Recovery Analysis” on page 363 discusses IBM 3270 error recovery analysis.
- | • “Message Format Service Normal BTAM Path” on page 364 discusses Message Format Service normal logic flow for BTAM activity.
- | • “Message Format Service Module Traces” on page 371 discusses Message Format Service module traces.
- | • “Tracing Errors in Module DFSCNXA0” on page 372 discusses tracing errors in module DFSCNXA0.
- | • “IDC0 Trace Table Entries” on page 379 discusses IDC0 trace table entries.
- | • “APPC/IMS Diagnostic Aids” on page 382 discusses the APPC/IMS diagnostic aids.
- | • “OTMA Diagnostic Aids” on page 400 discusses diagnostic information to help you analyze problems in OTMA.
- | • “Diagnosing Errors Diagnosing Fast Path Problemsrelated to Print Data Set Options: IMS Spool API Support” on page 405 discusses diagnosis of errors related to print data set options.

---

### 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.

- | The following topics provide additional information:

- | • “Entry Points”
- | • “Trace Records”
- | • “Trace Output” on page 327

## 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

- |          |   |
|----------|---|
| <b>1</b> | Process an input segment from a terminal.   |
| <b>2</b> | Perform a logical read operation to the terminal.   |
| <b>3</b> | Determine which system function is to be performed next for this line and terminal (or node). |
| <b>4</b> | Issue GET NEXT to message queue.  |
| <b>5</b> | Perform a logical write operation to the terminal.  |
| <b>6</b> | WRITE successful; dequeue message or call DDM at DD1.   |
| <b>7</b> | Notify master terminal of I/O error; cancel input; return output message to queue.            |
| <b>8</b> | Return output message to queue; cancel input.   |
| <b>9</b> | Generate an error message; cancel input; return output message to queue.                      |
| <b>A</b> | Idle the line; cancel output; return output message to queue.                                 |
| <b>B</b> | Resend the last message sent from a given LTERM.  |
| <b>C</b> | 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

- |          |   |
|----------|---|
| <b>1</b> | WRITE/SEND setup: Set up output buffer to write current buffer.                                   |
| <b>2</b> | WRITE/SEND interruption: Error check last output operation.                                       |
| <b>3</b> | READ/RECEIVE setup: Set up to perform a poll or read.   |
| <b>4</b> | READ/RECEIVE interruption: Error check, determine terminal responding, and deblock input segment. |
| <b>5</b> | Cleanup: Restore control blocks after DFSICIO0 error.   |
| <b>6</b> | Build: Move output message from a queue buffer (MFS buffer) to a line buffer.                     |
| <b>7</b> | Logon: VTAM OPNDST/CLSDST processing.   |
| <b>8</b> | Prepare for output: VTAM  |
| <b>F</b> | MFS output format control (DFSCOF0) 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 134 shows the format of a sample terminal communications task trace record.

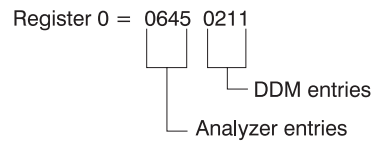


Figure 134. Example of a Terminal Communication Task Trace Entry

The sample terminal communication task trace entry in Figure 134 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 73.

Table 73. 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.
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 326). 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).

- | The following topics provide additional information:
- | • “Starting the Trace”
- | • “Stopping the Trace” on page 328
- | • “Printing the Trace Records” on page 329
- | • “Content of the Trace Records” on page 330
- | • “Diagnosing Line and Terminal Problems” on page 334

## 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 z/OS 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

**Note:**

- 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.

For a detailed description of the /TRACE command, see *IMS Version 9: Command Reference*.

## Stopping the Trace

To stop the DC trace, enter one of the following commands from the master terminal or the z/OS 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 format and print the trace records, use either of the following methods:

- **Knowledge-Based Log Analysis (KBLA)**

The KBLA Basic Record Formatting and Print Module (DFSKBLA3) creates log record header information that describes what the log record identifier represents, the time stamp at which the record was written, and allows the interpretation of the trace entries without using Table 74 on page 331.

For a high level view of the trace events flow, you can use the KBLA Summary Record Formatting Module (DFSKBLA8) to print only the header descriptions of the entries (no data).

For more information, see the “KBLA Log Formatting Modules” section in *IMS Version 9: Utilities Reference: System*.

- **File Select and Formatting Print utility (DFSERA10)**

To 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 0=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 0=5,T=X,L=1,V=67,C=M
OPTION PRINT 0=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 (DFSERA10), see *IMS Version 9: 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):<sup>4</sup>

| <b>ID</b>   | <b>Description</b>                                                                                                                                    |
|-------------|-------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>A xx</b> | Communication analyzer activity (DFSICIO0)                                                                                                            |
| <b>AERx</b> | Access method error                                                                                                                                   |
| <b>C xx</b> | Communication analyzer activity (DFSCIOC0 in DFSICIO0)                                                                                                |
| <b>CI04</b> | TM shared queues re-read error detected                                                                                                               |
| <b>CIO2</b> | DDM SDC read for output                                                                                                                               |
| <b>CIO3</b> | DDM conditional SDC "wash" output                                                                                                                     |
| <b>CMEA</b> | Before calling Message Control/Error exit DFSCMUX0                                                                                                    |
| <b>CMEB</b> | After calling Message Control/Error exit DFSCMUX0                                                                                                     |
| <b>CMEI</b> | Message Control/Error exit interface processing                                                                                                       |
| <b>COFC</b> | Entry to the output format control, MFS-supported devices (DFSCOF00)                                                                                  |
| <b>CRTU</b> | Output User Creation user exit routine failure                                                                                                        |
| <b>CVCT</b> | VTAM trace. This log record is written even though DC trace is not active on the terminal/link.                                                       |
| <b>CVCV</b> | 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. |
| <b>D xx</b> | Device-Dependent Module activity (DDM)                                                                                                                |
| <b>DDxx</b> | Output processing by DFSCOF00                                                                                                                         |
| <b>DSIM</b> | SIMLOGON attempt of a dynamic terminal                                                                                                                |
| <b>ESIM</b> | SIMLOGON error for a dynamic terminal                                                                                                                 |
| <b>FERR</b> | MFS-block fetch error                                                                                                                                 |
| <b>FESx</b> | Front-end switch user exit routine activity                                                                                                           |
| <b>FEXT</b> | Before field edit exit routine                                                                                                                        |
| <b>FMTx</b> | Message Format Service activity (MFS)                                                                                                                 |
| <b>HCSW</b> | 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. |
| <b>ICLR</b> | Message router activity                                                                                                                               |

4. An asterisk (\*) in this list is a wildcard character, meaning that any character can replace the asterisk.

- 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 74 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 74. DC Trace Records*

| Trace ID | Function                              | Traced by             | When Traced or /TRACE Option | What Is Traced                                                                 |
|----------|---------------------------------------|-----------------------|------------------------------|--------------------------------------------------------------------------------|
| A01      | Process input. <sup>1</sup>           | DFSICIO0 <sup>9</sup> | ALL, DDM                     | CTB, CLB, CXB, CRB, CIB, CCB, QBUF, IOBUF, INPCNTS, OUTCNTS, EMHB <sup>2</sup> |
| A02      | Do read. <sup>1</sup>                 | DFSICIO0 <sup>9</sup> | ALL, DDM                     | CTB, CLB, CXB, CRB, IOBUF, EMHB <sup>2</sup>                                   |
| A03      | What is next.                         | DFSICIO0 <sup>9</sup> | ALL, DDM                     | CTB, CLB, CRB, CTT                                                             |
| A04      | Get Next segment.                     | DFSICIO0 <sup>9</sup> | ALL, DDM                     | CTB, CLB, CNT                                                                  |
| A05      | Do write. <sup>1</sup>                | DFSICIO0 <sup>9</sup> | ALL, DDM                     | CTB, CLB, CXB, CRB, CCB, IOBUF, EMHB <sup>2</sup>                              |
| A06      | After good write.                     | DFSICIO0 <sup>9</sup> | ALL, DDM                     | IOB, CTB, CLB, CXB, CRB, CCB                                                   |
| A07      | After bad write. <sup>1</sup>         | DFSICIO0 <sup>9</sup> | ALL, DDM                     | IOB, CTB, CLB, CRB, CCB, IOBUF, EMHB <sup>2</sup>                              |
| A08      | Cancel message, do not DEQ.           | DFSICIO0 <sup>9</sup> | ALL, DDM                     | CTB, CLB, CRB                                                                  |
| A09      | Generate system message. <sup>1</sup> | DFSICIO0 <sup>9</sup> | ALL, DDM                     | CTB, CLB, CRB, MFS                                                             |
| A10      | Quiesce without stopping.             | DFSICIO0 <sup>9</sup> | ALL, DDM                     | CTB, CLB, CRB, CCB                                                             |
| A11      | Retrieve last DEQD message.           | DFSICIO0 <sup>9</sup> | ALL, DDM                     | CTB, CLB, CNT, CRB                                                             |

Table 74. DC Trace Records (continued)

| Trace ID | Function                                   | Traced by             | When Traced or /TRACE Option | What Is Traced                                                                                                               |
|----------|--------------------------------------------|-----------------------|------------------------------|------------------------------------------------------------------------------------------------------------------------------|
| A12      | Wait for ASYNC I/O or output ENQ.          | DFSICIO0 <sup>9</sup> | ALL, DDM                     | CTB, CLB, CRB, CCB, IOBUF, EMHB <sup>2</sup>                                                                                 |
| AER1     | Access method error.                       | DFSICIO0 <sup>9</sup> | Always                       | CTB, CLB, CNT, QBUF, SALL, CTT, PCB                                                                                          |
| AER2     | Access method error. <sup>3, 1</sup>       | DFSICIO0 <sup>9</sup> | Always                       | IOB, CTB, CLB, CNT, CXB, CRB, CIB, CCB, QBUF, IOBUF, SALL, CTT, PCB, EMHB <sup>2</sup>                                       |
| C00      | Get queue buffer.                          | DFSICIO0 <sup>9</sup> | ALL, MFS                     | CTB, CNT, CIB, SALL                                                                                                          |
| C01      | Reposition queue buffer.                   | DFSICIO0 <sup>9</sup> | ALL, MFS                     | CTB, CNT, CIB, SALL                                                                                                          |
| C02      | Get Next.                                  | DFSICIO0 <sup>9</sup> | ALL, MFS                     | CTB, CNT, CIB, SALL                                                                                                          |
| C03      | DEQ output.                                | DFSICIO0 <sup>9</sup> | ALL, MFS                     | CTB, CNT, CIB, SALL                                                                                                          |
| C04      | Place output back in queue.                | DFSICIO0 <sup>9</sup> | ALL, MFS                     | CTB, CNT, CIB, SALL                                                                                                          |
| C05      | Find output.                               | DFSICIO0 <sup>9</sup> | ALL, MFS                     | CTB, CNT, CIB, SALL                                                                                                          |
| C06      | Get new output message or QMGR call.       | DFSICIO0 <sup>9</sup> | ALL, MFS                     | CTB, CNT, CIB, SALL                                                                                                          |
| C07      | Free input buffer.                         | DFSICIO0 <sup>9</sup> | ALL, MFS                     | CTB, CNT, CIB, SALL                                                                                                          |
| C08      | Get output buffer.                         | DFSICIO0 <sup>9</sup> | ALL, MFS                     | CTB, CNT, CIB, SALL                                                                                                          |
| C09      | User output edit.                          | DFSICIO0 <sup>9</sup> | ALL, MFS                     | CTB, CNT, CIB, SALL                                                                                                          |
| C10      | Call queue MGR.                            | DFSICIO0 <sup>9</sup> | ALL, MFS                     | CTB, CNT, CIB, SALL                                                                                                          |
| C11      | Get DDM work buffer.                       | DFSICIO0 <sup>9</sup> | ALL, MFS                     | CTB, CNT, CIB, SALL                                                                                                          |
| C12      | Free DDM work buffer.                      | DFSICIO0 <sup>9</sup> | ALL, MFS                     | CTB, CNT, CIB, SALL                                                                                                          |
| C13      | Free receive-any buffer.                   | DFSICIO0 <sup>9</sup> | 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 call MSG CTRL Error exit.           | DFSCMEI0              | Before call DFSCMUX0         | If ITASK is a CLB or LLB: CTB, CLB, CRB, QBUF, IOBUF, INP/OUTP CNTS, DDM, MSNB                                               |
| CMEB     | After call MSG CTRL Error exit.            | DFSCMEI0              | After call DFSCMUX0          | If ITASK is a CLB or LLB: CTB, CLB, CRB, QBUF, IOBUF, INP/OUTP CNTS, DDM, MSNB<br>If ITASK is a PST: PST, MSGPRFX, SMB, MSNB |
| CMEI     | Error procedure in DFSCMEI0.               | DFSCMEI0              | On some errors               | If ITASK is a CLB or LLB: CTB, CLB, CRB, QBUF, IOBUF, INP/OUTP CNTS, DDM, MSNB<br>If ITASK is a PST: PST, MSGPRFX, SMB, MSNB |
| COFC     | Let MFS edit output.                       | DFSICIO0 <sup>9</sup> | ALL, DDM                     | CTB, CLB, CNT, CRB, CIB, IOBUF, EMHB <sup>2</sup>                                                                            |
| CRTU     | Output User Creation exit routine failure. | DFSCRTU0              | Always                       | See notes <sup>10</sup>                                                                                                      |
| CVCT     | VTAM TRACE/ABORT. <sup>1</sup>             | DFSCVCT0              | ALL, DDM                     | CTB, CLB, CNT, CRB, IOBUF, CTT, INPCNTS, EMHB <sup>2</sup>                                                                   |
| CVCV     | XRF class 2 takeover. <sup>1</sup>         | DFSCVCV0              | Always                       | CLB, CTB, CTT, LLB, LTb, LXB, LU6WA, CNT, CRB, SPQB, CTC, MSNB, EMHB, IOBUF, DDM                                             |
| D01      | Write setup.                               | DFSICIO0 <sup>9</sup> | ALL, DDM                     | CTB, CLB, CNT, CRB, CIB, QBUF, S25                                                                                           |
| D02      | Write interrupt. <sup>1</sup>              | DFSICIO0 <sup>9</sup> | ALL, DDM                     | IOB, CTB, CLB, CRB, IOBUF, S25, EMHB <sup>2</sup>                                                                            |

Table 74. DC Trace Records (continued)

| Trace ID          | Function                                  | Traced by                          | When Traced or /TRACE Option | What Is Traced                                                                    |
|-------------------|-------------------------------------------|------------------------------------|------------------------------|-----------------------------------------------------------------------------------|
| D03               | Read setup.                               | DFSICIO0 <sup>9</sup>              | ALL, DDM                     | CTB, CLB, CNT, CRB                                                                |
| D04               | Read interrupt. <sup>1</sup>              | DFSICIO0 <sup>9</sup>              | ALL, DDM                     | IOB, CTB, CLB, CRB, IOBUF, S25, EMHB <sup>2</sup>                                 |
| D05               | Cleanup.                                  | DFSICIO0 <sup>9</sup>              | ALL, DDM                     | IOB, CTB, CLB, CNT, CXB, CRB, CIB, CCB, MFS, QBUF, IOBUF, SALL, EMHB <sup>2</sup> |
| D07               | LOGON. <sup>1</sup>                       | DFSICIO0 <sup>9</sup>              | ALL, DDM                     | CTB, CLB, CNT, CRB                                                                |
| DD6M              | Output build (MFS).                       | DFSCOF0                            | ALL, DDM                     | CTB, CLB, CNT, CRB, CIB, SEG, MFS, IOBUF, S25, EMHB <sup>2</sup>                  |
| DD6S              | Output build (Non-MFS).                   | DFSCOF0                            | ALL, DDM                     | CTB, CLB, CNT, CRB, CIB, IOBUF, S25, EMHB <sup>2</sup>                            |
| DD8               | Prepare for output.                       | DFSCOF0                            | ALL, DDM                     | CTB, CLB, CNT, CRB, CIB, IOBUF, S25, EMHB <sup>2</sup>                            |
| DDM1              | Write set up through COFC.                | DFSCOF0                            | ALL, DDM                     | CTB, CLB, CNT, CRB, CIB, MFS, IOBUF, S25, EMHB <sup>2</sup>                       |
| FERR              | MFS block fetch error. <sup>3</sup>       | DFSCFEO0                           | Always                       | CIB, CTT, MFSBP, MFSTRACE 4                                                       |
| FES1              | Entry to front end switch user exit.      | DFSICIO0 <sup>9</sup>              |                              | CTB, CLB, CNT, QBUF, S25                                                          |
| FES2              | Exit from front end switch user exit.     | DFSICIO0 <sup>9</sup>              |                              | CTB, CLB, CNT, QBUF, S25                                                          |
| FEXT <sup>5</sup> | Before field edit exit.                   | DFSCFEI0                           | MFS                          | CTB, CIB                                                                          |
| FMT1              | Return from DFSFEIO or unformatted input. | DFSICIO0 <sup>9</sup>              | ALL, MFS                     | CTB, CLB, CIB, IOBUF, EMHB <sup>2</sup>                                           |
| FMT2              | MFS go to DFSFEIO formatted input.        | DFSICIO0 <sup>9</sup>              | ALL, MFS                     | CTB, CLB, CIB, IOBUF, EMHB <sup>2</sup>                                           |
| FMT3              | MFS complete process MSG segment.         | DFSICIO0 <sup>9</sup>              | ALL, MFS                     | CTB, CLB, CIB, MFS, QBUF                                                          |
| FMT4              | Get next input.                           | DFSICIO0 <sup>9</sup>              | ALL, MFS                     | CTB, CLB, CIB                                                                     |
| FMT6              | Clean up resources.                       | DFSICIO0 <sup>9</sup>              | ALL, MFS                     | CTB, CLB, CIB                                                                     |
| HCSW              | XRF class 1 takeover. <sup>1</sup>        | DFSHCSW0                           | Always                       | IOBUF, CNT, CRB, CTT, CTB, CLB                                                    |
| ICLR              | Message router.                           | DFSICLR0                           | Always                       | CTB, CLB, CTT, PCB                                                                |
| MTRP <sup>8</sup> | Block verification error.                 | DFSCFEO0                           |                              | CLB, CIB, MFS, CTT                                                                |
| MTRP <sup>7</sup> | 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                      | DFSCIO4                            | ALL DDM                      | copy ctl blk list from CVCT entry                                                 |
| SEXT <sup>6</sup> | Before segment edit exit.                 | DFSCFEI0                           | MFS                          | CTB, CIB                                                                          |
| TRCE              | Non-SNA 3270 error.                       | DFSDN130,<br>DFSDN140,<br>DFSDS060 | Always                       | IOB, CTB, CLB, S25, CTT                                                           |
| VTPO              | Rejected posting of ECB.                  | DFSVTPO0                           | ALL, DDM                     | See notes <sup>11</sup>                                                           |

Table 74. DC Trace Records (continued)

| Trace ID | Function | Traced by | When Traced or /TRACE Option | What Is Traced |
|----------|----------|-----------|------------------------------|----------------|
|----------|----------|-----------|------------------------------|----------------|

**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**

|          |                           |       |                            |
|----------|---------------------------|-------|----------------------------|
| <b>0</b> | Current pool space in use |       |                            |
| <b>4</b> | Maximum space used        |       |                            |
| <b>5</b> | 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 |
| <b>9</b> | Error status              |       |                            |
|          |                           | X'BB' | BLDL error                 |
|          |                           | X'FF' | READ error                 |

|           |                            |
|-----------|----------------------------|
| <b>A</b>  | Block name for BLDL error  |
| <b>10</b> | BLDL return code on error  |
| <b>12</b> | Sense from read error      |
| <b>14</b> | CSW status from read error |
| <b>16</b> | Block name for read error  |
| <b>20</b> | 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 is 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 336.
11. The VTPO trace entry is mapped in "Format of 6701 Log Record with VTPO Identifier" on page 337.

## 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 336.

**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 CLBFLAG 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 337.

**Format of 6701 Log Record with CRTU Identifier**

Table 75 provides a map of the formatted CRTU log record.

*Table 75. 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     | XL68     | Internal use                     |
| +4C    | CL8      | Input Lterm Name                 |
| +54    | 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.
56  STORAGE MANAGER ERROR - DFSPPOOL.
60  ERROR IN ADDING DYNAMIC SMB TO HASH TABLE.
64  INSERT EXIT (DFSINSX0) PARAMETER ERROR:
   INVALID DYNAMIC TRANSACTION DATA.
68  LOCAL CNT FOUND, BUT DESTINATION REGISTERED
   TO RESOURCE MANAGER AS A TRANSACTION.
72  LOCAL SMB FOUND, BUT DESTINATION REGISTERED
   TO RESOURCE MANAGER AS AN LTERM.
76  DESTINATION REGISTERED TO RESOURCE MANAGER AS
   A CPIC TRANSACTION, APPC DESCRIPTOR, OR MSNAME.
80  DESTINATION COULD NOT BE VALIDATED IN RESOURCE
   MANAGER DUE TO AN RM INTERFACE ERROR.
84  SMB CREATION REQUESTED, BUT DESTINATION WAS ALREADY
   REGISTERED TO RESOURCE MANAGER AS AN LTERM, CPIC
   TRANSACTION, APPC DESCRIPTOR, OR MSNAME.
88  SMB CREATION REQUESTED, BUT SHARED QUEUES IS NOT
   ACTIVE.

```

### 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 76 shows the VTCB Posting in DFSVTPO0.

Table 76. 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 76. 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 <sup>1</sup>                                          |
|        | 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 76. 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  FFFF9099 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  FFFF9099 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..H.....*
022480D8 000060  00000000 02235000  00000000 00000000  00000000 00000000  40000000 00000000  *.....&;.....*
022480F8 000080  00000000  *.....*

```

Figure 135. 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...BOP0 *
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 ..... *

```

Figure 135. Data Communication (DC) Trace Records (Part 2 of 2)

## Diagnosing Problems in the Queue Control Facility/Message Requeuer

The queue control facility (QCF)/message requeuer (MRQ) processor module (DFSQMRQ0), which is part of the IMS Transaction Manager (TM) component, provides diagnostics for diagnosing errors while running the IMS Queue Control Facility (QCF) licensed program (5697-099). Although problems can be diagnosed separately in the QCF product using SCRAPLOG records and in the IMS queue control facility processor module using 6701-MRQE diagnostic records, QCF and the queue control facility processor work together to allow inserting/loading, querying, recovering, deleting/unloading, recovering, or viewing messages on the IMS message queue data sets and shared message queue structures. Therefore, this section describes the QCF licensed program and its associated SCRAPLOG diagnostic records, as well as the IMS queue control facility 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 IQCSELCT, IQCCANCL, and IQCINSRT modules of QCF, 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 processed.

- QCF functions are designed to help you do:
  - Message queue recovery when it is desirable to return messages to the IMS queue for reprocessing
  - Application recovery when it is desirable to return messages to the IMS queue for reprocessing
  - IMS queue maintenance (you can query, browse, unload, and load IMS nonshared queue environments)
  - Message queue migration and fallback
  - Stress, regression, and application testing when transaction data is needed to simulate production loads or application input
- A queue overflow protection function monitors queue usage and takes action to prevent queue utilization from reaching critical thresholds (nonshared queue environment).
- An ISPF front end lets you select QCF functions and selection criteria to:
  - Query messages (or IMS status) on the queue
  - Unload (delete) messages from the queue
  - Load messages onto the IMS message queues
  - Release or terminate waiting tasks (nonshared queue environment)
  - Maintain the tables associated with queue overflow protection (nonshared queue environment)

l The following topics provide additional information:

- l • “The Query Control Facility Interface” on page 342
- l • “Using SCRAPLOG Diagnostic Records” on page 343

- | • “Using 6701-MRQE Diagnostic Records” on page 345
- | • “Obtaining Diagnostics in Addition to SCRAPLOG and 6701-MRQE” on page 347
- | • “How to Tell When Messages Have Been Successfully Requeued” on page 348

## The Query Control Facility Interface

Figure 136 details the IMS Query Control Facility (QCF 1.2) Interface to IMS. The figure is explained in the paragraphs that follow.

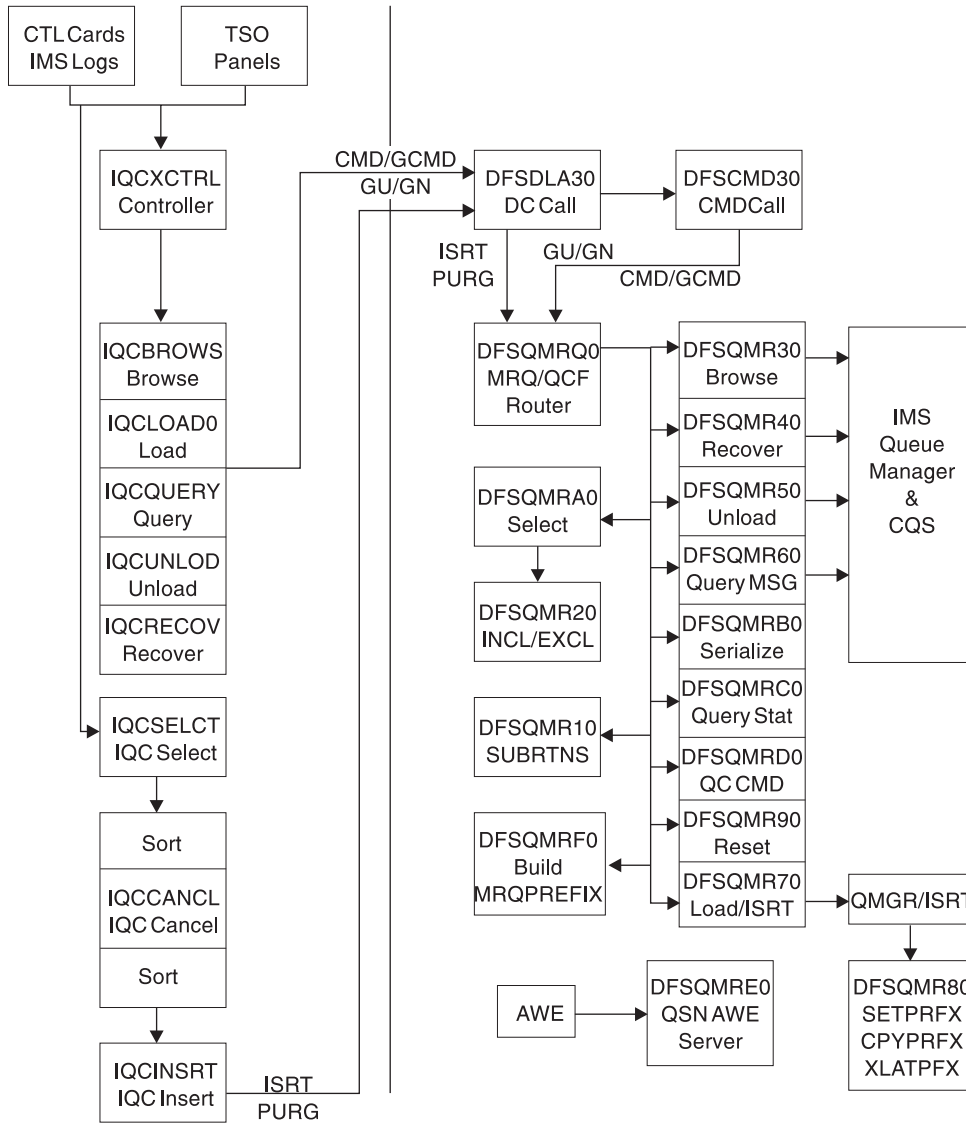


Figure 136. Query Control Facility Interface to IMS

The QCF functions (Figure 136) can be invoked with either user control card input, or a TSO/ISPF interface. The functions are Browse, Query, Load/Insert, Queue Overflow Protection, Recover, and Unload.

The functions are processed by function routines within QCF, and passed to corresponding function routines in IMS through a BMP application program interface (API). GCMD calls are used to invoke the function, and the messages, query, and status data is exchanged through GCMD, ISRT/PURGE, and GU/GN calls.



The IMS QCF function routines interface with the IMS Queue Manager and Common Queue Server (CQS) routines.

By using the standard AIB interface, errors detected are recorded with a QCF AIB return code = 00000F0, a unique AIBREASN code for each error, a TPCBSTAT code of MR, and a 6701-MRQE log record is written to the IMS online log data set (OLDS). The AIBREASN codes are printed in the reports (Browse, Query, Load, Recover, and Unload). They are documented in the DFSMRAEQ macro.

After the error is reported and logged, QCF and IMS skip to the next message, function, or terminate the BMP, depending on the error. The QCF IMS routines in IMS do not abend. To diagnose the error, the 6701-MRQE log records should be printed and analyzed. The API calls may also be traced by QCF (Trace control card), or within IMS by issuing the /TRACE SET ON PROGRAM MRQPSB. The QCF trace sends output to the QCFPRINT DD data set. The IMS trace logs type 6701 records to the OLDS.

**Related Reading:** See the *IMS Queue Control Facility for z/OS, User's Guide* for more details about the DFSMRAEQ macro and AIB error codes. For details on 6701-MRQE diagnostic records, see "Using 6701-MRQE Diagnostic Records" on page 345.

## Using SCRAPLOG Diagnostic Records

As part of your diagnosis process for problems with the Queue Control Facility/Message Requeuer, you use SCRAPLOG records. 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 are requeued.

### SCRAPLOG Records

The SCRAPLOG record consists of a 320-byte (hexadecimal 140) QCF prefix mapped by DFSMRQPR, 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. IMS Messages mapped by QLOGMSGP macro.

### Sample QCF record from scraplog data set

The following record (Figure 137 on page 344) represents a message scrapped by QCF/IMS and written to the scraplog data set. The first X'140' bytes is the QCF prefix, mapped by the DFSMRQPF macro. Offset X'88' into DFSMRQPF is the AIBREASN code = 00001084 = message is non-recoverable (in other words, INQUIRY=NORECOV on the IMS TRANSACT macro TRAN31B0).

The rest of the data is the message (offset 04 = X'03' = type 03 output message), mapped by macro QLOGMSGP.

```

5B RECORD
QCF prefix mapped by DFSMRQPF
00000000 000000 04610000 5B08C3C6 D4E2C700 08100102 01400000 00000000 00000000 00000000 *./..$QCFMSG.....*
00000020 000020 00000000 02000100 2001304F 22581647 4184032D E2E8E2F3 40404040 B6AB6C0E *.....|.D..SYS3 ..%.*
00000040 000040 26E03901 E2E8E2F3 40404040 B6AB6C0E 26E03901 00000000 00000000 00000000 *...SYS3 ..%.....*
00000060 000060 0001004C 00000000 08000002 40404040 40404040 E3D9C1D5 F3F1C2F0 D3F6F2D4 *...<..... TRAN31B0L62M*
00000080 000080 E5E2F140 00000000 00000000 00000000 000004D9 000000F0 00001084 000A000A *VS1 .....A....MR...0...D....*
000000A0 0000A0 E3D9C1D5 F3F1C2F0 40404040 40404040 00000000 00000000 D8C3C6E5 F1D9F240 *TRAN31B0 .....QCFV1R2 *
000000C0 0000C0 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 *.....*
000000E0 0000E0 TO 00000120 000120 SAME AS ABOVE
IMS message mapped by QLOGMSGP
00000140 000140 03110000 01D18194 08000002 08000002 02E40000 E2E8E2F3 40404040 B6AB6C0E *....JAM.....U..SYS3 ..%.*
00000160 000160 26E03901 E2E8E2F3 40404040 B6AB6C0E 26E03901 00000000 00000000 00000000 *...SYS3 ..%.....*
00000180 000180 00408100 C8000000 00000000 00000000 00010000 00000000 00000000 00000001 *..A.H.....*
000001A0 0001A0 F0000000 00027700 E3D9C1D5 F3F1C2F0 00000000 00000000 40404040 40404040 *.....TRAN31B0.....*
000001C0 0001C0 00108600 0264FC00 00000000 00000000 011E8700 00C2D588 8000D600 C9D4E2D5 *..F.....G..BNH..O..IMSN*
000001E0 0001E0 C5E34040 D3F6F2D4 E5E2F140 D3F6F2D4 C4C5F0F1 40404040 40404040 00000000 *ET L62MVS1 L62MDE01 ....*
00000200 000200 00000000 00027700 40404040 40404040 40404040 40404040 0C505A70 00000002 *.....&.....*
00000220 000220 E3D9C1D5 F3F1C2F0 D3F6F2D4 E5E2F140 00000000 B6AB6C0E 24746405 00000000 *TRAN31B0L62MVS1 .....%.....*
00000240 000240 00000000 00000000 00000000 00000000 00000000 00000000 00000008 E3D9C1D5 *.....TRAN*
00000260 000260 F3F1C2F0 50018046 15519555 55555555 55555555 55555555 55555555 55555555 *31B0&.....N.....*
00000280 000280 55555555 55555555 55555555 55555555 55555555 86A3A781 B0B7A415 55555555 *.....FTXA..U.....*
000002A0 0002A0 55555555 09151515 15151515 15151515 15151515 00000000 00000000 00000000 *.....*
000002C0 0002C0 00000000 00027700 40404040 40000000 00000000 00000000 00000000 00000000 *...SYS3 .....*
000002E0 0002E0 00000000 00000000 00000000 00000016 88004040 40404040 40404040 40404040 *.....H.*
000002E0 0002E0 00000000 00000000 00000000 00000016 88004040 40404040 40404040 40404040 *.....H.*
00000300 000300 4040D600 00108900 00018000 B6AB6C0E 26E49E81 00188A00 2001304F 22581647 * O...I.....%..U.A.....|....*
00000320 000320 4184032D 00000000 00000000 00688800 00000000 00000000 00000000 00000080 *.D.....U.A.....*
00000340 000340 00000000 00000000 00000000 00000000 00000000 00000000 0000000A 00000000 *.....*
00000360 000360 00000000 00000000 0000000A 000A000A E2E8E2F3 40404040 B6AB6C0E 26E03901 *.....SYS3 ..%.....*
00000380 000380 00000000 00000000 00000000 00000000 00000000 00908C00 0000000A 00000000 *.....*
000003A0 0003A0 00000003 E3D9C1D5 F3F1C2F0 F0000000 00000000 00000000 0A0A014C 40080000 00000000 *...TRAN31B0.....<.....*
000003C0 0003C0 00000000 00000000 00000000 00000000 00000000 00000000 08100000 00000000 *.....*
000003E0 0003E0 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 *.....*
00000400 000400 00000000 00000810 00000000 00000000 00000000 00000000 00000000 00000000 *.....*
00000420 000420 00000000 002D0300 E3D9C1D5 F3F1C2F0 40D6E4E3 C2D6E4D5 C440D4C5 E2E2C1C7 *.....TRAN31B0 OUTBOUND MESSAG*
00000440 000440 C540E3D6 40E3D9C1 D5F3F1C2 F0404040 40B6ABC0 0E26E5DF 01000000 00000001 *E TO TRAN31B0 ..%..V.....*
00000460 000460 E5 *V

```

Figure 137. QCF Prefix Mapped by DFSMRQPF

### Key Fields of SCRAPLOG Records and Their Offsets

Table 77 shows some key fields of the QCF records and their offsets.

Table 77. Key Fields in DFSMRQPF

| Offset | Label    | Length | Value    | Description                                                          |
|--------|----------|--------|----------|----------------------------------------------------------------------|
| 04     | MSGMRQID | 08     | \$QCFMSG | Prefix ID (First char = 5B which causes DFSERA30 to print as 5B rec) |
| 74     | MRPREDST | 08     | TRAN31B0 | Destination name                                                     |
| 94     | MRPRETRN | 04     | 000000F0 | AIBRETRN code, always this value for QCF errors                      |
| 98     | MRPREASN | 04     | 00001084 | AIBREASN code = message non recoverable                              |

Table 78. Key Fields in Message (offset 0140 = offset 00 into message)

| Offset | Label    | Length | Value    | Description                                                         |
|--------|----------|--------|----------|---------------------------------------------------------------------|
| 0140   | MSGLRLL  | 02     | 0361     | Length of message                                                   |
| 0144   | MSGLCODE | 01     | 01       | Log code, 01 = input message, 03 = output message                   |
| 0150   | MSGPRFLL | 02     |          | Length of total message prefix (user segments start at this offset) |
| 01A8   | MSGODSTN | 08     | TRAN31B0 | Message destination name                                            |

## Sample JCL for Printing SCRAPLOG Records

Figure 138 shows sample JCL that you can use to print SCRAPLOG records. You use these SCRAPLOG records to help diagnose problems with the Queue Control Facility (QCF)/Message Requeuer (MRQ).

```
//SCRAPPRT JOB
/* PRINT IQCSELCT SCRAPSEL
//JOB LIB DD DISP=SHR,DSN=IMS610.RESLIB
//SELECT EXEC PGM=DFSERA10,REGION=512K
//SYSPRINT DD SYSOUT=A
//SYSUT1 DD DSN=QCF.SCRAPSEL,DISP=SHR
//SYSIN DD *
CONTROL CNTL
OPTION PRINT E=DFSERA30
END
/*
//CANCEL EXEC PGM=DFSERA10,COND=EVEN,REGION=256K
/* PRINT IQCCANCL SCRAPCAN
//SYSPRINT DD SYSOUT=A
//SYSUT1 DD DSN=QCF.SCRAPCAN,DISP=SHR
//SYSIN DD *
CONTROL CNTL
OPTION PRINT E=DFSERA30
END
//INSERT EXEC PGM=DFSERA10,COND=EVEN,REGION=256K
/* PRINT IQCINSRT SCRAPLOG
//SYSPRINT DD SYSOUT=A
//SYSUT1 DD DSN=MRQ.SCRAPLOG,DISP=SHR
//SYSIN DD *
CONTROL CNTL
OPTION PRINT E=DFSERA30
END
/*
```

Figure 138. Sample JCL for Printing SCRAPLOG Records

You need to use your SCRAPLOG records in combination with 6701-MRQE records to effectively diagnose QCF problems.

## Using 6701-MRQE Diagnostic Records

This section provides the following details about 6701-MRQE diagnostic records:

- 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 QCF 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 QCF BMP (IQCINSRT) 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 the *IMS Queue Control Facility for z/OS User's Guide (SC26-9685)* and macro DFSMRAEQ.

When the IQCINSRT 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 Queue Control Facility for z/OS User's Guide (SC26-9685)* for an explanation of these reports.

### Sample JCL for Printing the 6701-MRQE Diagnostic Records

Figure 139 shows the sample JCL for printing 6701-MRQE records.

```
//LOGPRNT JOB
//JOB LIB 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 0=5,V=6701,L=2,C=M,E=DFSERA30
OPTION PRINT 0=9,V=MRQE,L=4,T=C,C=E,E=DFSERA30
END
/*
```

Figure 139. 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 control blocks and data areas shown in Table 79 which are logged if they are available at the time of the error.

Table 79. Control Blocks and Data Areas Logged at Time of Error for 6701-MRQE Records

| Block    | Description                                                                  | Mapping Macro                                                                     |
|----------|------------------------------------------------------------------------------|-----------------------------------------------------------------------------------|
| AIB      | Application Interface Block                                                  | DFSAIB                                                                            |
|          | AIBRETRN, AIBREASN codes                                                     | DFSMRAEQ                                                                          |
| CMDMSGP  | Command Call Buffer for Browse, Load, QSN, Query, Recover, or Unload command | DFSMRQCW (OCO)                                                                    |
| DFSSQQR  | Query buffer                                                                 | CQSQRQT                                                                           |
| I/O AREA | Input/Output Area                                                            | QLOGMSGP                                                                          |
| MRQCMDWK | Command Call Buffer for Browse, Load, QSN, Query, Recover, or Unload command | DFSMRQCW (OCO)                                                                    |
| MRQPREFX | QCF Prefix buffer                                                            | DFSMRQPF                                                                          |
| MRQWORK  | MRQ/QCF Work Area                                                            | Mapping macro                                                                     |
| MRSELROW | Include/Exclude work area                                                    | DFSMRQCT (OCO)                                                                    |
| MRSELWK  | Select work buffer                                                           | DFSMRQSW (OCO)                                                                    |
| PCB      | Program Control Block                                                        | IDLI TPCBBASE=0,CALLER=IMS                                                        |
| PST/EOB  | Partition Specification Table                                                | IDLI PSTBASE=0                                                                    |
| PSTDCA   | DL/I Call Parameter Area                                                     | No DSECT                                                                          |
| QMBA     | Queue Manager Buffer Area                                                    | DFSQMGR FUNC=QDSECT                                                               |
| QSAPWKAD | Queue Manager Work Area                                                      | QSAPWKAD                                                                          |
| QTPDST   | Queue Manager Destination Block                                              | ICLI CNTBASE=0, or IAPS SMBBASE=0 (CNT/LNB or SMB) DSECT for QAB/TIB not provided |
| REG14-12 | Registers 14 thru 12                                                         | No DSECT                                                                          |
| WORKMSG  | Work Message Buffer                                                          | QLOGMSGP                                                                          |

## 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 IQCINSRT 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 *IMS Queue Control Facility for z/OS User's Guide (SC26-9685)*.

## 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 QCF 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 QCF 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 9: 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 “IMS Transaction Trace” on page 358 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 is 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 Queue Control Facility/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 = 45) is set to indicate that this message was requeued by the Queue Control Facility/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/MSD systems when the message is sent across the MSC link.)

Figure 140 shows an input transaction to TRANCODE = TRAN31V0 from LTERM = IMSUS02 that was requeued by QCF.

```

01 RECORD
00000000 000000 01EE0000 01C18110 08000055 08000055 01CE1000 E2E8E2F3 40404040 B6AB6CBC *....AA.....SYS3 ..%.*
00000020 000020 C4E84B83 C9D4E2F1 40404040 B7BD992F E30E2241 80000100 00000000 00000000 *DY.CIMS1 ..R.T.....*
00000040 000040 00408100 C8400000 C4E3E2D3 E4F2F0F2 00020000 00000000 00000000 00000001 * .A.H ..DTSLU20.....*
00000060 000060 C9D4E2E4 E2F0F240 E3D9C1D5 F3F1E5F0 00000000 00000000 C4C6E2D4 D6F24040 *IMSUS02 TRAN31V0.....DFSM02 *
00000080 000080 00108600 014E7C00 00000000 00000000 00168800 C9D4E2E4 E2F0F240 40404040 *.F..+@.....H.IMSUS02 *
000000A0 0000A0 40404040 E4000018 89000000 0000B6AB 6CBCC4EA CD030000 00000000 00000018 * U...I.....%D.....*
000000C0 0000C0 8A002001 304F2301 19573676 032D0000 80000000 00000068 8B000000 00000000 *.....|.....*
000000E0 0000E0 00000000 00000000 00800000 00000000 00000000 00000000 00000000 00000000 *.....*
00000100 000100 00000000 000A0000 00000000 00000000 00000000 000A0024 000AE2E8 E2F34040 *.....SYS3 *
00000120 000120 4040B6AB 6CBCC4E8 4B830000 00000000 00000000 00000000 00000000 00000090 * ..%.DY.C.....*
00000140 000140 8C000000 000A0000 00000000 0003E3D9 C1D5F3F1 E5F0C9D4 E2E4E2F0 F240240A *.....TRAN31V0IMSUS02 ..*
00000160 000160 014C0008 00000000 00000000 00000000 00000000 00000000 00000000 00000000 *.<.....*
00000180 000180 00000810 00000000 00000000 00000000 00000000 01E3D9C1 D5F3F1E5 F0404040 *.....TRAN31V0 *
000001A0 0001A0 40404040 00000000 00000000 00000000 08100000 00000000 00000000 00000000 * ..*
000001C0 0001C0 00000000 00000000 00000000 00000010 0301E3D9 C1D5F3F1 E5F040C8 C94BB7BD *.....TRAN31V0 HI...*
000001E0 0001E0 992FE30F C9610000 00000000 021B *R.T.I/..... *

```

Figure 140. Sample Log Record Showing Successfully Requeued Message

## Diagnosing Message Routing Problems

In version 7, the user message routing exits DFSCMTR0, DFSNPRT0, DFSCMLR0/DFSCMLR1, and DFSCMPR0 were used to route or control message processing in a Transaction Manager (TM) or TM/Multiple Systems Coupling (MSC) environment. For releases after version 7, these user exits were consolidated into one single user exit, DFSMSCE0. 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.

The following topics provide additional information:

- “DFS070 UNABLE TO ROUTE MESSAGE RSN=xyxy” on page 349
- “Using the DFSMSCE0 Routing Exit Trace” on page 354
- “Using the Transaction/Program Trace to Diagnose Routing Errors” on page 356
- “Using the DC LINE/NODE/LINK TRACE to Diagnose Routing Problems” on page 357
- “Using 01/03 Log Record Trace” on page 358

## DFS070 UNABLE TO ROUTE MESSAGE RSN=xyyy

Message DFS070 is issued when any one of the following conditions occur:

- IMS attempts to enqueue a message.
- These TM/MSC exits attempt to reroute a message:
  - DFSMSCE0–Message Routing.
  - DFSMSTR0–Terminal 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 80. DFS070 Module Identifier Table*

| LABEL  | MODULE IDENTIFIER<br>(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 (DFSIIINB0)       |

Table 81 on page 350 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 81. DFS070 Reason (RSN) Codes Table

| LABEL    | REASON CODE<br>DEC/HEX | DESCRIPTION                                                                                                                                                                                                                                                                                                                                                                            |
|----------|------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| PFXUPRER | 02/02                  | User requested 2 user prefix segments (code 8E).<br><br>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).<br><br>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.<br><br>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.<br><br>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.<br><br>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.<br><br>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.<br><br>Programmer response: Failure to get storage for the MSEB block through the DFSBCB macro.                                                                                                                                                                                                                                                        |
| LRBADSID | 09/09                  | Bad SYSID detected.<br><br>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.                                                                                                                                                                                 |
| IPFX     | 10/0A                  | Queue Manager insert prefix error.<br><br>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 or the workload router segment, or both. 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.                                                                                                                                                                                                                                                                                                                                       |



Table 81. DFS070 Reason (RSN) Codes Table (continued)

|          |       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|----------|-------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| MSCEFL1E | 15/0F | <p>DFSMSCEC user exit routing flag is in error.</p> <p>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.</p>                                                                                                                                                                 |
| USRXIFER | 16/10 | <p>DFSUSRX interface error.</p> <p>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:</p> <ol style="list-style-type: none"> <li>1. 04 the user exit routine specified has not been defined (the address in UXDT is zero)</li> <li>2. 2) Unable to get an interface block using the DFSBCB macro. DFSBCB return code is in field, MSCEBSSRC in the DFSMSCEB block.</li> </ol> |
| IONAMCHG | 18/12 | <p>User exit changed the destination name of the I/O PCB message.</p> <p>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.</p>                                                                                                                                                              |
| IOROUTE  | 19/13 | <p>User exit requested reroute I/O PCB message.</p> <p>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.</p> <p>Refer to the user exit parameter in the 6701-MSCE record to determine which command was requested.</p>                                                                                                                                                                                |
| CMDINV   | 20/14 | <p>User exit changed the destination name to a command (such as: /CMDVERB).</p> <p>Programmer response: The user exit, DFSMSCE0 changed the destination name to a command.</p> <p>Refer to the user exit parameter in the 6701-MSCE record to determine which command was requested.</p>                                                                                                                                                                                                                                             |
| SQGINV   | 21/15 | <p>User Link receive exit override MSNAME in segment because destination is not an MSNAME.</p> <p>Programmer response: User exit, DFSMSCE0 in a shared queues group link receive exit failed due to the destination not being an MSNAME.</p>                                                                                                                                                                                                                                                                                         |
| REGFAIL  | 22/16 | <p>Local shared queue registration (DFSSQIF FUNC=INFRM) failed for the transaction when the user exit requested MSLR2LSQ=1 or MSTR2LSQ=1.</p>                                                                                                                                                                                                                                                                                                                                                                                        |
| NOTRANCD | 23/17 | <p>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).</p>                                                                                                                                                                                                                                                                                                     |
| DSIDINV  | 24/18 | <p>The Terminal, Link Receive or the Program Routing exit returned an invalid destination SYSID (for example: either field, MSTRDSID, MSLRDSID, or MSPRDSID is invalid).</p>                                                                                                                                                                                                                                                                                                                                                         |

Table 81. DFS070 Reason (RSN) Codes Table (continued)

|          |       |                                                                                                                                                                                                                                                                                                                                                                                                                      |
|----------|-------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 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: <ul style="list-style-type: none"> <li>• MSC is not available.</li> <li>• Or the source SYSID is not valid because the application program has not issued a get unique (GU).</li> <li>• The application program is a non-message driven BMP.</li> </ul> |
| NDR2INV  | 29/1D | The Program Routing exit requested a direct routing message be overridden by setting MSPR2NDR=1 however either: <ul style="list-style-type: none"> <li>• MSC is not available.</li> <li>• This is not a direct routed message with a MSNAME destination.</li> <li>• The overriding name in the front of the I/O area is not valid.</li> </ul>                                                                        |
| 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 (DFSFEJ0). 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.<br><br>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.<br><br>Refer to the sample version of the provided user exit DFSMSCE0's use of DFSMSCSV for more information.                                                             |
| INBIDLNG | 35/23 | Character string 'VECTOR' not present.<br><br>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.<br><br>Refer to the sample version of the provided user exit DFSMSCE0's use of DFSMSCSV for more information.                                    |

Table 81. DFS070 Reason (RSN) Codes Table (continued)

|          |       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|----------|-------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| INBNVECT | 35/23 | <p>Character string 'VECTOR' not present.</p> <p>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.</p> <p>Refer to the sample version of the user exit DFSMSCE0's use of DFSMSCSV for more information.</p>                                                                                                                                                                                                                                                                                                |
| PFXUINVA | 36/24 | <p>Upon return from the user exit IMS detected that the user prefix at MSCEUPR is invalid.</p> <p>Possible causes are:</p> <ul style="list-style-type: none"> <li>• Length not in range of 5 to 512 bytes.</li> <li>• Address of prefix is invalid. Must be address obtained by IMS or within HIOP pool.</li> <li>• Length has been changed (MSCEBUPRL).</li> <li>• Address of user exit prefix has changed (MSCEBUPR).</li> <li>• Prefix code not 8E.</li> </ul> <p>The programmer may need to turn on the DFSMSCE0 trace to trace the fields, MSCEBUPR and MSCEBUPRL within the DFSMSCEB block.</p>                 |
| PFXIINVA | 37/25 | <p>Upon return from the user exit, IMS detected the Workload Router prefix at MSCEIPR is invalid.</p> <p>Programmer response:</p> <ul style="list-style-type: none"> <li>• Length not in range of 5 to 512 bytes.</li> <li>• Address of prefix is invalid. Must be address obtained by IMS or within HIOP pool.</li> <li>• Length has been changed (MSCEBIPRL).</li> <li>• Address of workload router prefix has changed (MSCEBIPR).</li> <li>• Prefix code is not 8F.</li> </ul> <p>The programmer may need to turn on the DFSMSCE0 trace to trace the fields, MSCEBIPR and MSCEBIPRL within the DFSMSCEB block.</p> |
| EXIOVLAY | 38/26 | <p>User exit overlaid the 512 byte user work area buffer.</p> <p>Programmer response: The user exit, DFSMSCE0 appears to have overlaid the 512 byte workarea.</p> <p>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.</p> <p>Refer to the user exit DFSMSCEB in the 6701-MSCE record to help determine the overlay.</p>                                                                                                                                                                              |
| EXBOVLAY | 39/27 | <p>User exit overlaid the MSEB BCB block name (Overlay Check).</p> <p>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.</p> <p>Refer to the user exit parameter in the 6701-MSCE record to help determine the overlay.</p>                                                                                     |

Table 81. DFS070 Reason (RSN) Codes Table (continued)

|          |       |                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|----------|-------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| EXPOVLAY | 40/28 | User exit overlaid the parameter list (Overlay Check).<br><br>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.<br><br>Refer to the user exit parameter in the 6701-MSCE record to help determine the overlay. |
|----------|-------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

Codes 41 thru 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/MS 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=xyxy 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/MSD message routing exit, DFSMSCE0.
  - The Terminal Routing exit, DFSMSTR0.
- 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
```

Table 82 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 82. DFS081 Module Identifier Table*

| LABEL | MODULE IDENTIFIER<br>(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 83. 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|TR0T|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 exit 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 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 “IMS Transaction Trace” on page 358 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 | MSEDIRR0(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 are logged:

6701-A01 - DC analyzer (DFSIC100) 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

---

## IMS Transaction Trace

The IMS Transaction 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:

|           |                                            |
|-----------|--------------------------------------------|
| <b>04</b> | ABTERM IN PROGRESS                         |
| <b>08</b> | SYNC POINT PHASE 1                         |
| <b>0C</b> | SYNC POINT PHASE 2                         |
| <b>10</b> | PURGE TP PCBS                              |
| <b>14</b> | PHASE 1 SYNC POINT ENQ OUTPUT TO TEMP DEST |
| <b>18</b> | ROLB CALL                                  |
| <b>1C</b> | INVALID ABENDU0820                         |
| <b>20</b> | ABORT                                      |

Field PSTFUNCT in the PST contains the following calls:

|           |      |
|-----------|------|
| <b>01</b> | GU   |
| <b>03</b> | GN   |
| <b>41</b> | ISRT |
| <b>50</b> | SETO |
| <b>67</b> | INQY |
| <b>83</b> | CHNG |
| <b>85</b> | CHKP |
| <b>87</b> | CMD  |
| <b>88</b> | GCMD |
| <b>89</b> | ROLB |
| <b>8A</b> | ROLS |
| <b>8C</b> | SETS |
| <b>8F</b> | AUTH |
| <b>90</b> | PURG |

I Figure 141 on page 360 is an example of a IMS Transaction trace.

### **Example of IMS Transaction 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 *..L.|..... *
0271B0E4 000060 40404040 40404040 * * *
I/O AREA
02825000 000000 00340000 C3E4E2E3 D6D4C5D9 40D9C5D8 E4C5E2E3 E240C9D5 C6D6D9D4 C1E3C9D6 *....CUSTOMER REQUESTS INFORMATIO*
02825020 000020 40404040 41416000 0700A704 FFFFFFFF 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 00000000 00000000 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 00000000 0275DC40 00000000 00000000 00000000 *.....B&;..... *
0271B160 000100 00000000 028B8020 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 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 00000000 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 000000C8 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 00E15E6 00196FF2 00000000 00000000 *.....W..?..... *
0271B444 0003E4 00000000 00000000 00000000 00000000 00000000 00000000 028B8000 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 0000C404 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 *.....<..DWTOR *
0271B5E4 000584 000056E0 00000000 00196D3D 0280E524 00000000 00000001 0000FFFF 827BEC70 *.....V.....B#.. *
0271B604 0005A4 80000000 0271B648 82988A1E 82978630 00000000 0271B060 827BEC70 028B8068 *.....B...BPF.....-B#.. *
0271B624 0005C4 0271B060 0280E610 02825840 8298891C 02825000 026EB048 00000064 00C53D20 *...W.B.B.I..B&;>.....E.. *
0271B644 0005E4 029881E8 00000000 0271B600 0271B690 82978774 0297C2FE 00000000 02707540 *.AY.....BPG..PB..... *
0271B664 000604 02825840 0271B084 02825000 82825850 02825840 8298891C 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 8298891C *.....D.B&;.....QB.I.. *

```

Figure 141. IMS Transaction 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 8298891C 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 8298891C *.E...#...#.....VMB.I...*
0271B7E4 000784 0272E078 0271B060 00C19000 00C53D20 02998C2C 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 0271B060 0271B840 0271B8D0 82957237 829A19C8 00000000 0275F260 *.R.....E...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.....2-*
0271B924 0008C4 829A2579 829554F8 00C53D20 00C2A238 C3D5E340 000001FF 02C5A758 02C5A758 *B...BN..B.E...BS.CNT .....EX..EX...
0271B944 0008E4 00C2A238 00000000 027BEC70 0271B060 0275F260 00C53D20 029A248E 00000000 *.BS.....#.....-2-.E.....*
0271B964 000904 0271B918 0271B9A8 80115668 829CA578 000053E8 02766910 00000000 02B54000 *.....Y...;B.V...Y.....*
0271B984 000924 000053E8 02766900 00000000 02766910 0271B0EC 0271B060 02B56260 00C53D20 *...Y.....-2-.E.....*
0271B9A4 000944 00115588 00000000 0271B960 0271B9F0 80115668 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 80115668 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
0271BA44 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 00000000 00000000 00000000 00000000 00000000 *.....*
0271BC24 000BC4 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 *.....*
0271BC44 000BE4 00000000

```

Figure 141. IMS Transaction 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 either the IMS IPCS panel interface or the manual procedure (both shown below) to help you determine if this is the case. As you proceed through the steps, write down the information you gather.

### IMS IPCS Dump Formatter Panel Interface

The IMS IPCS dump formatter provides a panel driven interface to perform analysis. Using this interface, choose the “RECANY” selection in the EDA/TM (Enhanced Dump Analysis/Transaction Manager) option, shown in Figure 142 on page 362, to create an output that contains the RECANY information.

```

----- RECEIVE ANY BUFFERS FORMATTING OPTIONS -- Row 1 to 3 of 3
COMMAND ==> Scroll ==> PAGE

The Receive Any Buffer contains input data that IMS receives from
VTAM terminals.

S = SELECT Select choice plus required ARGument
M = SELECT (minimum data) and hit enter to process. Use UP/DOWN
X = SELECT (maximum data) to scroll.

Cmd Option Type ARG Argument description
v-----vvvvvvvv-----
_ RECANY TYPE All Receive Any Buffers (leave ARG blank)
_ RECANY FILTER Receive Any Buffer filtering option
_ RECANY FILTER ALLOCATE Filter for allocated buffers
***** Bottom of data *****

```

Figure 142. IPCS Dump Formatter EDA/TM Option

### Manual procedure

1. Find the address of the first RECANY buffer.
  - SCD+X'91C' = pointer to the first RECANY buffer (SCDRECPT)
  - SCD+X'920' = size of each RECANY buffer (SCDRCSIZ)
  - SCD+X'922' = 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 143. The registers in this save set are the registers saved on entry to each module.

```
***SAVE AREA SET***  
  
    EP DFSIC100  
    SA 22FE930  
  
    EP DFSCFE10  
    SA 22E930  
  
    EP DFSCFEP0  
    SA 22E990  
  
    EP DFSCIO00  
    SA 229490  
  
    EP DFSQMGR0  
    SA 22D990  
  
    EP DFSA0S80  
    SA 320548
```

Figure 143. 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 (DFSIC100) 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 z/OS console. These messages 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 150 lists all log records and illustrates the format of the X'67' log record.

---

## Message Format Service Normal BTAM Path

The diagrams in Figure 144 on page 365 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 330.

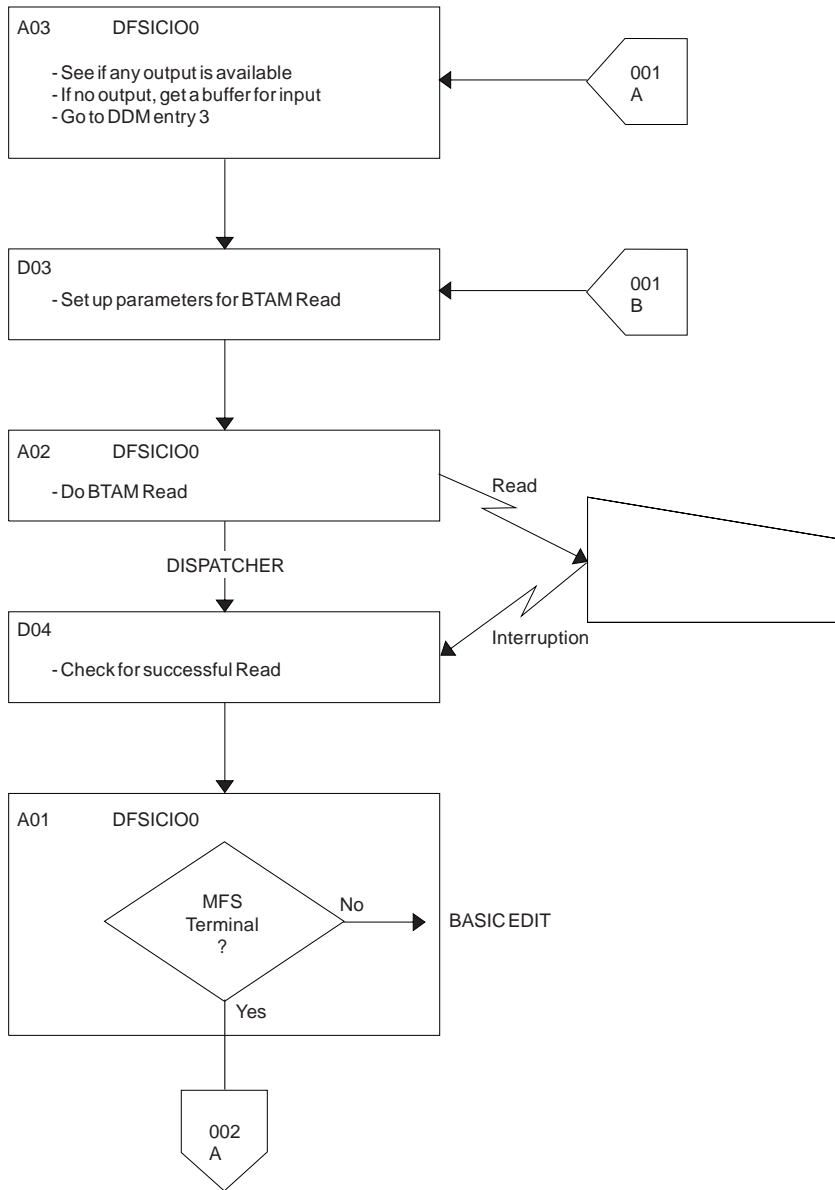


Figure 144. Message Format Service (MFS) Normal BTAM Path (Part 1 of 5)

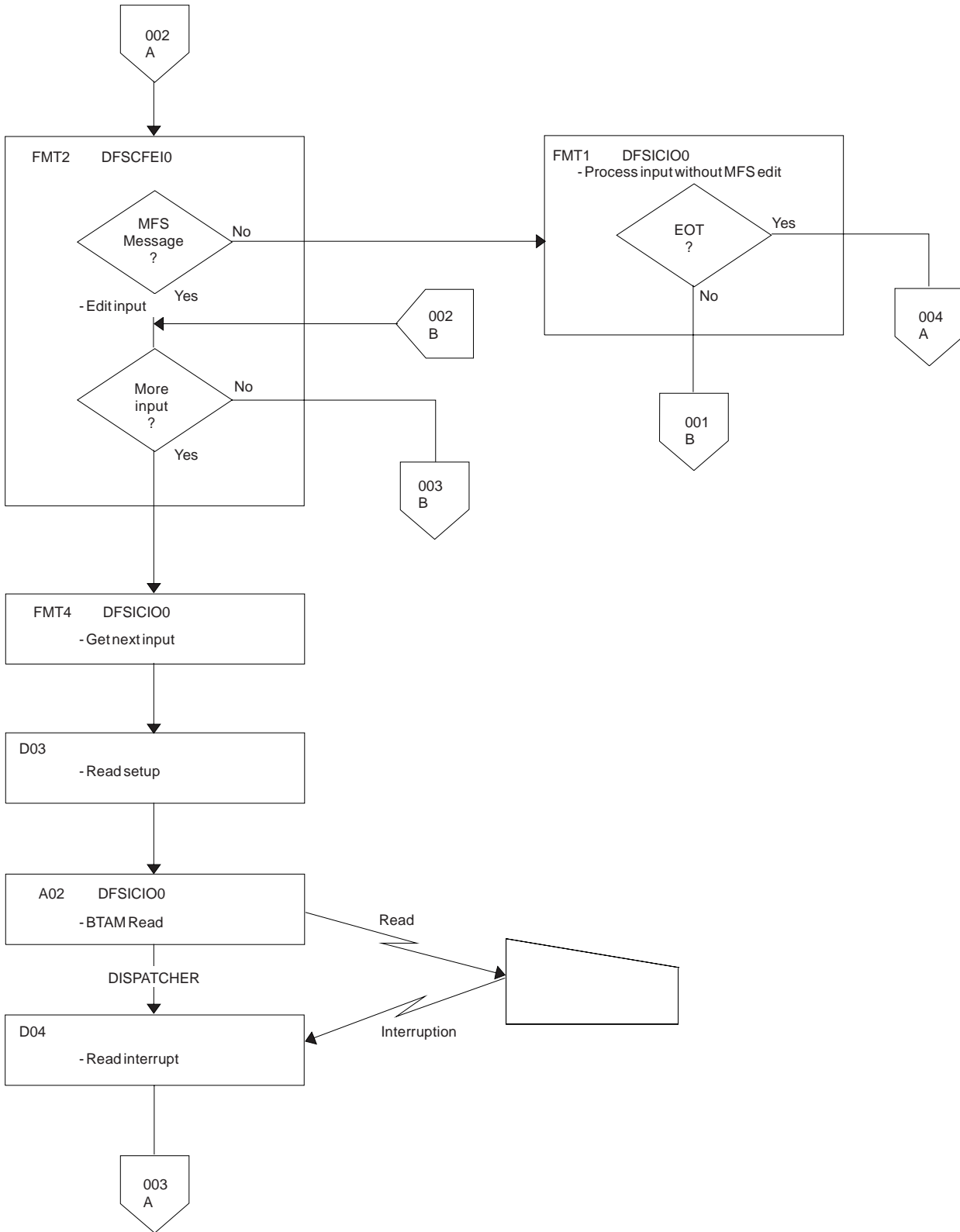


Figure 144. Message Format Service (MFS) Normal BTAM Path (Part 2 of 5)



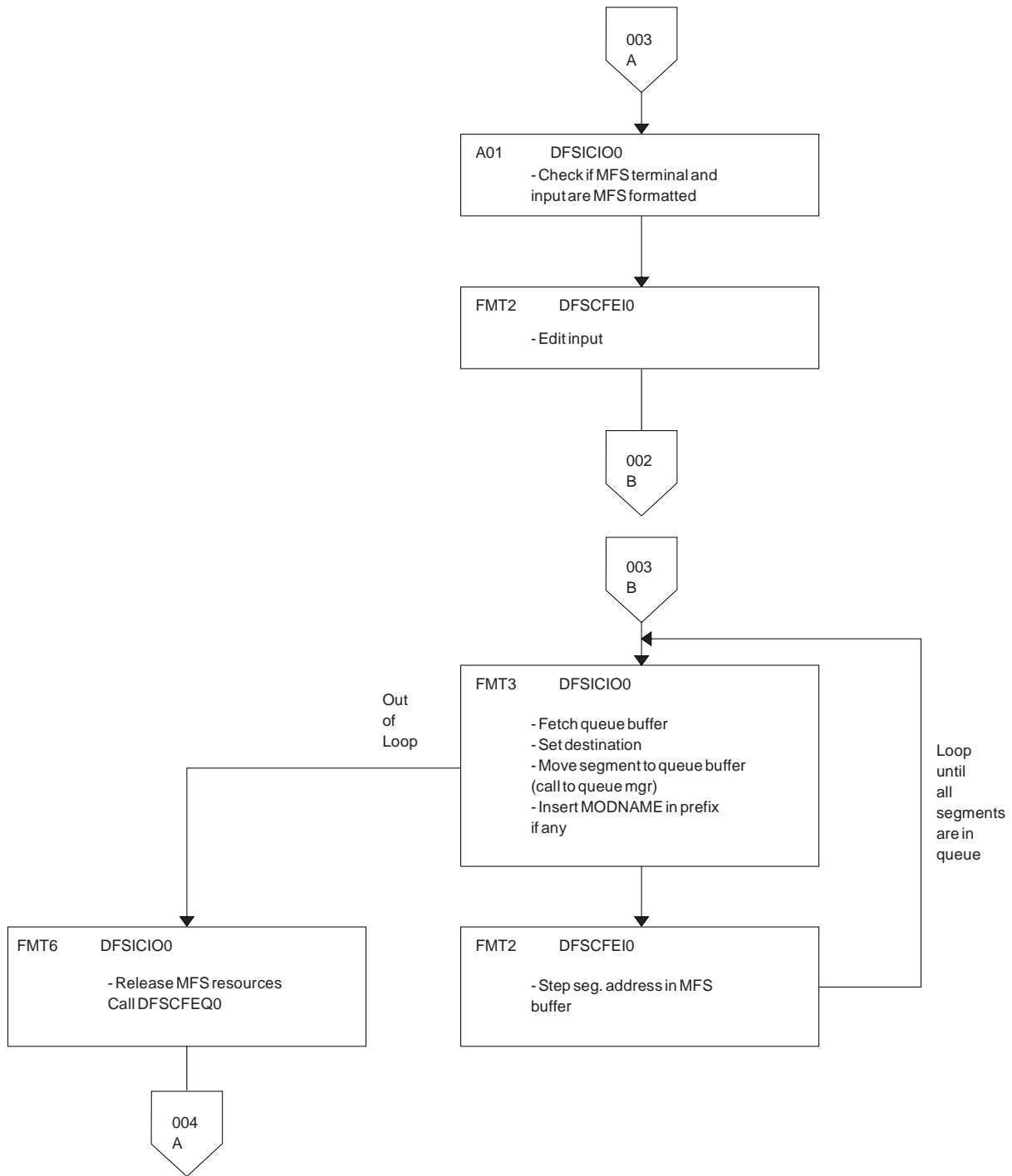


Figure 144. Message Format Service (MFS) Normal BTAM Path (Part 3 of 5)

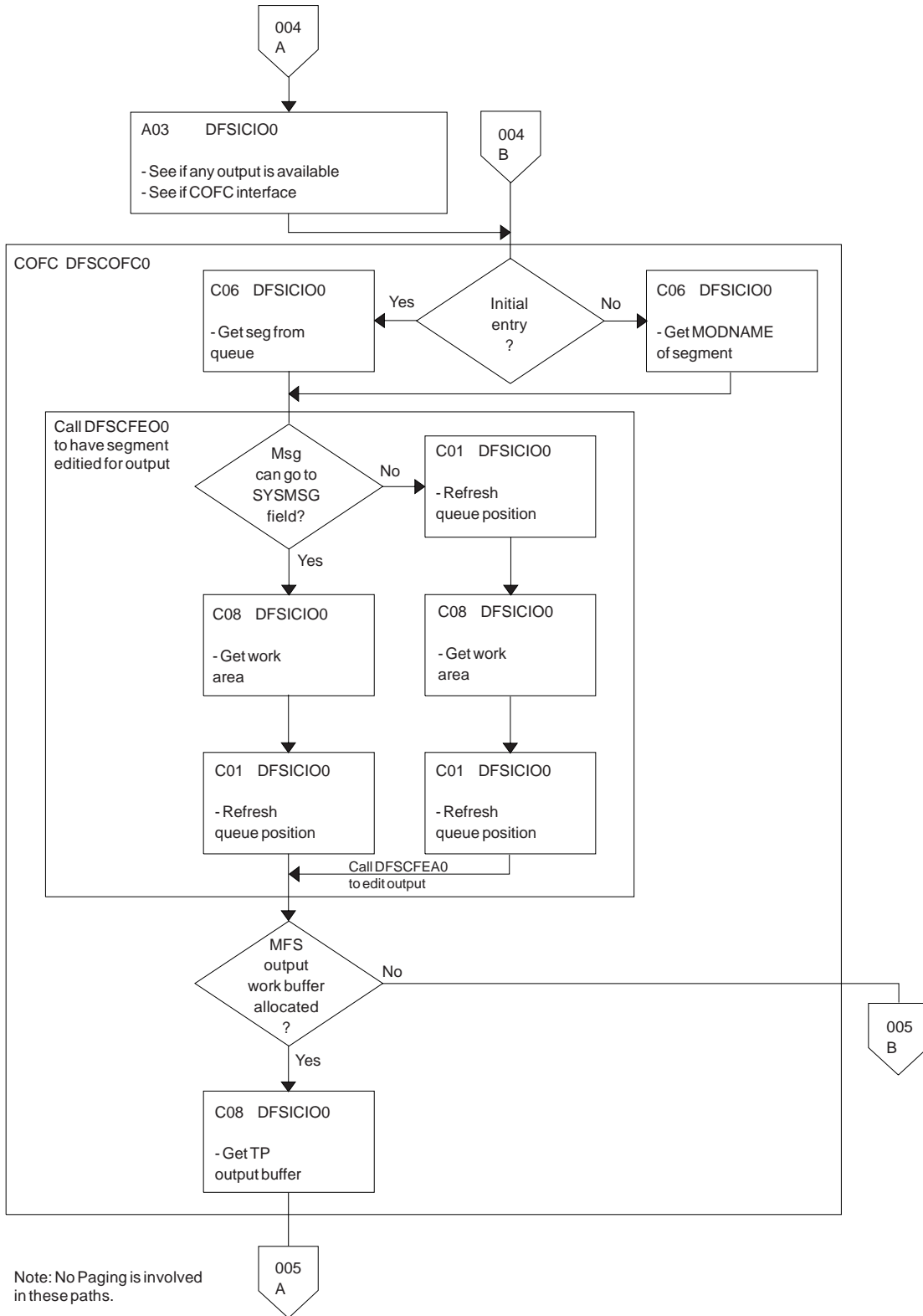


Figure 144. Message Format Service (MFS) Normal BTAM Path (Part 4 of 5)

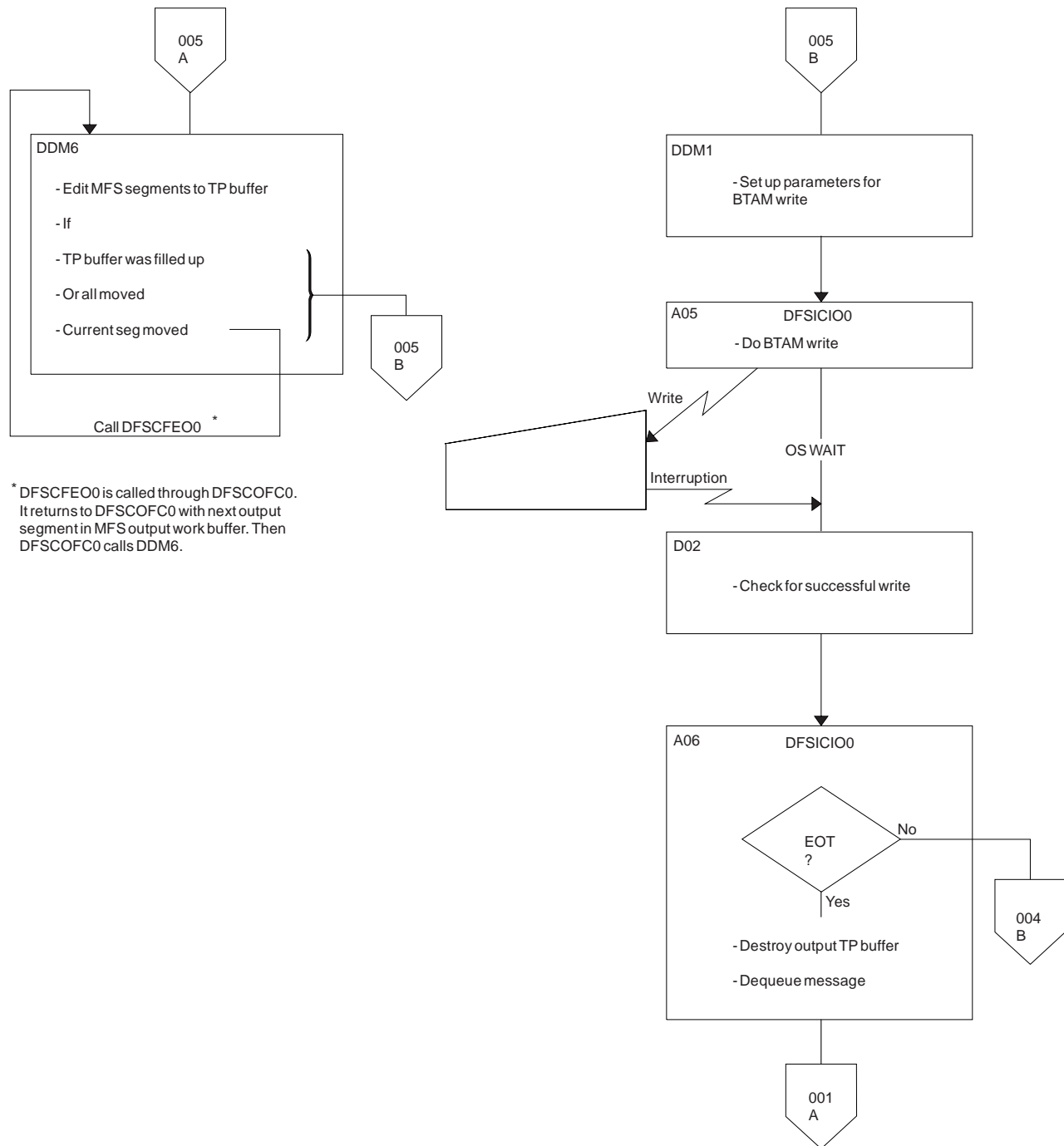


Figure 144. Message Format Service (MFS) Normal BTAM Path (Part 5 of 5)

## Diagnosing Message Format Service Problems

For information about starting, stopping, and printing the DC trace, see “DC Trace” on page 327.

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 DFSCFEIO.

- 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 145 shows the format.

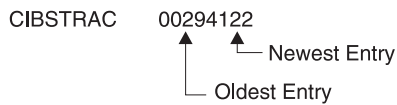


Figure 145. Example of CIBSTRAC Trace

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 146 shows the format.

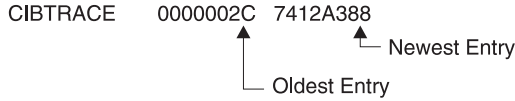


Figure 146. Example of CIBTRACE Trace

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.                                               |

---

**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 calls to VTAM exit routines.

The following topics provide additional information:

- | • “Location Codes for DFSCNXA0 Error Messages”
- | • “Qualifier Codes” on page 379

## | Location Codes for DFSCNXA0 Error Messages

| Message DFS3672I contains the location codes listed in Table 84. The message also identifies the exit routine 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=*ddd*

| 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 84. Location Codes for DFSCNXA0 Error Messages*

| Location Code<br>(Dec) | Location Code<br>(Hex) | Msg#<br>(DFS) | Exit | Explanation                                                       |
|------------------------|------------------------|---------------|------|-------------------------------------------------------------------|
| 19                     | 13                     | 3862          | LOG  | Non-master terminal initiating a session on the alternate system. |
| 20                     | 14                     | 3100          | LOG  | Node in FP input mode.                                            |
| 21                     | 15                     | 3645          | LOG  | Generic Resource name used but VGR for ISC was disabled.          |
| 22                     | 16                     | 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.                          |
| 7                      | 7                      | N/A           | NSXT | Polarity mismatch on MSC link.                                    |
| 8                      | 8                      | N/A           | NSXT | Polarity mismatch on MSC link.                                    |
| 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.                                            |

Table 84. Location Codes for DFSCNXA0 Error Messages (continued)

| Location Code (Dec) | Location Code (Hex) | Msg# (DFS) | Exit | Explanation                                                                      |
|---------------------|---------------------|------------|------|----------------------------------------------------------------------------------|
| 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 | VTCTB not found.                                                                 |
| 2                   | 2                   | N/A        | RELQ | Terminal defined with NORELQ option.                                             |
| 3                   | 3                   | N/A        | RELQ | No CID in nonparallel-session VTCTB.                                             |
| 4                   | 4                   | N/A        | RELQ | No CID in any parallel-session VTCTBs.                                           |
| 1                   | 1                   | 1915       | SCIP | No pointer to RPL.                                                               |
| 2                   | 2                   | 1917       | SCIP | Node not found.                                                                  |
| 3                   | 3                   | 3862       | SCIP | VTCTB not found (XRF Alt.).                                                      |
| 4                   | 4                   | 3862       | SCIP | Invalid temporary VTCTB (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.                                                                 |
| 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 85 deal with ISC processing—either as a result of LOGON or SCIP exits being driven. This is reflected in the DFS3672 message through the appending of 'I' to the exit type.

Table 85. 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  | First structured field not 0. |
| 6                   | 6                   | 2066       | ISC  | User field length = 0.        |



Table 85. Codes Related to ISC Processing (continued)

| Location Code (Dec) | Location Code (Hex) | Msg# (DFS) | Exit | Explanation                                |
|---------------------|---------------------|------------|------|--------------------------------------------|
| 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  | Second 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).                   |
| 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  | Second logon entry for same session.       |

The codes in Table 86 may occur during ISC BINDRACE processing.

Table 86. 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 87 deal with MSC errors.

Table 87. 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 88 deal with MSC SCIP errors.

Table 88. Codes Related to MSC SCIP Errors

| Location Code (Dec) | Location Code (Hex) | Msg# (DFS) | Exit | Explanation          |
|---------------------|---------------------|------------|------|----------------------|
| 71                  | 47                  | N/A        | MSC  | CID already present. |
| 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 89 deal with dynamic logon errors.

Table 89. 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 VTCTB trying to logon to alternate system. |
| 83                  | 53                  | 2037       | LOG  | /STA DC not done.                                      |
| 84                  | 54                  | 2104       | LOG  | Invalid temporary VTCTB exists.                        |
| 85                  | 55                  | 3862       | LOG  | Invalid temporary VTCTB 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 MTO logging not on.                  |
| 95                  | 5F                  | 3111       | LOG  | Node stopped.                                          |
| 96                  | 60                  | 3101       | LOG  | Node logging off.                                      |

Table 89. Codes Related to Dynamic Logon Errors (continued)

| Location Code (Dec) | Location Code (Hex) | Msg# (DFS) | Exit | Explanation                     |
|---------------------|---------------------|------------|------|---------------------------------|
| 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 90 deal with existing ISC session errors.

Table 90. 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 Routine Processing

The location codes in Table 91 deal with user-logon-exit routine processing.

Table 91. Codes Related to User-Logon-Exit Routine 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 or ASOT value from user logon exit routine         |
| 126                 | 7E                  | 3645       | N/A  | User logon exit routine erased 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 92 deal with logon-related errors.

Table 92. 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 97 on page 379. |
| 143                 | 8F                  | 3646       | N/A  | Inconsistent attributes—see Table 97 on page 379. |
| 144                 | 90                  | 3645       | N/A  | Could not get SOPB storage.                       |

Table 92. Codes Related to Logon Errors (continued)

| Location Code (Dec) | Location Code (Hex) | Msg# (DFS) | Exit | Explanation                                                                                          |
|---------------------|---------------------|------------|------|------------------------------------------------------------------------------------------------------|
| 145                 | 91                  | 3645       | N/A  | Parsing of userdata failed. See Table 95 on page 379.                                                |
| 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  | The 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 routine returned invalid descriptor.                                                 |
| 164                 | A4                  | 3644       | N/A  | Could not get SOPB storage.                                                                          |
| 165                 | A5                  | 3651       | N/A  | No default descriptor found.                                                                         |

### Codes Related to Logon Descriptor Processing

The location codes in Table 93 deal with logon descriptor processing.

Table 93. 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 LU 1 or NTO device type.                                                                       |

### Codes Related to Logging-on Device Characteristics

The location codes in Table 94 deal with logging-on device characteristics and their compatibility with the logon descriptor being requested.

Table 94. Codes Related to Logging-on Device Characteristics

| Location Code (Dec) | Location Code (Hex) | Msg# (DFS) | Exit | Explanation                                                           |
|---------------------|---------------------|------------|------|-----------------------------------------------------------------------|
| 191                 | BF                  | 3646       | LOG  | Invalid SLU 1 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 or 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 95 deal with ETO-related parsing errors (associated with a DFS3645I message).

Table 95. 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 96 deal with VTCB-creation errors (associated with a DFS3644 message).

Table 96. 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.                   |
| 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 97 deal with screen-attribute errors (associated with a DFS3646I message).

Table 97. 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. Run the MFS DCT (DFSUTB00) utility.                                                                                      |
| 2                   | 2                   | N/A        | N/A  | No match for 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 X'7F' is specified, then a valid screen size must also be specified. |

## IDC0 Trace Table Entries

### Error Messages Issued by DFSCNXA0

Table 98 lists codes that identify error messages issued by DFSCNXA0. The code is placed in the MsgID field of an IDC0 trace entry.

Table 98. Codes that Identify Error Messages Issued by DFSCNXA0

| Code (Dec) | Code (Hex) | Msg# (DFS) |
|------------|------------|------------|
| 0          | 00         | 2104       |
| 4          | 04         | 3111       |
| 8          | 08         | 2037       |

Table 98. Codes that Identify Error Messages Issued by DFSCNXA0 (continued)

| Code (Dec) | Code (Hex) | Msg# (DFS) |
|------------|------------|------------|
| 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       |
| 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** Time stamp

**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

## APPC/IMS Diagnostic Aids

This topic details the following diagnostic aids:

- “LU Manager Trace”
- “LU 6.2 Module-to-Code Cross-Reference Table” on page 390
- “APPC/MVS Verb-to-Code Cross-Reference Table” on page 391
- “DFS1959E Message Information” on page 392
- “Diagnostics for Use with Synchronous APPC and OTMA with Shared Queues” on page 400
- “SNAPs and Dumps” on page 405

## 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 154.

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

Table 99 shows the general format of an LU manager trace record. Each record is 8 words long. Word 0 holds standard information for each record.

Table 99. LU Manager Trace Record Format

| WORD 0 |         | WORD 1 | WORD 2 | WORD 3 | WORD 4 | WORD 5 | WORD 6 | WORD 7 |
|--------|---------|--------|--------|--------|--------|--------|--------|--------|
| ID     | SEQ NUM |        |        |        |        |        |        |        |

**where**            **represents**

**ID**                Two-byte trace ID.

**SEQ NUM**        Two-byte trace sequence number assigned by the IMS trace component.

Words 1 thru 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'7B07'** XCF sendmsg

| **Word 1**  
|                • **Byte 0:** Module number - see Table 100 on page 390  
|                • **Bytes 2-3:** Function Return Code  
| **Word 2**        A(TIB)  
| **Words 3-6**    TIB MSG PREFIX URTOKEN  
| **Word 7**        Time stamp (STCK)

| **TRACE ID = X'7B08'** AWE server TIB POST

| **Word 1**  
|                • **Byte 0:** Module number - see Table 100 on page 390  
|                • **Bytes 2-3:** Return code  
| **Word 2**        A(ECB)  
| **Word 3**        AOS POST CODE  
| **Word 4**        A(TIB)

| **TRACE ID = X'7B09'** Request sent to RRS AWE server

| **Word 1**  
|                • **Byte 0:** Module number - see Table 100 on page 390  
|                • **Bytes 2-3:** AWRRFUNC  
| **Word 2**        A(ECB)  
| **Word 3**        AOS POST CODE  
| **Words 4–7**    MSG PREFIX URTOKEN

**TRACE ID = X'7C01'** Normal return from APPC/MVS

**Word 1**  
• **byte 0:** Module number - See Table 100 on page 390.  
• **byte 1:** ATB call number - See Table 101 on page 391.  
• **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 147 on page 389 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 0=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  10
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 147. Example of an LU Manager Trace

## LU 6.2 Module-to-Code Cross-Reference Table

You can use Table 100 to associate code xx in message DFS1959E and the module number in trace records X'7Bxx' and X'7Cxx' with a module.

Table 100. LU 6.2 Module-to-Code Cross-Reference Table

| Mod Num<br>(Dec) | Mod Num<br>(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                  |
| 12               | 0C               | DFS6LUS2 | LU 6.2 services interface 3                  |
| 16               | 10               | DFSRAC60 | RACF interface module                        |
| 21               | 15               | DFS6RST0 | LU 6.2 restart processor                     |
| 22               | 16               | DFS6CKP0 | LU 6.2 checkpoint processor                  |
| 24               | 18               | DFSGIDC0 | 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) |
| 40               | 28               | DFSCMLC0 | MSC SQ APPC/OTMA Message Router              |
| 41               | 29               | DFSCMS00 | MS Analyzer                                  |
| 50               | 32               | DFSXLUM0 | LUM TCB Initialization routine               |
| 51               | 33               | DFSYIOE0 | OTMA Input and Output user exit              |
| 52               | 34               | DFSXXCF0 | XCF TCB initialization                       |
| 53               | 35               | DFSXRM00 | 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                   |



Table 100. LU 6.2 Module-to-Code Cross-Reference Table (continued)

| Mod Num (Dec) | Mod Num (Hex) | Module   | Description                                      |
|---------------|---------------|----------|--------------------------------------------------|
| 71            | 47            | DFSAOSW0 | APPC/OTMA SMQ AWE server                         |
| 72            | 48            | DFSRGFS0 | RRS Server, AWE PROCESSOR                        |
| 90            | 5A            | DFSXAOS0 | DFSXAOS0 APPC/OTMA SMQ Enablement Initialization |

## APPC/MVS Verb-to-Code Cross-Reference Table

You can use Table 101 to associate the ATB call number in trace records X'7Cxx' with an APPC/MVS verb.

Table 101. 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             | ATBGTA2   | 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             | ATBUNID   | Unidentify scheduler from APPC/MVS      |
| 1C             | ATBIDN4   | Identify scheduler to APPC/MVS          |
| 1D             | ATBUID4   | Unidentify scheduler from APPC/MVS      |
| 1E             | ATBVERS   | Version service                         |
| 1F             | ATBALC5   | Allocate a conversation                 |

Table 101. APPC/MVS Verb-to-Code Cross-Reference Table (continued)

| Verb Num (Hex) | Verb Name | Verb Description        |
|----------------|-----------|-------------------------|
| 20             | ATBSTO5   | Set timeout value       |
| 21             | 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 100 on page 390. 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 DFSPPOOL to acquire storage for PL/AS variables using the DFSLUMGT macro.

**xx99** Failure in DFSPPOOL to release storage for PL/AS variables using 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 using DFSPPOOL macro.

**0403** Failure to free LUMP pool buffer using DFSPPOOL 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.

### **DFSAOSW0**

| <b>RC</b>   | <b>Description</b>                                 |
|-------------|----------------------------------------------------|
| <b>7101</b> | Unknown request code.                              |
| <b>7109</b> | Zero TIB address for send output.                  |
| <b>7110</b> | Failure in QUERY of DFSXCF macro.                  |
| <b>7116</b> | Zero header address for send output.               |
| <b>7121</b> | Failure to get AWE storage using DFSBCB macro.     |
| <b>7133</b> | Transaction not found for notify.                  |
| <b>7134</b> | Other than transaction found for notify.           |
| <b>7136</b> | Wrong message number in SEND DFS MESSAGE function. |
| <b>7144</b> | XCF parameter length too large.                    |
| <b>7144</b> | Unknown subfunction for Common XCF Communications. |
| <b>7150</b> | Failure to get LUMP storage using DFSPPOOL macro.  |
| <b>7190</b> | Failure in QUERY in DFSXCF macro.                  |

### **DFSAPPC0**

| <b>RC</b>   | <b>Description</b>                                                  |
|-------------|---------------------------------------------------------------------|
| <b>0201</b> | DFSQMGR Get Unique call failure, RC not 0.                          |
| <b>0202</b> | DFSQMGR Get Next call failure, RC not 0 and QTP1EOM=0.              |
| <b>0203</b> | DFSQMGR Enqueue call failure, RC not 0.                             |
| <b>0204</b> | DFSQMGR Dequeue call failure, RC not 0.                             |
| <b>0205</b> | DFSQMGR Insert Move call failure, RC not 0.                         |
| <b>0206</b> | DFSQMGR Insert Move call failure, RC not 0.                         |
| <b>0207</b> | DFSQMGR Cancel Input call failure, RC not 0.                        |
| <b>0208</b> | Failure to read DFSAPPC message from shared queues.                 |
| <b>0209</b> | DFSQMGR Insert Move without LU62 MSG PREFIX call failure, RC not 0. |
| <b>0210</b> | DFSQMGR Get Next call failure, RC not 0 and QTP1EOM=0.              |
| <b>0211</b> | DFSQMGR Get Next call failure, RC not 0.                            |
| <b>0212</b> | DFSQMGR Get Unique call failure, RC not 0.                          |
| <b>0250</b> | Failure to find or create asynchronous control block.               |
| <b>0260</b> | Router call failure. DFSICLR0 call, RC not 0.                       |
| <b>0270</b> | DFSUSE FUNC=NOUSE call failure, RC not 0.                           |

### **DFSATB00**

| <b>RC</b>   | <b>Description</b>                                        |
|-------------|-----------------------------------------------------------|
| <b>0901</b> | Calling module requesting unsupported APPC/MVS verb name. |

### **DFSCMD00**

| <b>RC</b> | <b>Description</b> |
|-----------|--------------------|
|-----------|--------------------|

- 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 Locate call failure, RC not 0.
- 0321 Failure to get LUMP pool buffer using DFSPPOOL macro.
- 0322 Failure to free LUMP pool buffer using DFSPPOOL 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 area 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 using DFSCMAP0.
- 4102 Failure in LUMIF GU call using DFSCMAP0.
- 4103 Failure in LUMIF GU call using DFSCMAP0.

## DFSHCI00

### RC Description

- 3501 Failure to get AWE storage using DFSBCB.

## DFSRLM00

### RC Description

- 0501 AWE extension not a FMH5 Attach request.
- 0502 Synchronous control block creation failure using DFS62DST FUNC=FINDD.
- 0503 Error freeing XAWE. Unknown storage pool.
- 0504 Error freeing XAWE using STORAGE macro.
- 0505 AWE not an FMH5 Attach request.
- 0506 Error posting DFSRLM10 using DFSSERVR macro.
- | 0507 Failure in Identify Protected Conversation Context.

## 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 using DFSPPOOL macro.
- 0608 Failure to free LUMP pool buffer using DFSPPOOL 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.
- I 0618 Failure in DFSBCB to free AWE.
- 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.
- I 0622 Failure in DFSQMGR to cancel a message, RC not 0.
- I 0623 Failure in DFSQMGR to delete a message, RC not 0.
- I 0624 Failure in DFS62FD0 getting an asynchronous control block (DFS62DST FUNC=FIND).

## 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.
- I 0107 Failure to get AWE using DFSBCB macro.
- 0111 DFSSLUM0 has been called to deliver a message with zero length to the front-end IMS system. A DFS2224 message will be sent instead.
- 0121 Failure to get LUMP pool buffer using DFSPPOOL macro.
- 0122 Failure to free LUMP pool buffer using DFSPPOOL 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 using DFSBCB macro.
- 3102 Error freeing XAWE using 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 using DFSPPOOL macro.
- 3111 Error freeing CIOP storage using DFSPPOOL 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 using DFSSSERVR macro.
- 3115 Error Checking synchronous control block using DFSSSERVR macro.
- 3116 VTAM MODIFY USERVAR failed for activation of primary system.

## DFS6IDC0

### RC Description

- 2401 Unable to obtain storage for BPAM buffer using STORAGE macro.
- 2402 Unable to release storage for BPAM buffer using STORAGE macro.
- 2403 Unknown DFS warning message number.
- 2404 Failure to get LUMP pool buffer using DFSPPOOL macro.
- 2405 Failure to free LUMP pool buffer using DFSPPOOL macro.

## DFS6LUS0

### RC Description

- 1004 No synchronous control block given in SEND service call.
- 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 using DFSBCB macro.
- 1013 Unable to create an asynchronous control block using DFS62DST FUNC=FOUND.
- 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(FOUND).
- 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 using 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.
- | 1031 Invalid TPN=DFSSIDE in CHNG service call.
- | 1032 Unable to find LU 6.2 descriptor entry in BLDPRE service call using DFS62DST macro.
- | 1060 Failure in DFSBCB to get AWE.
- | 1061 Failure in DFSBCB to free AWE.
- | 1062 Failure to get LUMP pool buffer using DFSPPOOL macro.
- | 1063 Failure in SENDMSG using DFSXCF macro.
- | 1064 Failure to free LUMP pool buffer using DFSPPOOL macro.

### DFS6LUS1

- | <b>RC</b> | <b>Description</b>                                                                              |
|-----------|-------------------------------------------------------------------------------------------------|
| 1110      | Unknown service call in main program.                                                           |
| 1117      | No message prefix or synchronous/asynchronous control block given in INQY service call.         |
| 1123      | Unable to clean up TP.                                                                          |
| 1124      | Unable to clean up in the address space.                                                        |
| 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. |
| 1127      | DFSLUS1 cannot find TIB/QAB.                                                                    |
| 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.                         |
| 1142      | Unable to find or to create a synchronous control block in FPGU service call.                   |
| 1143      | Unable to free a synchronous control block (DFS62DST FUNC=RELEASE).                             |
| 1150      | Return code from ATBRCVW in PH1 service call (abort synchpoint).                                |
| 1151      | Return code from ATBGTA2 in PH1 service call (abort synchpoint).                                |

### DFS6LUS2

- | <b>RC</b> | <b>Description</b>                                     |
|-----------|--------------------------------------------------------|
| 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.             |
| 1207      | Failure to get LUMP pool buffer using DFSPPOOL macro.  |
| 1208      | Failure to free LUMP pool buffer using DFSPPOOL macro. |

- | **1209** Failure to get MSEB storage using DFSBCB macro.
- | **1210** Failure to free MSEB storage using DFSBCB macro.
- | **1211** Failure to get HIOP storage using DFSPPOOL macro.
- | **1212** Failure to free HIOP storage using DFSPPOOL macro.
- | **1224** CQS 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.
- 3213** DFSCS failed adding synchronous control block to chain.
- 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.
- | **3222** Failure to free HIOP storage using DFSPPOOL macro.
- | **3223** Failure to free HIOP storage using DFSPPOOL macro.
- | **3224** Failure to free MSEB storage using DFSBCB macro.



## 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

- 5101** Failure in DFSQMGR Get Unique for notify message.
- 5102** Failure in DFS62FD0 finding an asynchronous control block for notify message.
- 5109** Unknown return code from z/OS clean address space call.
- 5110** Unknown return code from z/OS unidentify call.
- 5111** IXCLEAVE unsuccessful.

## DFSHAV70

### RC Description

- 3709** Unknown return code from z/OS clean address space call.
- 3710** Unknown return code from z/OS unidentify call.
- 3711** IXCLEAVE unsuccessful.

## DFSXLUM0

### RC Description

- 5009** Unknown return code from z/OS clean address space call.
- 5010** Unknown return code from z/OS unidentify call.
- 5011** IXCLEAVE unsuccessful.

## DFS1965 APPC/MVS Call Failure

A call to APPC/MVS had an unexpected return code. The call for FUNCTION=aaaaaaa 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

## Diagnostics for Use with Synchronous APPC and OTMA with Shared Queues

Synchronous APPC and OTMA message processing in the Shared Queues environment introduces additional diagnostic considerations for the message flow. In addition to the APPC and OTMA traces already used, some other facilities include:

- IMS Resource Recovery Trace. For more information, see “Resource Recovery Services Trace” on page 225.
- z/OS Resource Recovery Trace. For more information, see the “Component Trace” section in *z/OS MVS Diagnosis: Tools and Service Aids*.
- z/OS APPC Trace. For more information, see the “Component Trace” section in *z/OS MVS Diagnosis: Tools and Service Aids*.
- Console dumps of the RRS and APPC address and data spaces. For more information, see the “SVC Dump” section in *z/OS MVS Diagnosis: Tools and Service Aids*.

## 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.

---

## OTMA Diagnostic Aids

This topic describes the following diagnostic information to help you analyze problems in OTMA.

- “OTMA Trace”
- “OTMA Module-to-Code Cross-Reference Table” on page 403
- “OTMA Verb-to-Code Cross-Reference Table” on page 404
- “DFS1269E Message Information” on page 405
- “OTMA Log Records” on page 405
- “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 154.

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

Table 102 shows the format of OTMA trace records. Each record is eight words long. Word 0 holds standard information.

Table 102. OTMA Trace Record Format

| WORD 0 |            | WORD 1 | WORD 2 | WORD 3 | WORD 4 | WORD 5 | WORD 6 | WORD 7 |
|--------|------------|--------|--------|--------|--------|--------|--------|--------|
| ID     | SEQ<br>NUM |        |        |        |        |        |        |        |

| <b>where</b>   | <b>represents</b>                                                |
|----------------|------------------------------------------------------------------|
| <b>ID</b>      | 2-byte trace ID                                                  |
| <b>SEQ NUM</b> | 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/z/OS entry

**Word 1**           Byte 0: Module number

                  Byte 1: XCF call number

**Words 2-7**       Control message

TRACE ID = X'5B02'XCF/z/OS 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 Trace Entry for User Exits**

| TRACE ID = X'5A05'User exit DFSYIOE0 module entry

| **Word 1**       Byte 0: module number X'33'  
 |                Byte 1-3: 0

| **Word 2**       A(ECB)

| **Word 3**       0

| **Word 4**       0

| **Word 5**       0

| **Word 6-7**    Time stamp (STCK)

| TRACE ID = X'5A06'User exit DFSYIOE0 module exit

| **Word 1**       Byte 0: module number X'33'  
 |                Byte 1-3: 0

| **Word 2**       A(ECB)

| **Word 3**       Exit RC set by the module

| **Word 4**       0

| **Word 5**       0

| **Word 6-7**    Time stamp (STCK)

| TRACE ID = X'5A07'User exit DFSYPRX0 module entry

| **Word 1**       Byte 0: module number X'31'  
 |                Byte 1-3: 0

| **Word 2**       A(ECB)

| **Word 3**       0

| **Word 4**       0

| **Word 5**       0

- | **Word 6-7**      Time stamp (STCK)
  
- | TRACE ID = X'5A08'User exit DFSYPRX0 module exit
- | **Word 1**          Byte 0: module number X'31'
- |                    Byte 1-3: 0
- | **Word 2**          A(ECB)
- | **Word 3**          Exit RC set by the module
- | **Word 4**          0
- | **Word 5**          IMS internal processing code
- | **Word 6-7**      Time stamp (STCK)
  
- | TRACE ID = X'5A09'User exit DFSYDRU0 module entry
- | **Word 1**          Byte 0: module number X'32'
- |                    Byte 1-3: 0
- | **Word 2**          A(ECB)
- | **Word 3**          0
- | **Word 4**          0
- | **Word 5**          0
- | **Word 6-7**      Time stamp (STCK)
  
- | TRACE ID = X'5A0A'User exit DFSYDRU0 module exit
- | **Word 1**          Byte 0: module number X'32'
- |                    Byte 1-3: 0
- | **Word 2**          A(ECB)
- | **Word 3**          Exit RC set by the module
- | **Word 4**          0
- | **Word 5**          IMS internal processing code
- | **Word 6-7**      Time stamp (STCK)

## | OTMA Module-to-Code Cross-Reference Table

You can use Table 103 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 103. OTMA Module-to-Code Cross-Reference Table*

| <b>Mod Num<br/>(Dec)</b> | <b>Mod Num<br/>(Hex)</b> | <b>Module</b> | <b>Description</b>                     |
|--------------------------|--------------------------|---------------|----------------------------------------|
| 19                       | 13                       | DFSYLUS0      | OTMA fast services                     |
| 20                       | 14                       | DFSYSTO0      | OTMA storage manager                   |
| 21                       | 15                       | DFSYRR00      | OTMA destination reroute setup routine |
| 22                       | 16                       | DFSYIO00      | OTMA input/output setup routine        |
| 23                       | 17                       | DFSYCM20      | OTMA command processor                 |
| 24                       | 18                       | DFS6DC0       | Read and build LU 6.2 descriptors      |

Table 103. OTMA Module-to-Code Cross-Reference Table (continued)

| Mod Num<br>(Dec) | Mod Num<br>(Hex) | Module   | Description                                |
|------------------|------------------|----------|--------------------------------------------|
| 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/ST0 (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               | DFSYPSO0 | TPIPE output AWE server DFSYPSO0           |
| 39               | 27               | DFSYSND0 | OTMA XCF interface                         |
| 40               | 28               | DFSCMLC0 | MSC Shared queues APPC/OTMA message router |
| 41               | 29               | DFSCMS00 | MSC ANALYZER                               |
| 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               | DFSYQFX0 | OTMA queue fixer                           |
| 49               | 31               | DFSYPRX0 | OTMA pre-routing exit routine DFSYPRX0     |
| 50               | 32               | DFSYDRU0 | OTMA default DRU exit routine DFSYDRU0     |
| 51               | 33               | DFSYIOE0 | OTMA input/output edit user exit routine   |

## OTMA Verb-to-Code Cross-Reference Table

You can use Table 104 to associate the XCF call number in trace record X'5B'xx with a z/OS XCF verb.

Table 104. z/OS XCF Verb-to-Code Cross-Reference Table

| Verb Num<br>(Hex) | Verb Name | Verb Description                            |
|-------------------|-----------|---------------------------------------------|
| 01                | IXCCREAT  | Defines a member to XCF                     |
| 02                | IXCJOIN   | Enables a member to join a group            |
| 03                | IXCQUERY  | Return information about groups and members |
| 04                | IXCMMSGO  | Sends a message to another active member    |
| 05                | IXCMMSGI  | 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 103 on page 403. 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                                                                    |
|-------------|--------------------------------------------------------------------------------|
| xx98        | Failure in DFSPPOOL to acquire storage for a variable with the DFSYMAGT macro. |
| xx99        | Failure in DFSPPOOL to release storage for a variable with the DFSYMARL macro. |

Other reason codes are module dependent.

## OTMA Log Records

To activate OTMA logging, enter one of the following trace commands from the master terminal or the z/OS 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 Diagnosing Fast Path Problemsrelated 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.

- | The following topics provide additional information:
- | • “Understanding Parsing Errors”
- | • “Debugging and Diagnostic Aids Provided by IMS Spool API” on page 409

## 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” on page 406 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 section

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.=yyyyyyy) 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'00000D0' 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 Produced"
- | • "Special Abend Processing"
- | • "Service Error Log Record 67D0" on page 410

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 through 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 9: 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 section describes the service aids that can help you analyze internal resource lock manager (IRLM) problems. These service aids are:

- “IRLM Lock Trace Analysis Using KBLA” discusses IRLM Lock Trace Analysis using Knowledge-Based Log Analysis (KBLA).
- “IRLM Dumps” on page 412 discusses IRLM dumps.
- “SYS1.LOGREC” on page 412 discusses software LOGREC records.
- “z/OS Component Trace” on page 412 discusses the z/OS Component Trace.

In addition, the IRLM generates diagnostic messages that begin with the prefix DXR. These messages are documented in *IMS Version 9: Messages and Codes, Volume 1*.

### IRLM Lock Trace Analysis Using KBLA

KBLA (Knowledge-Based Log Analysis) offers an approach for solving IRLM and IMS problems using IMS logs (OLDS/SLDS). This ISPF panel driven interface for the IRLM Lock Trace Analysis program can be accessed from KBLA Option 4.5, as shown in Figure 148.

```

== K.B.L.A. IRLM Lock Trace Analysis ==

COMMAND ==>

Input Trace DSN. USER.IMS1.OLDSP0.OTMU01.DECKS2 Cataloged? Y
IMS Log Version. . . . . 9
COPY1 DSN. . . .

Output DSN Keyword. . . . . PQ88956 The Output DSN will be:
USER.keyword.S.KBLA.*
Create dataset with raw output . . . (Y/N) USER.keyword.R.KBLA.*
Raw output sort selection:
Sorted by Database Name. . . . (A/D/N)
Sorted by RBA . . . . . (A/D/N)
Sorted by Database Name & RBA. . (A/D/N)
Sorted by PST Number . . . . . (A/D/N)
Sorted by Wait Elapsed Time. . . (A/D/N)

Log DSNs were extracted from RECON .
PDS member containing logs . . . .
    
```

Figure 148. IRLM Lock Trace Analysis

For more information, see the “IRLM Lock Trace Analysis Utilities (DFSKTLA0, DFSKTLB0, DFSKTLC0)” section in *IMS Version 9: Utilities Reference: System*.

---

## IRLM Dumps

The IRLM uses the SDUMP system services of z/OS 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 z/OS subsystem name (IRLMNM in the IRLM procedure). IRLM dump formatters that are shipped with IRLM 2.1 and 2.2 are specific to their release. If both releases of IRLM are active, IPCS must be configured to have access to the appropriate IRLM PDS. The release of the IRLM in the dump must match the release of the IRLM in the IPCS STEPLIB. For more information about making load libraries available to IPCS, see the “General Information about Writing an IPCS Exit Routine” section in *z/OS MVS Interactive Problem Control System (IPCS) Customization*.

To access z/OS 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 9: 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.

---

## z/OS Component Trace

Use the z/OS 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 z/OS TRACE CT command is described in *IMS Version 9: Command Reference* and *MVS/ESA System Commands*.

This command can only be entered from the master console. The command requires an appropriate level of z/OS 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 z/OS locking component.
- XCF** Trace all interactions with z/OS cross-system coupling services.
- XIT** Trace just asynchronous interactions with the z/OS 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.

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

```

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
    
```

The command that produced this output is: CTRACE COMP(IRLE) SUB((SLM)) FULL

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 00000008          |
| +0078    |           | C9D4E2C5        | 40404040 0423AD20 00000000 00000000 |
| +008C    |           | 00000000        |                                     |
| +0090    | TLA3..... | 000B0040        | 071663D8                            |
| +0098    | UDB.....  | C9D4E2C5        | 40404040 00000000 00000000 00080000 |
| +00AC    |           | 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 section describes diagnostic information to help you analyze problems in Fast Path. This includes:

- “Diagnosing Fast Path Problems”
- “DEDB Control Interval (CI) Problem Assistance Aids” on page 419
- “Fast Path External Trace” on page 424
- “Locating Fast Path Control Blocks and Tables” on page 422

### 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.

The following topics provide additional information:

- “ABENDU1026 Analysis”
- “Fast Path Transaction Retry” on page 418

### 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 topic 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 105 shows you where to find ODF information.

*Table 105. Locating Information About the Offline Dump Formatter (ODF)*

| For Information About                         | Refer to                                     |
|-----------------------------------------------|----------------------------------------------|
| Obtaining dumps suitable for input to the ODF |                                              |
| Running the ODF                               | IMS Utilities Reference: Transaction Manager |
| Using the ODF to solve problems               | “Formatting IMS Dumps Offline” on page 154   |

Before beginning the analysis, you need:

- A copy of the DFS2712I message
- A dump formatted by the ODF
- A copy of *IMS Version 9: 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 149. 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 149. 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 149, the module name is DBFMRCU0 and the subcode is 0053.

2. To find the meaning of the subcode, look up ABENDU1026 in *IMS Version 9: 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 9: 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 149, 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 150 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 149 on page 416, register 13 contains the address 0070767C.
  - In the \*\*DPST section of the formatted dump in Figure 150, 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 DBFMRCU003/21/8618.02PT01119 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 DBFMFG0002/04/8617.58PP35272 1B
SA 0070767C   WD1 00000000   HSA 00707634   LSA 007076C2   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 DBFMRSR002/13/8716.56PP58251 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 00B8CF78   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 00B8CF78   R8 02932A08   R9 02903310   R10 0070B040   R11 0086DF20   R12 008285F0
    
```

Figure 150. Example of a Save Area Set

6. In Figure 150, 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 418.)

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 topics 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 9: Messages and Codes, Volume 2* 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 415, 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 topic. 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 127.

This topic 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 topic are:

|                  |                         |
|------------------|-------------------------|
| <b>DOVF</b>      | Dependent overflow      |
| <b>IOVF</b>      | Independent overflow    |
| <b>RAP BLOCK</b> | Root-anchor point block |
| <b>SDEP</b>      | Sequential dependent    |

The following topics provide additional information:

- “CI Type Identification”
- “DEDB CI Formats”

### 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 |
|-------------------------|------------|
| <b>REORG CI</b>         | 00         |
| <b>RAP</b>              | 01         |
| <b>DOVF</b>             | 02         |
| <b>IOVF (SPACE MAP)</b> | 04         |
| <b>IOVF</b>             | 08         |
| <b>SDEP</b>             | 10         |

### DEDB CI Formats

This topic 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). Figure 151 shows the EQE list format: FLG (1 byte), EQE CNT (3 bytes), 10 available EQE entries (40 bytes).

|       | FLG<br>*               | EQE<br>CNT | EQE<br>ENTRY | ..... | EQE<br>ENTRY |
|-------|------------------------|------------|--------------|-------|--------------|
| bytes | 1                      | 3          | 4            |       | 4            |
|       | ----- 10-Entries ----- |            |              |       |              |

\* X'80' means more than 10 EQEs or error in 2nd CI.

Figure 151. EQE list in CI 1

**RAP CI**

Figure 152 shows the RAP CI.

```

0       2       4           8
FSEAP  0203   RBA of current  Segments, FSEs and Scraps
                overflow CI

      (02) - Indicates CUSN
                is in this CI.
```

Figure 152. RAP CI

**First DOVF CI**

The first DOVF CI has the format shown in Figure 153.

```

0       2       4           8
FSEAP  0203   RBA of current  Segments, FSEs and Scraps
                overflow CI

      (02) - Same as RAP CI -- these two bits combined
      (01) - Look here for space -- make the 03 in byte 3.
```

Figure 153. 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 the format shown in Figure 154.

```

0       2       4           8
FSEAP  02     RBA of next  Segments, FSEs and Scraps
                DOVF CI with
                space, last
                contains zeros
```

Figure 154. Other DOVF CIs

**First IOVF CI**

The CI shown in Figure 155 on page 421 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 155. First IOVF CI

### Other IOVF CIs

Figure 156 is a data CI - 119 data CIs follow each space map CI.

```

0          2    4          8
FSEAP    0802 4000uow#   Segments, FSEs and Scraps (allocated,
0008      0802 80000000   to UOW number; 0 is the first UOW).
                               FSE (CI not allocated).
    
```

(02) indicates CUSN is in this CI

Figure 156. Other IOVF CIs

### 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. Figure 157 shows the SDEP CI.

```

0          2 3    4          8
0000      1000  Partner name Segments inserted sequentially
                               and cannot be updated
    
```

(01) - Time stamp exists  
(01) - SDEP CI is full.

Figure 157. SDEP 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  
sss=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 bbbbbb 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 9: 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:

$$X'FFFF' - X'FFF6' = X'19' = 25 \text{ (decimal)}$$

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.ADFSSMPL. Remember to use XREF(FULL).

Table 106 on page 423 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.

Table 106. Fast Path Control Blocks and Work Areas that Appear in IMS Dumps

| Load List Name | Fast Path Block/Work Area       | Appearance in Dump |
|----------------|---------------------------------|--------------------|
| DBFCNT0        | Fast Path Global Control Blocks | IMS STM Task       |
| DFSEPnnn       | Fast Path EPSTs (nnn=000-999)   | IMS STM task       |

At Fast Path initialization, module DBFINI20 calculates the amount of contiguous ECSA storage that is needed in order to load DBFCNT0, 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:

$$\text{buffers} \times \text{buffer size} + \text{MSDB\_size}$$

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 107 when you are figuring out which specific control blocks are needed in your Fast Path environment.

The possible control block structure of DBFCNT0 appears in Table 107.

Table 107. Control Block Structure of DBFCNT0

| 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 DBFCNT0 are dumped.

## Fast Path External Trace

The Fast Path (FP) External Trace, not the same as Fast Path internal trace added in IMS Version 9, is a powerful tool to diagnose problems with Fast Path DL/I calls. Examples of such problems might be unexpected DL/I status codes or abends such as U1026. It is best suited to problems which can be easily recreated, and is not intended to be run routinely. The overhead and output volume of the trace can be very large. It is primarily intended for use by IBM Service specialists, but users might also find it useful. Also, you might be asked by IBM Service to capture Fast Path External trace data for analysis by IBM specialists.

Only dependent region activity is traced. This trace cannot be used to collect data on control region processes. Since most Fast Path DL/I call flow is normally done in the dependent region, this is not a serious limitation. However, if the PARDLI=1 option is used, DL/I processing is performed under the CTL TCB, which limits the usefulness of Fast Path External Trace.

**Recommendation:** Do not trace PARDLI=1 execution. Recreate the problem, if possible, without PARDLI=1.

The following topics provide additional information:

- “Trace Activation”
- “Trace Deactivation”
- “Diagnostic Data” on page 425
- “Fast Path Trace Entries” on page 425
- “Fast Path External Trace Examples” on page 425

## Trace Activation

There are three ways the trace can be activated:

- The Fast Path Transaction Retry function normally attempts to activate FP External Trace when a transaction is retried in an MPP or IFP region after an abend in Fast Path code. In this case, the trace is activated internally for the dependent region executing the retry and not for other dependent regions. The trace is deactivated after one retry attempt. The Fast Path Transaction Retry function dynamically allocates an FPTRACE DD statement as a JES SPOOL file, it also closes and deallocates an FPTRACE DD statement when the trace is deactivated at end of retry. The intent of this function is to provide first-failure data capture.
- A CCTL DRA thread can also request that FP External Trace be activated for a particular thread during the Create Thread process. Refer to the documentation for the CCTL for more information.
- The trace can also be activated with a /TRA SET ON TABLE FAST command. The Fast Path External Trace writes diagnostic data to a FPTRACE DD statement in the dependent region JCL. After the trace is activated, presence or absence of the FPTRACE DD statement determines whether data is traced for each active dependent region, including CCTL DRA threads. A spool file (SYSOUT=x) can be used for FPTRACE DD statement, or a DASD file used. DCB attributes are forced to LRECL=133, BLKSIZE=133, RECFM=FA by IMS when the DCB is opened.

**Recommendation:** Consider using a spool file (SYSOUT=x) rather than a disk file.

A certain amount of data related to the trace activation itself is traced (written to FPTRACE) before the determination is made that the trace is actually active or inactive.

**Recommendation:** Do not include an FPTRACE DD statement in your standard dependent region JCL. Add it only as required and then remove it after the trace data has been collected.

## Trace Deactivation

The trace is deactivated with a /TRA SET OFF TABLE FAST command.

## Diagnostic Data

Data is formatted as it is written. No offline formatting of the trace data is required.

### Trace Point Identifiers

The FP trace captures module flows, and at certain points, logic flows within modules. In most cases, there is a trace point at entry to a module and another at exit from the module. There might be additional trace points within the module. Each trace point has a unique 4-character identifier. To indicate nesting within call flows, using this unique 4-character identifier, the identifier is shifted rightwards at each level. Each trace entry is prefixed by the identifier located in columns 1-13. The relative position of the identifier within columns 1-13 indicates nesting level for example:

```
IRC1.....
.MCL0.....
```

Because the identifier has 4-characters and 13 positions are available, 9 levels of nesting are possible. Output lines with no identifier in columns 1-13 are continuations of the previous entry. It is convention that the module entry and exit trace entries differ by only one character. Usually, the module exit identifier is the same as the module entry identifier, except for one character. Normally, the first character of the identifier is shifted up one alphabetically for example:

```
.MCL0.....
.NCL0.....
```

### Trace Point Time Stamps

Trace point time stamps are labeled with TOD=xxxxxxx. The hexadecimal digits are the middle 4-bytes of an 8-byte STCK time stamp. The high order digit is approximately 1 second.

**Note:** Fields labeled TIME and DATE within trace entries refer to the compile date and time of the module involved, and have no relation to trace time.

### Trace Initialization Entries

Entries COT1 to TRAN at the beginning of the trace file refer to FP Trace initialization and should be ignored. Note that these entries are produced if a FPTRACE DD statement is present, even if the trace is not enabled.

### Key Trace Point Data Items

The data traced for each trace point varies. However, each field has a label which makes it simple to determine the contents of the entry. Here are some of the common and useful labels:

|                  |                                                                                 |
|------------------|---------------------------------------------------------------------------------|
| <b>Ra#b,Rab,</b> | Registers a-b follow.                                                           |
| <b>CALL</b>      | DL/I call function.                                                             |
| <b>TOKN</b>      | UOR's recovery token.                                                           |
| <b>MODU</b>      | Module entry point address.                                                     |
| <b>EPST</b>      | EPST address of the dependent region.                                           |
| <b>SSA</b>       | The first 30 bytes of the call SSA. Might contain residual data for short SSAs. |

### Fast Path Trace Entries

The Fast Path trace entries are documented in Appendix F, "Fast Path Trace Entries," on page 591.

### Fast Path External Trace Examples

The Fast Path external trace is shown in the examples below. Each portion of the trace is briefly explained in the text preceding it.

**Trace entries from COT1 to SIEX are tracing the initialization of FPTRACE:**

```

COT1.....TOD=B6B46252 WKAR=0A6FCC94 R0#F=00000031 007BF6B0 8AD8D6D8 00004700 8A6FC634 00000001 00CC4B20 7ABC7570 8AD8D6D8 007F6A
.....D8
BLTE.....TOD=B6B46259 R15=8AE1A060 DATE=01/0310.3 0#10=00004700 007AE900 8AD8D6C4 00004700 8A6FC634 0A6FCC94 00CC4B20 7ABC7570 8
.....AD8D6D8 007F6AD8 0A6FC040
BLTX.....R0#F=000000DD 007AE900 000000DC 007AF360 8A6FC634 0000000C 000000DD 007AE91C 00000001 007F6AD8 0A6FC040 007AE91C 8AE1A0
.....60 0A5BE738 000000DE 00000000 EPST=0A6FC040 TFTD=007AE900C4C2C6E3C6E3D6D400000000 TSTD=0000000000000000000000000000000000
.....XTOM=00000000
COT2.....R0#F=00004700 007AE900 007AE900 00004700 8A6FC634 0A6FCC94 00CC4B20 7ABC7570 8AD8D6D8 007F6AD8 0A6FC040 00CC5B78 8ADC00
.....D0 0A5BE6F0 8ADC033E 00000000 EPST=0A6FC040 TFTD=007AE900C4C2C6E3C6E3D6D400000000 TSTD=0000000000000000000000000000000000
.....XTOM=00000000
COT3.....R0#F=00004700 007AE900 007AE900 00004700 8A6FC634 0A6FCC94 00CC4B20 7ABC7570 8AD8D6D8 007F6AD8 0A6FC040 00CC5B78 8ADC00
.....D0 0A5BE6F0 8ADC033E 00000000 EPST=0A6FC040 TFTD=007AE900C4C2C6E3C6E3D6D400000000 TSTD=0000000000000000000000000000000000
.....XTOM=00000000
COTX.....TOD=B6B465B8 WKAR=0A6FCC94 R0#F=00004700 007AE900 007AE900 00004700 8A6FC634 0A6FCC94 00CC4B20 7ABC7570 8AD8D6D8 007F6A
.....D8 0A6FC040 00CC5B78 8ADC00D0 0A5BE6F0 8ADC033E 00000000 MODU=8ADC00D0 DATE=08/01/031 TIME=10.32U LCHA=UP9HCT011
COTE.....TOD=B6B465B9 WKAR=0A6FCC94 R0#F=00000032 007BF6B0 0000063AC 8ADC00D0 8A6FC624 00000001 00CC4B20 7ABC7570 00CC5B78 007F6A
.....D8 0A6FC040 00CC5B78 8ADC00D0 0A5BE6F0 8ADC010C 00000000 MODU=8ADC00D0 DATE=08/01/031 TIME=10.32U LCHA=UP9HCT011
COTY.....TOD=B6B465BA WKAR=0A6FCC94 R0#F=00000030 007BF6B0 8AD91DD8 8ADC00D0 8A6FC624 00000001 00CC4B20 7ABC7570 00CC5B78 007F6A
.....D8 0A6FC040 00CC5B78 8ADC00D0 0A5BE6F0 8ADC019E 00000000 MODU=8ADC00D0 DATE=08/01/031 TIME=10.32U LCHA=UP9HCT011
STS9.....TOD=B6B46614 R15=8ADDC410 DATE=01/0310.3 0#10=00000950 007BF6B0 0000063AC 8ADDC410 8A6FC624 00000001 00CC4B20 7ABC7570 0
.....0CC5B78 007F6AD8 0A6FC040
STSX.....TOD=B6B4664D R15=00000000 DATE=01/0310.3 0#10=00000002 0A5BE060 007AE900 8ADDC410 8A6FC624 00000001 00CC4B20 7ABC7570 0
.....0CC5B78 007F6AD8 0A6FC040
SIEX.....TOD=B6B4664E R15=00000000 DATE=01/0310.3 0#10=00000950 007BF6B0 0000063AC 8ADDC410 8A6FC624 00000001 00CC4B20 7ABC7570 0
.....0CC5B78 007F6AD8 0A6FC040

```

**End of trace initialization:**

```

FPR3.....TOD=B6B60ECA WKAR=0A6FCC94 R0#F=000121F8 0A69602C C7C8E440 00000000 00000001 0A5BE060 0BC14F5B C4C5D7C1 D9E3D4E3 4D0060
.....18 0A6FC040 00010000 8ADBA272 0001EE48 0001EC80 0E48B350 MODU=8ADBA272 DATE=08/05/03P TIME=PQ6040 LCHA=01 11AB

```

**DL/I call start in DBFIRC10:**

```

IRC1.....TOD=B6B60ECC R015=00000000 00000C17 C7C8E440 0A69602C 0BC04F54 0A5BE060 00000000 8A71C580 D9E3D4E3 0A9F1048 0A6FC040 00
.....CC5B78 8ADBA272 0A5BE618 0A6FC9F2 00000001 LCRC=0A9F1048 TOKEN=E2E8E2F340404040000000006000000000 EPCB=0A69602C PCBA=0A52F
.....35C PCBD=000121F8 ESCD=00CC5B78 MADR=8AD477B0 EPST=0A6FC040 WKAR=0A6FCC94 SVIO=0BC04F54 SVIL=0000 SVSN=0001
.MCL0.....TOD=B6B60EEC CALL=GHU EPST=0A6FC040 WKAR=0A6FCC94 EPCB=0A69602C CCID=00 PRGP=00000000 LKFP=00000000 PCB=0A52F3
.....5C SSA=DEPARTMT(DEPTKEY = R1210000001 MODU=0AD4AD80 DATE=08/01/031 TIME=10.25E LCHA=E)
} }

```

**Call is GHU - first 30 bytes of SSA are traced:**

```

..SAGE.....TOD=B6B60ED0 R0#9=00000000 00000C17 C7C8E440 0A69602C 0BC04F54 0AD4B67C FFFFE20 FEE02D87 00CC4B20 000000FF R14=8AD4AF9
.....0 R15=0AD728C8 EPCB=0A69602C FLGM=00 DMAC=00000000 ARBA=00000000 UOWO=00000000 CCID=8C CCNT=00000001 UBLK=00000000 KUBL
.....=00000000 DEDB=10 LKFP=00000000 PRGP=00000000 CLEV=0A6960EC MLTE=0BC04F54 POPT=00 SGLS=00 CLOC=00000000 DMHR=00000000 P
.....RBA=00000000 CRBA=00000000 NRBA=00000000 GRBA=00000000 XRBA=00000000 SGCD=00 PROF=0000 LEVL=00 KEYL=00 FLGA=00 FLGB=00
.....ACCK=0000 KEYO=0000 SDBS=00000000 MLTE=0BC04F54 SFRX=00 SFSX=00 SFWX=00 SFZX=00 SFMX=00 PREF=0000 SCNT=00000000 SWC1=00
.....SWC2=00 SWC3=00 LOPR=00 SNAP=00000000 EPST=0A6FC040 SCVL=00 REOP=00 FDLN=00 FDOF=0000 DEDB=00 COMP=00000000 STAT=
.....WCH=00 MODU=0AD728C8 DATE=08/01/031 TIME=10.28E LCHA=E)
} }

```

**SSA handler for GET type calls:**

```

...SAGI.....TOD=B6B60EDB R0#9=00000000 00000C17 C7C8E440 0A69602C 00000000 0A6FC588 FFFFE20 FEE02D87 00CC4B20 000000FF R14=8AD7293
.....6 R15=0AD72030 EPCB=0A69602C FLGM=00 DMAC=00000000 ARBA=00000000 UOWO=00000000 CCID=8C CCNT=00000001 UBLK=00000000 KUBL
.....=00000000 DEDB=10 LKFP=00000000 PRGP=00000000 CLEV=0A6960EC MLTE=00000000 POPT=80 SGLS=CE CLOC=040C0000 DMHR=00FC7B38 P
.....RBA=070C2000 CRBA=00000000 NRBA=814171C0 GRBA=00000000 XRBA=00000000 SGCD=9E PROF=0008 LEVL=07 KEYL=0C FLGA=10 FLGB=00
.....ACCK=80CE KEYO=B594 SDBS=078D2000 MLTE=00000000 SFRX=00 SFSX=00 SFWX=04 SFZX=0C SFMX=00 PREF=8002 SCNT=00000000 SWC1=00
.....SWC2=00 SWC3=04 LOPR=00 SNAP=00000000 EPST=0A6FC040 SCVL=00 REOP=00 FDLN=00 FDOF=0000 DEDB=00 COMP=00000000 STAT=
.....WCH=00 MODU=0AD72030 DATE=08/01/031 TIME=10.28E LCHA=E)
} }
...VSNA.....TOD=B6B60EDF R0#9=00000000 00000C17 C7C8E440 0A69602C 00000000 0A6FC588 8BC14F5B FEE02D87 00CC4B20 000000FF R14=8AD7208
.....6 R15=0AD74478 EPCB=0A69602C FLGM=00 DMAC=00000000 ARBA=00000000 UOWO=00000000 CCID=8C CCNT=00000001 UBLK=00000000 KUBL
.....=00000000 DEDB=10 LKFP=00000000 PRGP=00000000 CLEV=0A6960EC MLTE=00000000 POPT=80 SGLS=CE CLOC=040C0000 DMHR=00FC7B38 P
.....RBA=070C2000 CRBA=00000000 NRBA=814171C0 GRBA=00000000 XRBA=00000000 SGCD=9E PROF=0008 LEVL=07 KEYL=0C FLGA=10 FLGB=00
.....ACCK=80CE KEYO=B594 SDBS=078D2000 MLTE=00000000 SFRX=00 SFSX=00 SFWX=04 SFZX=0C SFMX=00 PREF=8002 SCNT=00000000 SWC1=00
.....SWC2=00 SWC3=04 LOPR=00 SNAP=00000000 EPST=0A6FC040 SCVL=00 REOP=00 FDLN=00 FDOF=0000 DEDB=00 COMP=00000000 STAT=
.....WCH=00 MODU=0AD74478 DATE=08/01/031 TIME=10.28E LCHA=E)
} }

```

**Verify segment name:**

```

...VSNA.....TOD=B6B60EE3 R0#9=00000080 00000C17 C7C8E440 0A69602C 0A6960EC 0A6FC588 8BC14F6E 0A6FC5E4 0A696168 000000FF R14=0000001
.....3 R15=00000000 EPCB=0A69602C FLGM=00 DMAC=00000000 ARBA=00000000 UOWO=00000000 CCID=8C CCNT=00000001 UBLK=00000000 KUBL
.....=00000000 DEDB=10 LKFP=00000000 PRGP=00000000 CLEV=0A6960EC MLTE=0A6960EC POPT=00 SGLS=01 CLOC=00000000 DMHR=00000000 P
.....RBA=00000000 CRBA=00000000 NRBA=00000000 GRBA=00000000 XRBA=00000000 SGCD=01 PROF=0000 LEVL=01 KEYL=0B FLGA=88 FLGB=00
.....ACCK=000C KEYO=0008 SDBS=0AD09210 MLTE=0A6960EC SFRX=00 SFSX=00 SFWX=00 SFZX=00 SFMX=00 PREF=0006 SCNT=00000001 SWC1=00

```



```

| .....ACCK=000C KEY0=0008 SDBS=0AD09210 MLTE=0A6960EC SFRX=00 SFSX=00 SFWX=00 SFZX=00 SFMX=00 PREF=0006 SCNT=00000001 SWC1=00
| ..... SWC2=02 SWC3=00 LOPR=82 SNAP=0A696168 EPST=0A6FC040 SCVL=01 REOP=80 FDLN=0B FDOF=0008 DEDB=80 COMP=0A4998D0 STAT=
| .....WCH=00 MODU=0AD4E428 DATE=08/01/031 TIME=10.26E LCHA=E}
| }

```

**Determine possibility of randomizing:**

```

| .....MD03....TOD=B6B613FB WKAR=0A6FCC94 R0#F=00000004 0A4998C0 0A4998C6 0A69602C 0A6960EC 00000010 00000001 FEE02D87 00000001 000000
| .....01 0A6FC040 00CC5B78 0AD4E428 0A5BE780 8AD71866 00000000 MODU=0AD4E428 DATE=08/01/031 TIME=10.26E LCHA=E}
| }

```

**No position:**

```

| .....MD49....TOD=B6B613FC WKAR=0A6FCC94 R0#F=00000004 0A4998C0 0A4998C6 0A69602C 0A6960EC 00000010 00000001 FEE02D87 00000001 000000
| .....01 0A6FC040 00CC5B78 0AD4E428 0A5BE780 8AD71866 0A4998D0 MODU=0AD4E428 DATE=08/01/031 TIME=10.26E LCHA=E}
| }

```

**Use randomizer:**

```

| .....MDRA....TOD=B6B613FD R0#9=00000004 0A4998C0 0A4998C6 0A69602C 0A6960EC 00000010 00000001 FEE02D87 00000001 00000001 R14=8AD7186
| .....6 R15=0A4998D0 EPCB=0A69602C FLGM=04 DMAC=00000000 ARBA=00000000 UWOW=00000000 CCID=8C CCNT=00000001 UBLK=00000000 KUBL
| .....=00000000 DEDB=10 LKFP=00000000 PRGP=00000000 CLEV=0A6960EC MLTE=0A6960EC POPT=00 SGLS=01 CLOC=00000000 DMHR=00000000 P
| .....RBA=00000000 CRBA=00000000 NRBA=00000000 GRBA=00000000 XRBA=00000000 SGCD=01 PROF=0000 LEVL=01 KEYL=0B FLGA=88 FLGB=00
| .....ACCK=000C KEY0=0008 SDBS=0AD09210 MLTE=0A6960EC SFRX=00 SFSX=00 SFWX=00 SFZX=00 SFMX=00 PREF=0006 SCNT=00000001 SWC1=00
| ..... SWC2=02 SWC3=00 LOPR=82 SNAP=0A696168 EPST=0A6FC040 SCVL=01 REOP=80 FDLN=0B FDOF=0008 DEDB=80 COMP=0A4998D0 STAT=
| .....WCH=00 MODU=0AD4E428 DATE=08/01/031 TIME=10.26E LCHA=E}
| }
| .....MGAP....TOD=B6B61416 R0#F=00000004 00000000 0000000A 0A69602C 0A6960EC 00000001 00000001 FEE02D87 0AD091C0 0A69602C 0A6FC040 00
| .....CC5B78 0AD54858 0A5BE780 8AD71906 0AD54858

```

**Get Root anchor point:**

```

| .....ERAN....TOD=B6B61417 R0#F=00000006 0AD0923C 0000000C 0A4998D0 0000000C 00000078 0A6FC140 FEE02D87 0AD091C0 0A69602C 0A6FC040 00
| .....CC5B78 0AD54858 0A5BE780 8AD71906 8B6900E8 TOD=B6B61417 R0#9=00000006 0AD0923C 0000000C 0A4998D0 0000000C 00000078 0A6F
| .....C140 FEE02D87 0AD091C0 0A69602C R14=8AD71906 R15=8B6900E8 EPCB=0A4998D0 FLGM=00 DMAC=00000000 ARBA=00000000 UWOW=000000
| .....00 CCID=00 CCNT=00000000 UBLK=00000000 DEDB=00 MLTE=0000000C LEVL=80 SDBS=00000000 EPST=0A6FC040 SWAR=DEPTKEY =
| .....

```

**Entry to randomizer:**

```

| .....XРАН....TOD=B6B61419 WKAR=0A6FCC94 R0#F=00000000 0AD09288 0000000C 0A4998D0 0000000C 00000078 0A6FC140 FEE02D87 0AD091C0 0A6960
| .....2C 0A6FC040 00CC5B78 0AD54858 0A5BE780 8AD54982 00000000 MODU=0AD54858 DATE=08/01/03P TIME=PQ7029 LCHA=96 ABCE}

```

**Entry to randomizer:**

```

| .....NGAP....TOD=B6B6141A R0#F=00000000 0AD09288 00000000 00001004 00000004 00000000 0A6FC140 FEE02D87 0AD091C0 0A69602C 0A6FC040 00
| .....CC5B78 0AD54858 0A5BE780 D9C1D5C4 00000000
| .....MGRF....TOD=B6B6141B WKAR=0A6FCC94 R0#F=00000000 00000000 0000000A 0A69602C 0A6960EC 00000001 00000001 FEE02D87 0AD09288 000000
| .....00 0A6FC040 00CC5B78 0AD56F70 0A5BE780 8AD71912 0AD56F70 MODU=0AD56F70 DATE=08/01/031 TIME=10.26E LCHA=E}
| }

```

**Get Root:**

```

| .....MBED....TOD=B6B6141C AREA=DEPTARO
| .....0AD09288 00000000 RE#F=8AD5730A 8A71C580

```

**Get Control (CI RBA x'1000')**

```

| .....EXXC....TOD=B6B6141D WKAR=0A6FCC94 R0#F=00000000 00000000 0A6FC284 0A69602C 0A6960EC 8AD56FC4 00001000 00001000 0AD09288 000000
| .....00 0A6FC040 00CC5B78 0AD5AB48 0A5BE810 8AD486BA 0AD5AB48 MODU=0AD5AB48 DATE=IVELOCK08 TIME=8/05/0 LCHA=03E}
| }

```

**Get EXCL CI lock (CI RBA x'1000')**

```

| .....NGXC....TOD=B6B6141F R0#9=0000D800 0A5BE060 00000000 09E6A040 00000008 00001000 00001000 0A960CB0 0AD09288 00000000 RE#F=00 00
| .....XCRB=0A960CB0 NEXT=00000000 SHDC=00000000 OPST=0A6FC040 UWOW=00001000 FLGS=80 DMHR=00000000
| .....MSRB..R0#F=FFFF4040 0AA922C0 0AD09288 0A69602C 8AD48710 0AD3FA20 00001000 0A960CB0 0AD09288 0AA922C0 0A6FC040 00CC5B78 0AD6E8
| .....30 0A5BE858 8AD48B1E 0AD6E830 EPST=0A6FC040 WKAR=0A6FCC94 MODU=0AD6E830 DATE=08/05/031 TIME=13.58P LCHA=PQ71804 0

```

**Synchronous read of CI:**

```

| .....VSOR.TOD=B6B61424 WKAR=0A6FCC94 R0#F=FFFF4040 00000000 0AD09288 0A69602C 0AA922C0 0AD3FA20 00001000 0A960CB0 0AD09288 0AA922
| .....C0 0A6FC040 00CC5B78 0AD98E88 0A5BE8A0 8AD6E902 0AD98E88 MODU=0AD98E88 DATE=08/05/031 TIME=14.00K LCHA=KX20294 0
| .....VSOR.TOD=B6B61426 WKAR=0A6FCC94 R0#F=0AB38800 00000000 0AD09288 00000001 00000000 0A95B268 00001000 00000000 0AD09288 0AA922
| .....C0 0A6FC040 00CC5B78 0AD98E88 0A5BE8A0 0009A800 00000000 MODU=0AD98E88 DATE=08/05/031 TIME=14.00K LCHA=KX20294 0

```



.SSAX.....SSA=DEPARTMT

**First SSA:**

```

.SSR9.....TOD=B6B61474 R15=0AD72D18 DATE=01/0310.2 #0#10=00000000 00000C18 D9C5D7D3 0A69602C 0BC04F54 0AD4B68C FFFFE10 FF3C46E1 0
.....0CC4B20 000000EE 0A6FC040
.VSNA.....TOD=B6B61475 R0#9=00000000 00000C18 D9C5D7D3 0A69602C 0A6960EC 0A6FC588 8BC14F5B FF3C46E1 00CC4B20 000000EE R14=8AD72DD
.....A R15=0AD74478 EPCB=0A69602C FLGM=E0 DMAC=0AD09288 ARBA=00001004 UOWO=00000000 CCID=E0 CCNT=00000003 UBLK=00000000 KUBL
.....=00000000 DEDB=10 LKFP=0A6960EC PRGP=0A6960EC CLEV=0A6960EC MLTE=0A6960EC POPT=00 SGLS=01 CLOC=0AB38008 DMHR=0AA922C0 P
.....RBA=00001004 CRBA=00001008 NRBA=00001268 GRBA=00001268 XRBA=00001004 SGCD=01 PROF=0000 LEVL=01 KEYL=0B FLGA=88 FLGB=00
.....ACCK=000C KEYO=0008 SDBS=0AD09210 MLTE=0A6960EC SFRX=00 SFSX=00 SFWX=00 SFZX=00 SFMX=00 PREF=0006 SCNT=00000002 SWC1=60
.....SWC2=02 SWC3=00 LOPR=82 SNAP=0A696168 EPST=0A6FC040 SCVL=01 REOP=80 FDLN=0B FDOF=0008 DEDB=80 COMP=0A4998D0 STAT=
.....WCH=00 MODU=0AD74478 DATE=08/01/031 TIME=10.28E LCHA={E}
{ }
.VSNA.....TOD=B6B61478 R0#9=00000000 00000C18 D9C5D7D3 0A69602C 0A6960EC 0A6FC588 8BC14F5B 00000000 00CC4B20 000000EE R14=0A69616
.....8 R15=00000000 EPCB=0A69602C FLGM=E0 DMAC=0AD09288 ARBA=00001004 UOWO=00000000 CCID=E0 CCNT=00000003 UBLK=00000000 KUBL
.....=00000000 DEDB=10 LKFP=0A6960EC PRGP=0A6960EC CLEV=0A6960EC MLTE=0A6960EC POPT=00 SGLS=01 CLOC=0AB38008 DMHR=0AA922C0 P
.....RBA=00001004 CRBA=00001008 NRBA=00001268 GRBA=00001268 XRBA=00001004 SGCD=01 PROF=0000 LEVL=01 KEYL=0B FLGA=88 FLGB=00
.....ACCK=000C KEYO=0008 SDBS=0AD09210 MLTE=0A6960EC SFRX=00 SFSX=00 SFWX=00 SFZX=00 SFMX=00 PREF=0006 SCNT=00000002 SWC1=40
.....SWC2=00 SWC3=00 LOPR=00 SNAP=0A696168 EPST=0A6FC040 SCVL=01 REOP=80 FDLN=0B FDOF=0008 DEDB=80 COMP=0A4998D0 STAT=
.....WCH=00 MODU=0AD74478 DATE=08/01/031 TIME=10.28E LCHA={E}
{ }

```

**Validating segment name:**

```

.SSR9.....TOD=B6B6147D R15=00000000 DATE=01/0310.2 #0#10=00000000 00000C18 D9C5D7D3 0A69602C 0A6960EC 0A6FC588 8BC14F5B 00000000 0
.....0CC4B20 000000EE 0A6FC040
.MRPL.....TOD=B6B6147D R0#9=00000000 00000C18 D9C5D7D3 0A69602C 0A6960EC 0AD4B68C 8BC14F5B 00000000 00CC4B20 0A5BE060 R14=8AD4B01
.....2 R15=0AD6AD40 EPCB=0A69602C FLGM=E0 DMAC=0AD09288 ARBA=00001004 UOWO=00000000 CCID=E0 CCNT=00000003 UBLK=00000000 KUBL
.....=00000000 DEDB=10 LKFP=0A6960EC PRGP=0A6960EC CLEV=0A6960EC MLTE=0A6960EC POPT=00 SGLS=01 CLOC=0AB38008 DMHR=0AA922C0 P
.....RBA=00001004 CRBA=00001008 NRBA=00001268 GRBA=00001268 XRBA=00001004 SGCD=01 PROF=0000 LEVL=01 KEYL=0B FLGA=88 FLGB=00
.....ACCK=000C KEYO=0008 SDBS=0AD09210 MLTE=0A6960EC SFRX=00 SFSX=00 SFWX=00 SFZX=00 SFMX=00 PREF=0006 SCNT=00000002 SWC1=40
.....SWC2=00 SWC3=00 LOPR=80 SNAP=0A696168 EPST=0A6FC040 SCVL=01 REOP=80 FDLN=0B FDOF=0008 DEDB=80 COMP=0A4998D0 STAT=
.....WCH=00 MODU=0AD6AD40 DATE=08/01/03P TIME=PQ7304 LCHA=49 AE}

```

**Replace Call Handler:**

```

...PI09.....TOD=B6B61482 WKAR=0A6FCC94 R0#F=00000000 0BC04F54 0AB38008 0A69602C 0A6960EC 0AD4B68C 0A6960EC 00000000 00CC4B20 0A5BE0
.....60 0A6FC040 00CC5B78 0AD678C0 0A5BE6F0 8AD6AE8A 0AD678C0 MODU=0AD678C0 DATE=08/01/03
...PI09.....TOD=B6B61483 WKAR=0A6FCC94 R0#F=0A499672 00000000 0AB38008 0A69602C 0A6960EC 00000000 0A499666 00000000 0000025A 0A5BE0
.....60 0A6FC040 00CC5B78 0AD678C0 0A5BE6F0 0A115556 00000000 MODU=0AD678C0 DATE=08/01/03

```

**Process I/O Area for Replace:**

```

...MUH1.....TOD=B6B61484 WKAR=0A6FCC94 R0#F=00000260 00000008 0000000B 0A69602C 0A6960EC 0AD4B68C 00000008 00000000 00000260 0AA922
.....C0 0A6FC040 00CC5B78 0AD737A8 0A5BE6F0 8AD6B0A2 0AD737A8 MODU=0AD737A8 DATE=08/01/031 TIME=10.28E LCHA={E}
{ }
...MUHE.....TOD=B6B61485 WKAR=0A6FCC94 R0#F=00000018 00000025 0A1150C9 0000022B 0AB3803D 0000022B 00000008 00000025 0AB38025 0AA922
.....C0 0A6FC040 00CC5B78 0AD73158 0A5BE738 8AD73850 0AD73158 MODU=0AD73158 DATE=08/01/031 TIME=10.28E LCHA={E}
{ }
...NUHE.....TOD=B6B61486 WKAR=0A6FCC94 R0#F=00000018 00000025 0000001C 0000022B 00000000 00000004 0000003C 00000001 0AB38025 0AA922
.....C0 0A6FC040 00CC5B78 0AD73158 0A5BE738 8AD73850 0AD73158 MODU=0AD73158 DATE=08/01/031 TIME=10.28E LCHA={E}
{ }
...MUH1.....TOD=B6B61487 WKAR=0A6FCC94 R0#F=00000018 00000025 0A1152F4 00000000 0AB38268 00000000 00000008 00000025 0AB38025 0AA922
.....C0 0A6FC040 00CC5B78 0AD737A8 0A5BE6F0 8AD73850 00000000 MODU=0AD737A8 DATE=08/01/031 TIME=10.28E LCHA={E}
{ }

```

**Record changes to buffer:**

```

.MRPL.....TOD=B6B61488 R0#9=00000260 00000008 0000000B 0A69602C 0A6960EC 0AD4B68C 00000008 00000000 00000260 0AA922C0 R14=8AD6B0A
.....2 R15=00000000 EPCB=0A69602C FLGM=E0 DMAC=0AD09288 ARBA=00001004 UOWO=00000000 CCID=E0 CCNT=00000003 UBLK=00000000 KUBL
.....=00000000 DEDB=10 LKFP=0A6960EC PRGP=0A6960EC CLEV=0A6960EC MLTE=0A6960EC POPT=00 SGLS=01 CLOC=0AB38008 DMHR=0AA922C0 P
.....RBA=00001004 CRBA=00001008 NRBA=00001268 GRBA=00001268 XRBA=00001004 SGCD=01 PROF=0000 LEVL=01 KEYL=0B FLGA=88 FLGB=00
.....ACCK=000C KEYO=0008 SDBS=0AD09210 MLTE=0A6960EC SFRX=00 SFSX=00 SFWX=00 SFZX=00 SFMX=00 PREF=0006 SCNT=00000002 SWC1=40
.....SWC2=00 SWC3=00 LOPR=80 SNAP=0A696168 EPST=0A6FC040 SCVL=01 REOP=80 FDLN=0B FDOF=0008 DEDB=80 COMP=0A4998D0 STAT=
.....WCH=00 MODU=0AD6AD40 DATE=08/01/03P TIME=PQ7304 LCHA=49 AE}

.SEG4.....TOD=B6B6148D R15=00000000 DATE=01/0310.2 #0#10=00000000 00000C18 0A52F35C 0A69602C 0A6960EC 0AD4B68C 8BC14F5B 00000000 0
.....0CC4B20 0A5BE060 0A6FC040
.NCL0.....R0#F=00000000 00000C18 0A52F35C 0A69602C 0A6960EC 8BC14F5B 8BC14F5B 00000000 00CC4B20 0A52F35C 0A6FC040 00CC5B78 0AD4AD
.....80 0A5BE660 00000030 00000000 EPCB=0A69602C CCID=E0 LCID=8C PRGP=0A6960EC LKFP=0A6960EC DBPC=0A52F35C STC= A LEV=01 S
.....FD=DEPARTMT
.....0080000100400010100010B8800000C00080AD0 STAS=

```

**REPL call ends:**



```
OPMV.....TOD=B6B61490 R0#F=00000000 00000000 D9C5D7D3 0A69602C 0BC04F54 0A5BE060 0A6FC491 00000000 00CC4B20 0BC04F54 0A6FC040 00
.....CC5B78 0ADBA930 0A5BE618 0BC04F54 00000000 IOAR=0A6FC491 IOAL=0000 IOAD=800A6FC4910A1155560A71C5800000000000000 EPST=0
.....A6FC040 MOVVP=0A6FC491 GETL=0000
SYN1.....TOD=B6B6149B R15=0AD35F68 DATE=05/0314.0 0#10=00CC4B20 0A5BE060 8A390E8E 0AE1EA34 0A5BE060 0A358698 0A9F1048 00000003 0
.....A35879C 0A5BE060 0A6FC040
```

**Begin synpoint:**

```
.SLOG.....TOD=B6B6149C R0#F=0A6FC040 00000001 00000000 0AE1EA34 0A5BE060 00000000 0A9F1048 00000003 0A35879C 0A5BE060 0A6FC040 00
.....CC5B78 0AD34168 0A5BE738 8AD3617E 0AD34168 MODU=0AD34168 DATE=08/05/03U TIME=UP9BDN LCHA=N0112
```

**Log 5950 CI updates:**

```
..SLGE.....TOD=B6B6149F R15=000007B4 DATE=05/0313.5 0#10=00000004 00001530 0A6FC040 00000010 00CC6EF0 00000070 0A6FC7F4 00CC4B20 0
.....A369920 0A5BE060 0A6FC040
..SLGE.....TOD=B6B614A0 R15=00000004 DATE=05/0313.5 0#10=3A53400A 000014C0 0A6FC040 0AB38000 0A6FD044 00000000 00000002 00CC4B20 0
.....AD09288 0AD091C0 0A6FC040
..SLGE.....TOD=B6B614A1 R15=000007B4 DATE=05/0313.5 0#10=00000000 00001498 0A6FC040 00007E99 0A88F31E 00000088 0A6FC7F4 00CC4B20 0
.....A369920 0A5BE060 0A6FC040
..SLGE.....TOD=B6B614A2 R15=00000000 DATE=05/0313.5 0#10=00000000 00001498 0A6FC040 00007E99 0A88F31E 00000088 0A6FC7F4 00CC4B20 0
.....A369920 00000000 0A6FC040
```

**Logger calls FP Logger Exit:**

```
.TLOG.....TOD=B6B614A3 R0#F=001D431E 0A5BE060 00000000 0A6FC27C 00000090 00000000 0A6FC7DC 0A6FC7DC 00000000 00CC4B20 0A6FC040 00
.....CC5B78 0AD34168 0A5BE738 8AD3486C 00000000 MODU=0AD34168 DATE=08/05/03U TIME=UP9BDN LCHA=N0112
TYN1.....TOD=B6B614A4 R15=00000000 DATE=05/0314.0 0#10=0A6FC040 00800001 00000000 0AE1EA34 305BE060 8A71C580 0A9F1048 00000003 0
.....A35879C 0A5BE060 0A6FC040
```

**End Phase I:**

```
.SYN2.....TOD=B6B614A4 R15=0AD36748 DATE=05/0314.0 0#10=00000000 0A5BE060 8A390E8E 00000000 0A5BE060 0A358698 00CC4B20 00000003 0
.....A35879C 0A5BE060 0A6FC040
```

**Start Phase II:**

```
..SLG2.....TOD=B6B614A5 R15=00000000 DATE=05/0314.0 0#10=00800001 00000000 0A5BE060 00000000 00000000 00000000 00CC4B20 00000003 0
.....A35879C 0A5BE060 0A6FC040
..SYP2.....TOD=B6B614A7 R0#F=00000000 00000000 00000000 00000000 0A5BE060 0A358698 00CC4B20 00000003 0A35879C 0A5BE060 0A6FC040 00
.....CC5B78 0AD370D0 0A5BE738 8AD36938 0AD370D0 MODU=0AD370D0 DATE=08/05/03P TIME=PQ6949 LCHA=94 AE}

...SPIX.....TOD=B6B614A8 R15=0AD35920 DATE=01/03
.....AD09288 00000000 0A6FC040
...XPIX.....TOD=B6B614A9 WKAR=0A6FCC94 R0#F=00000001 0A960CB0 00000000 0A6FC264 00000000 00000000 00CC4B20 0A960CB0 00000000 000000
.....00 0A6FC040 00CC5B78 0ADA6668 0A5BE810 8AD35AA2 0ADA6668 MODU=0ADA6668 DATE=08/05/03K TIME=KZC007 LCHA=77 0E}

...NPIX.....TOD=B6B614AE WKAR=0A6FCC94 R0#F=0A6FC040 00000000 8ADA7284 0ADA4C78 09E6A03C 00001000 FFFFE90 00000000 0AD09288 000000
.....00 0A6FC040 00CC5B78 0ADA6668 0A5BE810 8ADA4D58 00000000 MODU=0ADA6668 DATE=08/05/03K TIME=KZC007 LCHA=77 0E}

...NPIX.....TOD=B6B614AF WKAR=0A6FCC94 R0#F=00000001 00000000 00000000 0A6FC264 00000000 00000000 00CC4B20 00000000 00000000 000000
.....00 0A6FC040 00CC5B78 0AD35920 0A5BE7C8 8AD35AA2 0ADA6668 MODU=0AD35920 DATE=08/01/03
```

**Release locks:**

```
...TYP2.....TOD=B6B614B0 R0#F=00000000 00000000 00000000 00000000 0A5BE060 0A358698 00000000 00000003 0A35879C 0A5BE060 0A6FC040 00
.....CC5B78 0AD370D0 0A5BE738 8AD37168 00000000 MODU=0AD370D0 DATE=08/05/03P TIME=PQ6949 LCHA=94 AE}

...SHDQ.....TOD=B6B614B1 R15=0AD30B50 DATE=/01/03PQ7 0#10=00000000 00000000 00000000 0A5BE060 0A358698 00CC4B20 00000003 0
.....A35879C 0A5BE060 0A6FC040
...SHDX.....TOD=B6B614B2 R15=00000000 DATE=/01/03PQ7 0#10=00000000 00000000 00000000 0A6966A8 00000000 00000000 00CC4B20 00000003 0
.....A696018 0A5BE060 0A6FC040
...SDEQ.....TOD=B6B614B2 R0#F=00000000 00000000 00000000 00000000 0A5BE060 0A358698 00CC4B20 00000003 0A35879C 0A5BE060 0A6FC040 00
.....CC5B78 0AD2F3B8 0A5BE738 8AD369BC 0AD2F3B8 MODU=0AD2F3B8 DATE=08/01/031 TIME=10.31K LCHA=KZC0077 1
...BENQ.....TOD=B6B614B3 DATE=01/0310.1 FC=08 UDNA=00000000 UDSR=0000 QBLK=00 PARM=2003225F0002292773
.....

...RENO.....TOD=B6B614B4 R0#F=0A6FC040 0A6FC040 0A6FC4EC 00000000 0A5BE060 0A358698 00CC4B20 00000003 0A35879C 00000000 0A6FC040 00
.....CC5B78 0AD2DB78 0A5BE780 8AD2F582 00000000 MODU=0AD2DB78 DATE=08/01/031 TIME=10.10K LCHA=KZC0077 0
...TDEQ.....TOD=B6B614B5 R15=00000000 DATE=01/0310.3 0#10=0A6FC040 00000000 00000008 0A6966A8 00000000 00000001 00000000 00000003 0
.....A696018 00000000 0A6FC040
```

**Dequeue other resources:**

```
..TYN2.....TOD=B6B614B6 R15=8A71C580 DATE=05/0314.0 0#10=00000000 00800001 00000000 00000000 0A5BE060 0A358698 00CC4B20 00000003 0
.....A35879C 0A5BE060 0A6FC040
```

| **End Phase II:**

| ..SYN2.....TOD=B6B614B7 R15=0AD36748 DATE=05/0314.0 0#10=00000000 8A5BE060 00000000 0AE1EA34 0A5BE060 0A358698 00CC4B20 00000003 0  
| .....A35879C 0A5BE060 0A6FC040  
| ..TYN2.....TOD=B6B614B8 R15=0AD36748 DATE=05/0314.0 0#10=00000000 8A5BE060 00000000 0AE1EA34 0A5BE060 0A358698 00CC4B20 00000003 0  
| .....A35879C 0A5BE060 0A6FC040

|

---

## Chapter 12. MSC—Multiple Systems Coupling Service Aids

This section 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:

- “Multiple Systems Coupling Communication Task Trace” discusses an MSC communication task trace.
- “Multiple Systems Coupling Device-Dependent Module” provides a description of the various entry points in the device-dependent modules.
- “Multiple Systems Coupling Traces” on page 435 provides a description of MSC coupling traces.
- “Diagnosing Link Problems” on page 435 discusses the diagnosis of link problems.
- “Channel-to-Channel Access Method Trace Stack (LXB Trace)” on page 441 provides 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                    |

|             |                                          |
|-------------|------------------------------------------|
| <b>DM05</b> | Not used                                 |
| <b>DM06</b> | Not used                                 |
| <b>DM07</b> | Connect or disconnect the link           |
| <b>DM01</b> | 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.

Table 108 summarizes the MSC communication task trace.

*Table 108. Multiple Systems Coupling Communication Task Trace*

| Traced By | Entry Point | Function                                           | Trace Indent |
|-----------|-------------|----------------------------------------------------|--------------|
| DFSCMS00  | DFSCMA01    | Process Input                                      | AM01         |
| DFSCMS00  | DFSCMA02    |                                                    | AM02         |
| DFSCMS00  | DFSCMA03    | What's Next?                                       | AM03         |
| DFSCMS00  | DFSCMA05    |                                                    | AM05         |
| DFSCMS00  | DFSCMA06    | After Good Write                                   | AM06         |
| DFSCMS00  | DFSCMA08    | Wash Message                                       | AM08         |
| DFSCMS00  | DFSCMA09    | Generate Message                                   | AM09         |
| DFSCMS00  | DFSCIO10    | Quiesce Link                                       | AM10         |
| DFSCMS00  | DFSCIO12    | Wait for I/O or Message Enqueue                    | AM12         |
| DFSCMS00  | DFSCIOC0    | Get Work Buffer                                    | CM00         |
| DFSCMS00  | DFSCIOC0    | Reposition Queue Buffer                            | CM01         |
| DFSCMS00  | DFSCIOC0    | Get Next                                           | CM02         |
| DFSCMS00  | DFSCIOC0    | Dequeue Message                                    | CM03         |
| DFSCMS00  | DFSCIOC0    | Wash Output                                        | CM04         |
| DFSCMS00  | DFSCIOC0    | Find Output                                        | CM05         |
| DFSCMS00  | DFSCIOC0    | Get New Output                                     | CM06         |
| DFSCMS00  | DFSCIOC0    | Free Input Queue Buffer                            | CM07         |
| DFSCMS00  | DFSCIOC0    | Free Work Buffer                                   | CM08         |
| DFSCMS80  | DFSCMS80    | Abort Processing (First LTB)                       | MSS1         |
| DFSCMS80  | DFSCMS80    | Abort Processing (Second LTB)                      | MSS2         |
| DFSCMS81  | DFSCMS81    | Prior to DDM I/O                                   | DM01         |
| DFSCMS00  | DFSCIO03;06 | Write Setup                                        | DM01         |
| DFSCMS00  | DFSCIO00    | Write Interrupt                                    | DM02         |
| DFSCMS00  | DFSCIO01;03 | Read Setup                                         | DM03         |
| DFSCMS00  | DFSCIO00    | Read Interrupt                                     | DM04         |
| DFSCMS00  | DFSCIO00;03 | Connect/Disconnect I/O Interrupt                   | DM07         |
| DFSCMEI0  | DFSCMEI0    | Message Control/Error exit processing              | CMEI         |
| DFSCMEI0  | DFSCMEI0    | Before calling Message Control/Error exit DFSCMUX0 | CMEA         |
| DFSCMEI0  | DFSCMEI0    | After calling Message Control/Error exit DFSCMUX0  | CMEB         |

---

## Multiple Systems Coupling Traces

### 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).

The SYSID is increased to 2 bytes and it 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
```

KBLA (Knowledge-Based Log Analysis) provides MSC link performance analysis through an ISPF driven interface. KBLA uses IMS MSC log records to calculate the overall performance of each link that is subject to data traffic in the system.

There are two different report types that are generated:

### Summary

A report that contains the average response time, in milliseconds (msec), of the total number of send and receive data values for each link trace.

**Detail** A report that contains the individual response times, in milliseconds (msec), for every send and receive data value for each MSC link that has been traced.

The formatting of this record is only available if the IMS input log contains X'6701' records that are generated via the /TRACE SET ON LINK *link#* command. For more information, see the "MSC Link Performance Formatting Utility (DFSKMSC0)" section in the *IMS Version 9: Utilities Reference: System*.

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 using /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 and is 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.

*DM01 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.

Table 109 shows the significant fields in MSS1 and MSS2 records.

*Table 109. Significant Fields in MSS1 and MSS2 Records*

| Field | Description                                                                                                            |
|-------|------------------------------------------------------------------------------------------------------------------------|
| BSC   | POST code (first byte of LLB)<br>DECTYPE<br>DECFLAGS<br>DECERRST<br>DECRESPN<br>IOB<br>I/O buffers (data and response) |
| MTM   | POST code (first byte of LLB)<br>DECTYPE<br>I/O buffers (data and response)                                            |
| CTC   | POST code (first byte of LLB)<br>DECTYPE<br>IOSB<br>I/O buffers (data and response)<br>LBX                             |

Table 109. Significant Fields in MSS1 and MSS2 Records (continued)

| Field | Description                                                             |
|-------|-------------------------------------------------------------------------|
| VTAM  | POST code (first byte of LLB)<br>DECTYPE<br>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.

- | The following topics provide additional information:
- | • “DFSCMC00 (MSC Analyzer)” on page 442
- | • “DFSCMC50 (Shutdown Processing Routine)” on page 442

- | • “DFSCMC40 (Attention DIE Routine)”
- | • “DFSCNC40 (I/O Request DIE Routine)” on page 443
- | • “DFSCMC10 (Channel-End Appendage)” on page 443
- | • “DFSCMC10 (Abnormal-End Appendage)” on page 444
- | • “DFSCMC10 (Shutdown Appendage)” on page 444
- | • “MSC Routine Trace—BUFMSVID” on page 446

## 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**

| <b>Value</b> | <b>Meaning</b>                                                                          |
|--------------|-----------------------------------------------------------------------------------------|
| <b>0</b>     | Second entry into this routine was taken; nothing was done                              |
| <b>1</b>     | LXBCLIB switch was reset                                                                |
| <b>2</b>     | 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**

| <b>Value</b> | <b>Meaning</b>                                                                    |
|--------------|-----------------------------------------------------------------------------------|
| <b>0</b>     | Nothing was done                                                                  |
| <b>1</b>     | LXB was posted with STARTUP complete; the link is available for a WRITE operation |
| <b>2</b>     | LXB was posted with STARTUP complete; STARTUP message was received                |
| <b>3</b>     | During STARTUP processing, no-operation command was scheduled                     |
| <b>5</b>     | LXB was posted; message received                                                  |
| <b>6</b>     | LLB was posted; message received                                                  |
| <b>8</b>     | During STARTUP processing, control command was scheduled                          |
| <b>9</b>     | LLB was posted; an error occurred on message that was written                     |
| <b>A</b>     | LLB was posted; an error occurred on message that was received                    |
| <b>B</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 158 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 158 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 158. 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'** DFSCMC10 (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).

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 section describes diagnostic aids that help you analyze problems in DBRC. Included are:

- “Diagnosing from a RECON List”
- “RECON Record Types”
- “DBRC Internal Trace” on page 450
- “DBRC Internal Unformatted Trace Example” on page 462
- “DBRC External Trace” on page 469
- “DBRC API Return and Reason Codes” on page 472

---

### 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 9: 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 topic describes the content of the keys in the RECON records. To view the layout of the entire RECON record, see Table 110 on page 448. Consider these points as you examine the records:

- The RECON key size is 32 bytes.
- The last three bytes of the key are one of the following:
  - Reserved, and contain zeros.
  - First byte=0 and the last 2 bytes=key segment number (this one is added for unlimited RECON record).
- Time stamps have the following characteristics:
  - Time stamps are 12 bytes.
  - The symbolic UTC format is:  
YYYYDDDFHHMMSSTHMIJUAQQS  
An example of the UTC format is: 2004006F211432800000032D
  - DSPTIMES (DFSTIMES) contains time stamp structure information.

Table 110 on page 448 shows the RECON record types.

Table 110. RECON Record Types

| Common Name            | Part Name | List ID  | Release | Key Fields                                                                                   |
|------------------------|-----------|----------|---------|----------------------------------------------------------------------------------------------|
| RECON Header           | DSPRCNRC  | RECON    | R-1     | DBD: hex zeros<br>DDN: hex zeros<br>Type: X'01'<br>Time: hex zeros                           |
| RECON Header Extension | DSPRCR1   | *****    | R-3     | DBD: hex zeros<br>DDN: hex zeros<br>Type: X'01'<br>Time: X'00000000008'                      |
| Audit Trail Record     | DSPMUPHD  | *****    | 2.1     | DBD: hex zeros<br>DDN: hex zeros<br>Type: X'02'<br>Time: sequence number                     |
| RECON DMB Table Record | DSPRDMBT  | *****    | 9.1     | DBD: hex zeros<br>DDN: hex zeros<br>Type: X'03'<br>Time: hex zeros                           |
|                        |           |          |         | <b>Note:</b> Not listed in RECON Listing. An IDCAMS print will show the record if it exists. |
| PRILOG                 | DSPLOGRC  | PRILOG   | R-1     | DBD: hex zeros<br>DDN: hex zeros<br>Type: X'05'<br>Time: time stamp                          |
| Interim PRILOG         | DSPLOGRC  | IPRI     | R-2     | DBD: hex zeros<br>DDN: hex zeros<br>Type: X'06'<br>Time: time stamp                          |
| LOGALL                 | DSPLGARC  | LOGALL   | R-1     | DBD: hex zeros<br>DDN: hex zeros<br>Type: X'07'<br>Time: time stamp                          |
| SECLOG                 | DSPLOGRC  | SECLOG   | R-1     | DBD: hex zeros<br>DDN: hex zeros<br>Type: X'09'<br>Time: time stamp                          |
| Interim SECLOG         | DSPLOGRC  | ISEC     | R-2     | DBD: hex zeros<br>DDN: hex zeros<br>Type: X'0A'<br>Time: time stamp                          |
| PRISLDS                | DSPLOGRC  | PRISLD   | R-3     | DBD: X'FFFFFFFF00000043'<br>DDN: subsystem name<br>Type: X'43'<br>Time: time stamp           |
| PRITSLDS               | DSPLOGRC  | PRITSLDS | 5.0     | DBD: X'FFFFFFFF00000044'<br>DDN: subsystem name<br>Type: X'44'<br>Time: time stamp           |
| Interim PRISLDS        | DSPLOGRC  | IPRISL   | R-3     | DBD: X'FFFFFFFF00000045'<br>DDN: subsystem name<br>Type: X'45'<br>Time: time stamp           |
| Interim PRITSLDS       | DSPLOGRC  | IPRITSLD | 5.0     | DBD: X'FFFFFFFF00000046'<br>DDN: subsystem name<br>Type: X'46'<br>Time: time stamp           |

Table 110. RECON Record Types (continued)

| Common Name            | Part Name | List ID  | Release | Key Fields                                                                         |
|------------------------|-----------|----------|---------|------------------------------------------------------------------------------------|
| SECSLDS                | DSPLOGRC  | SECSLD   | R-3     | DBD: X'FFFFFFFF00000047'<br>DDN: subsystem name<br>Type: X'47'<br>Time: time stamp |
| SECTSLDS               | DSPLOGRC  | SECTSLDS | 5.0     | DBD: X'FFFFFFFF00000048'<br>DDN: subsystem name<br>Type: X'48'<br>Time: time stamp |
| Interim SECSLDS        | DSPLOGRC  | ISECSL   | R-3     | DBD: X'FFFFFFFF00000049'<br>DDN: subsystem name<br>Type: X'49'<br>Time: time stamp |
| Interim SECTSLDS       | DSPLOGRC  | ISECTSLD | 5.0     | DBD: X'FFFFFFFF00000050'<br>DDN: subsystem name<br>Type: X'50'<br>Time: time stamp |
| Change Accum Group     | DSPCAGRC  | CAGRP    | R-1     | DBD: hex zeros<br>DDN: CA group name<br>Type: X'0F'<br>Time: hex zeros             |
| Change Accum Execution | DSPCHGRC  | CA       | R-1     | DBD: hex zeros<br>DDN: CA group name<br>Type: X'11'<br>Time: time stamp            |
| DBDS Group             | DSPDGRC   | DBDSGRP  | 2.1     | DBD: X'0000000000000007'<br>DDN: DBDS group name<br>Type: X'16'<br>Time: hex zeros |
| Database Header        | DSPDBHRC  | DB       | R-2     | DBD: DBD name<br>DDN: DDN name<br>Type: X'18'<br>Time: hex zeros                   |
| Partition              | DSPPTNRC  | DB       | 7.1     | DBD: DBD name<br>DDN: Partition name<br>Type: X'19'<br>Time: hex zeros             |
| Database Data Set      | DSPDSHRC  | DBDS     | R-1     | DBD: DBD name<br>DDN: DDN name<br>Type: X'20'<br>Time: hex zeros                   |
| Area Recovery          | DSPDSHRC  | DBDS     | R-3     | DBD: DBD name<br>DDN: area name<br>Type: X'20'<br>Time: hex zeros                  |
| Area Auth              | DSPDBHRC  | DBDS     | R-3     | DBD: DBD name<br>DDN: area name<br>Type: X'21'<br>Time: hex zeros                  |
| ALLOC                  | DSPALLRC  | ALLOC    | R-1     | DBD: DBD name<br>DDN: DDN or area name<br>Type: X'28'<br>Time: time stamp          |
| Image Copy             | DSPIMGRC  | IMAGE    | R-1     | DBD: DBD name<br>DDN: DDN or area name<br>Type: X'2D'<br>Time: time stamp          |

Table 110. RECON Record Types (continued)

| Common Name            | Part Name | List ID | Release | Key Fields                                                                          |
|------------------------|-----------|---------|---------|-------------------------------------------------------------------------------------|
| Reorg                  | DSPRRGRC  | REORG   | R-2     | DBD: DBD name<br>DDN: DDN or area name<br>Type: X'32'<br>Time: time stamp           |
| Recovery               | DSPRCVRC  | RECOV   | R-1     | DBD: DBD name<br>DDN: DDN or area name<br>Type: X'37'<br>Time: time stamp           |
| Backout                | DSPBKORC  | BACKOUT | 4.1     | DBD: X'FFFFFFFF00000035'<br>DDN: subsystem name<br>Type: X'35'<br>Time: hex zeros   |
| Global Service Group   | DSPGSRC   | GSG     | 5.0     | DBD: X'FFFFFFFFFFFFFF0000'<br>DDN: subsystem name<br>Type: X'3A'<br>Time: hex zeros |
| Tracking Subsystem     | DSPSSRC   | SSYS    | 5.0     | DBD: X'FFFFFFFF0000003E'<br>DDN: subsystem name<br>Type: X'3E'<br>Time: hex zeros   |
| Subsystem              | DSPSSRC   | SSYS    | R-2     | DBD: X'FFFFFFFFFFFFFFFFF'<br>DDN: subsystem name<br>Type: X'3F'<br>Time: hex zeros  |
| Available CA Execution | DSPCHGRC  | CA      | R-1     | DBD: hex zeros<br>DDN: hex zeros<br>Type: X'51'<br>Time: time stamp                 |
| PRIOLDS                | DSPOLDRC  | PRIOLD  | R-3     | DBD: X'FFFFFFFF00000053'<br>DDN: subsystem name<br>Type: X'53'<br>Time: hex zeros   |
| Interim PRIOLDS        | DSPOLDRC  | IPRIOL  | R-3     | DBD: X'FFFFFFFF00000055'<br>DDN: subsystem name<br>Type: X'55'<br>Time: hex zeros   |
| SECOLDS                | DSPOLDRC  | SECOLD  | R-3     | DBD: X'FFFFFFFF00000057'<br>DDN: subsystem name<br>Type: X'57'<br>Time: hex zeros   |
| Interim SECOLDS        | DSPOLDRC  | ISECOL  | R-3     | DBD: X'FFFFFFFF00000059'<br>DDN: subsystem name<br>Type: X'59'<br>Time: hex zeros   |
| Available Image Copy   | DSPIMGRC  | IMAGE   | R-1     | DBD: DBD name<br>DDN: DDN or area name<br>Type: X'6D'<br>Time: hex zeros            |

## 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



## 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 DSPxxxx module in the Save Area trace of the dump. For most DSPxxxx 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 Internal Unformatted Trace Example" on page 462.

## 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 469.

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 documented in the following:

- "Trace Header Record"
- "Module Call, Module Return, and DSPSTACK Trace Entries" on page 453
- "BGNCABN0, DSPCABN0, BGNRETRY, DSPCRTR0, and CRTR0XIT Trace Entries" on page 455
- "DSPURI00 Trace Entries" on page 457

## Trace Header Record

Figure 159 on page 453 shows the DBRC trace header record.

```
|      words 0-1 – Identifier TRACETBL
|      word 2 – Length of the trace
|      word 3 – Count of trace calls made
|      word 4 – Beginning of trace table
|      word 5 – End of trace table
|      word 6 – Next entry to update
|      word 7 – Double word alignment
```

Figure 159. 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 160 on page 454 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.

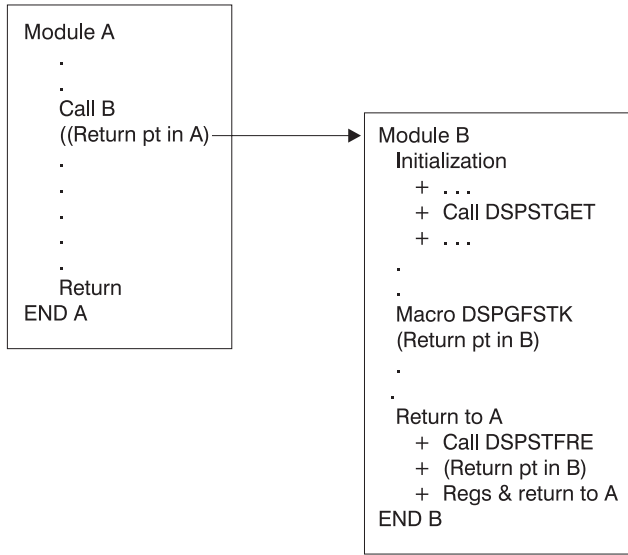


Figure 160. DBRC Trace Processing Flow

Figure 161, Figure 162 on page 455, and Figure 163 on page 455 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 160.

Figure 161 shows a 1-line trace entry that is produced when module A calls module B. A 1-line trace entry that is produced when module B calls DSPSTGET to obtain initial work space storage after being called by module A.

- words 0-1 – Identifier which consists of:
  - An arrow ("") indicating that the module is being called.
  - The nesting level of module B. 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.
- word 2 – Offset in module A of call to module B
- word 3 – Entry point address of module B
- word 4 – Save area address of the calling module (A)
- word 5 – Beginning address of the temporary storage obtained for module B (B's save area address)
- words 6-7 – Trace time stamp

Figure 161. One-Line Trace Entry Produced When Module A Calls Module B

Figure 162 on page 455 shows a 1-line trace entry that is produced when module B calls DSPSTFRE to release all of its temporary storage before returning to module A.



```

|
| words 0-1 – Identifier which consists of:
|           - A left arrow (")indicating that the module is
|             returning.
|           - The nesting level of module A. Nesting levels
|             are shown in one or two decimal digits up to
|             99 (nesting level 0 is DSPUIIN00).
|           - The last five characters of the module name
|             returning.
|
| word 2 – Offset in module A to which module B returns
|
| word 3 – Offfest in module B where it returns to module A
|
| word 4 – Save area address of module A that called module B
|
| word 5 – Beginning address of the temporary storage being
|           released for module B by module DSPSTFRE
|
| words 6-7 - Trace time stamp

```

*Figure 162. One-Line Trace Entry Produced When Module B Returns to Module A.*

Figure 163 shows a 1-line trace entry that is produced when module B issues macro DSPGFSTK, which calls DSPSTGET to obtain additional temporary storage.

```

|
| words 0-1 – Identifier DSPSTACK
|
| word 2 – Return point address in the module B to which
|           DSPSTGET returns after acquiring additional
|           temporary storage for the module.
|
| word 3 – Entry point address of module B
|
| word 4 – Save area address of the module (B)
|
| word 5 – Beginning address of the additional temporary
|           storage obtained for module B
|
| words 6-7 - Trace time stamp

```

*Figure 163. DSPSTACK Trace Entry*

## **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 165, Figure 166, Figure 167 on page 457, and Figure 168 on page 457 show the layout of the entries issued from BGNCABN0, DSPCABN0, and DSPCRTR0.

```

|
| words 0-1 – Identifier BGNCABN0
|
| word 2 – A(DSPGDB)
|
| words 3-5 – Zeros
|
| words 6-7 - Time stamp
|

```

This is normally followed by either DSPCABN0 or a BGNRETRY entry.

*Figure 164. BGNCABN0 Trace Entry*

Figure 165 shows DBRC terminated because of an unrecoverable error.

```

|
| words 0-1 – Identifier DSPCABN0
|
| word 2 – A(DSPGDB)
|
| words 3-5 – Zeros
|
| words 6-7 - Time stamp
|

```

This is the last logical entry in the trace table.

*Figure 165. DSPCABN0 Trace Entry*

Figure 166 shows DBRC recovered from an abend condition and is beginning to execute a retry sequence of code.

```

|
| words 0-1 – Identifier BGNRETRY
|
| word 2 – A(DSPGDB)
|
| words 3-5 – Zeros
|
| words 6-7 - Time stamp
|

```

*Figure 166. BGNRETRY Trace Entry*

Figure 167 on page 457 shows the router made a trace call before passing control to the next DBRC routine scheduled to process the request identified by a DFSBRLSB.

Line 1:  
 words 0-1 – Identifier BGNRETRY  
 word 2 – A(DSPGDB)  
 words 3-5 – Zeros  
 words 6-7 – Time stamp

Line 2:  
 word 0 – Address of BLBPRNT field in DFSBRLSB  
 words 2-7 – Data from DFSBRLSB (next 60 bytes after field BRLBPRNT)

Line 3:  
 words 0-7 – Data from DFSBRLSB (continued from previous line)

Figure 167. DSPCRTR0 Trace Entry

Figure 168 shows the function requested in the DSPCRTR0 trace entry completed.

Line 1:  
 words 0-1 – Identifier CRTR0XIT  
 word 2 – A(DFSBRLSB)  
 words 3-5 – Data from DFSBRLSB: function flags, exit flags, address of DSPGDB. (These are the same fields displayed in the DSPCRTR0 entry, but they might have been modified by the request.)  
 words 6-7 – Time stamp

Line 2:  
 words 0-4 – DFSBRLSB prefix  
 words 4-7 – First 44 bytes of DFSBRLSB

Line 3:  
 words 0-7 – DFSBRLSB (continued from previous line)

Figure 168. CRTR0XIT Trace Entry

## 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 111 shows the 8-character identifier and block-area pointer for each call.

Table 111. 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 169 on page 459.) |
| 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.                                                                         |

Table 111. Calls to the Trace Routine in DSPURI00 (continued)

| 8-Character Identifier | Block-Area Pointer | Explanation                                                                                                                                                                                                                                                                                                                                                                                                                |
|------------------------|--------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| GETFEED                | FILRESLT(I)        | After DSPURI00 issues an I/O request, the GETFEED procedure is called to trace specific information related to the I/O operation. Some of this information comes from DSPVFILE, some from the VSAM RPL, some from the record key and some from the I/O parameter block, DSPIOPAR. In addition, the RPL request is translated into a character printable code that describes the I/O operation. See Figure 170 on page 460. |
| CLOSER1                | FILRESLT(I)        | DSPURI00 starts a true close of the RECON data set.                                                                                                                                                                                                                                                                                                                                                                        |
| 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 171 on page 462.)                                                                                                                                                                                                                                                                                               |

**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 169 on page 459, Figure 170 on page 460, and Figure 171 on page 462 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 information about the I/O operation. 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.

Line 1:  
 words 0-1 – DSPURI00  
 word 2 – GDB address  
 words 3-5 – Binary zeros  
 words 6-7 – Time stamp

Line 2:  
 words 0-1 – MODIRCAR  
 word 2 – c1c2  
 word 3 – Func  
 words 4-7 – 16-byte entry message

Line 3:  
 words 0-5 – Key, blank, or repl ddname (key area)  
 word 6 – addr  
 word 7 – leng

Figure 169. DSPURI00 Entry Trace Entry

**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

```

Line 1:
  words 0-1 – GETFEED
  word 2 – DSPGDBA
  words 3-5 – Binary zeros
  words 6-7 – Time stamp

Line 2:
  word 0 – RPLFDBWD
  word 1 – FILLRECL
  word 2 – FILNEWCA
  word 3 – FILNEWEX
  word 4 – FILCICNT
  word 5 – FILCACNT
  word 6 – FILEXCNT
  word 7 – FILMAX

Line 3:
  word 0 – FILCISZ
  word 1 – bytes 1-2 – FILFLAGS
             bytes 3-4 – FILOPERR
  word 2 – FILBUFPT
  word 3 – FILRCDPT
  word 4 – FILRCDLN
  word 5 – bytes 1-2 – SEGMENT NUMBER
             byte 3 – RECON NUMBER
             byte 4 – RPLREQ
  word 6 bytes 1-4 and word 7 bytes 1-2 – PRINTABLE
   RPLREQ
  word 7 – bytes 3-4 – NOT USED

```

Figure 170. GETFEED Trace Entry for One RECON

|                          |                                                           |
|--------------------------|-----------------------------------------------------------|
| <b>dspgdba</b>           | Address of the DSPGDB                                     |
| <b>time stamp</b>        | Trace time stamp                                          |
| <b>RPLFDBWD</b>          | RPL feedback word                                         |
| <b>FILLRECL</b>          | Logical record length                                     |
| <b>FILNEWCA</b>          | Starting high-used relative byte address (RBA)            |
| <b>FILNEWEX</b>          | Starting high-allocated RBA                               |
| <b>FILCICNT</b>          | RECON changed counter value                               |
| <b>FILCACNT</b>          | Current high-used RBA                                     |
| <b>FILEXCNT</b>          | Current high-allocated RBA                                |
| <b>FILMAX</b>            | VSAM maximum record size                                  |
| <b>FILCISZ</b>           | Data control interval (CI) size                           |
| <b>FILFLAGS</b>          | RECON processing status flags (open, reserved, empty)     |
| <b>FILOPERR</b>          | Open SVC reason code if RC is not 0                       |
| <b>FILBUFPT</b>          | Pointer to header record buffer                           |
| <b>FILRCDPT</b>          | Pointer to the record in the VSAM I/O buffer or user area |
| <b>FILRCDLN</b>          | Length of record                                          |
| <b>SEGMENT NUMBER</b>    |                                                           |
|                          | Record segment number                                     |
| <b>RECON COPY NUMBER</b> |                                                           |
|                          | Recon number used in this request                         |

**RPLREQ** RPL request type

**RPL REQUEST PRINTABLE CODE**

English word that is later translated into a printable code used to make a request to VSAM

**RPL REQ PRINTABLE CODE**

This is translation of the RPLREQ field into a printable code that is close to being the English word for the request made to VSAM.

Table 112 shows the translated RPLREQ printable codes.

*Table 112. Translated RPLREQ Printable Codes*

| Printable Code | HEX | RPLEQ           | RPL Request             |
|----------------|-----|-----------------|-------------------------|
| GET            | 00  | GET             | Retrieve a record       |
| PUT            | 01  | PUT             | Write a record          |
| CHECK          | 02  | CHECK           | Wait for completion     |
| POINT          | 03  | POINT           | Position for access     |
| ENDREQ         | 04  | ENDREQ          | Terminate a request     |
| ERASE          | 05  | ERASE           | Delete a record         |
| VERIFY         | 06  | VERIFY          | Synchronize end of data |
| *****          | 07  | Not used        | Not used                |
| DATPRE         | 08  | DATA PREFORMAT  |                         |
| IDXPRES        | 09  | INDEX PREFORMAT |                         |
| FORCIO         | 0A  | Force I/O       |                         |
| GETIX          | 0B  | GET INDEX       |                         |
| PUTIX          | 0C  | PUT INDEX       |                         |
| SCHBFR         | 0D  | SCHBFR          | Search Buffer           |
| MRKBFR         | 0E  | MRKBFR          | Mark Buffer             |
| WRTBFR         | 0F  | WRTBFR          | Write Buffer            |
| CNVTAD         | 10  | CNVTAD          |                         |
| MNTACQ         | 11  | MNTACQ          |                         |
| ACQRNG         | 12  | ACQRANGE        |                         |
| TRMRPL         | 13  | TERMRPL         |                         |
| VERREF         | 14  | VERIFY REFRESH  |                         |

```

Line 1:
  words 0-1 – DSPURI00
  word 2 – GDB address
  words 3-5 – Binary zeros
  words 6-7 – Time stamp

Line 2:
  words 0-1 – ENDICAR
  word 2 – c1c2
  word 3 – Func
  words 4-7 – 16-byte entry message

Line 3:
  words 0-5 – Key, blank, or repl ddname (key area)
  word 6 – addr
  word 7 – lncr

```

Figure 171. DSPURI00 Exit Trace Entry

|                             |                                                                                                                                                                                  |
|-----------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>time stamp</b>           | Trace time stamp                                                                                                                                                                 |
| <b>c1c2</b>                 | The DD statement number (1-3) of the copy 1 and copy 2 RECON, if any, on exit from DSPURI00                                                                                      |
| <b>func</b>                 | Function and option bits received from caller in DSPIRCAR                                                                                                                        |
| <b>16-byte exit message</b> | 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 |
| <b>key area</b>             | For a successful locate request, contains the 32-byte key of the RECON record returned to caller. Otherwise, contains a repeat of MODIRCAR contents                              |
| <b>addr</b>                 | Address of the record found for a successful locate. Otherwise, 0                                                                                                                |
| <b>lncr</b>                 | Length of the record found for a successful locate, or the return code to be passed back to the module that called DSPURI00                                                      |

---

## DBRC Internal Unformatted Trace Example

```

| The following example shows module-call, module return entries, DSPURI00 trace entries, and other
| entries (DSPCABN0, BGNCABN0, DSPCRTR0, CRTR0XIT). Toward the end of the trace, the DSPCABN0
| trace entry indicates that DBRC was terminated because of an unrecoverable error.
|

```



|                                           |          |          |          |          |          |          |          |          |                                     |
|-------------------------------------------|----------|----------|----------|----------|----------|----------|----------|----------|-------------------------------------|
| 0BA8A700                                  | E3D9C1C3 | C5E3C2D3 | 00025900 | 00000A0A | 0BA8A720 | 0BAAFF20 | 0BAABAA0 | 0BAAFFA0 | *TRACETBL.....yx.....*              |
| 0BA8A720                                  | 606EF1E3 | C9D4C5F0 | 8BA85500 | 0BA35B18 | 0BA01A30 | 0BA88010 | 03274F21 | 25566338 | *->1TIME0.y...t\$.y... .....*       |
| 0BA8A740                                  | F04C60E3 | C9D4C5F0 | 0BA85500 | 000010F4 | 0BA01A30 | 0BA88010 | 03274F21 | 25566338 | *0<-TIME0.y....4....y... .....*     |
| 0BA8A760                                  | 606EF1E4 | D9C9F0F0 | 8BA85674 | 0BA4AE18 | 0BA01A30 | 0BA88010 | 03274F21 | 25566339 | *->1URI00.y...u....y... .....*      |
| 0BA8A780                                  | C4E2D7E4 | D9C9F0F0 | 0BA86728 | 00000000 | 00000000 | 00000000 | 03274F21 | 25566339 | *DSPURI00.y..... .....*             |
| 0BA8A7A0                                  | D4D6C4C9 | D9C3C1D9 | 404001D8 | D6000600 | D7C8E8E2 | C9C3C1D3 | 40D6D7C5 | D5404040 | *MODIRCAR .QO...PHYSICAL OPEN *     |
| 0BA8A7C0                                  | 40404040 | 40404040 | 40404040 | 40404040 | 40404040 | 40404040 | 00000000 | 00000000 | *                                   |
| 0BA8A7E0                                  | 606EF2E4 | C3D7F4F0 | 00000A86 | 0BA826F8 | 0BA88010 | 0BA88958 | 03274F21 | 25566340 | *->2UCP40...f.y.8.y...yi... .....*  |
| 0BA8A800                                  | F14C60E4 | C3D7F4F0 | 0BA4B89E | 00000B4E | 0BA88010 | 0BA88958 | 03274F21 | 25566340 | *1<-UCP40.u....+y...yi... .....*    |
| 0BA8A820                                  | 606EF2E4 | D9C9F1F0 | 00000F92 | 0BA50210 | 0BA88010 | 0BA88958 | 03274F21 | 25566340 | *->2URI10...k.v...y...yi... .....*  |
| 0BA8A840                                  | 606EF3E4 | D9C9F2F0 | 0000020A | 0BA522A8 | 0BA88958 | 0BA88D78 | 03274F21 | 25566340 | *->3URI20....v.y.yi...y... .....*   |
| 0BA8A860                                  | 606EF4E4 | C1D3D3F0 | 00000260 | 0BA81E00 | 0BA88D78 | 0BA89088 | 03274F21 | 25566340 | *->4UALL0...-.y...y...y.h... .....* |
| 0BA8A880                                  | F34C60E4 | C1D3D3F0 | 0BA52508 | 000001D8 | 0BA88D78 | 0BA89088 | 03274F21 | 25567076 | *3<-UALL0.v.....Q.y...y.h... .....* |
| 0BA8A8A0                                  | 606EF4E4 | C1D3D3F0 | 00000260 | 0BA81E00 | 0BA88D78 | 0BA89088 | 03274F21 | 25567077 | *->4UALL0...-.y...y...y.h... .....* |
| 0BA8A8C0                                  | F34C60E4 | C1D3D3F0 | 0BA52508 | 000001D8 | 0BA88D78 | 0BA89088 | 03274F21 | 25567837 | *3<-UALL0.v.....Q.y...y.h... .....* |
| 0BA8A8E0                                  | 606EF4E4 | C1D3D3F0 | 00000260 | 0BA81E00 | 0BA88D78 | 0BA89088 | 03274F21 | 25567837 | *->4UALL0...-.y...y...y.h... .....* |
| 0BA8A900                                  | F34C60E4 | C1D3D3F0 | 0BA52508 | 000001D8 | 0BA88D78 | 0BA89088 | 03274F21 | 25568557 | *3<-UALL0.v.....Q.y...y.h... e.e.*  |
| 0BA8A920                                  | F24C60E4 | D9C9F2F0 | 0BA5041A | 00000224 | 0BA88958 | 0BA88D78 | 03274F21 | 25568557 | *2<-URI20.v.....yi...y... e.e.*     |
| 0BA8A940                                  | 606EF3D9 | D3C9F0F0 | 0000047E | 0BA22E18 | 0BA88958 | 0BA88D78 | 03274F21 | 25568557 | *->3RLI00...=.s...yi...y... e.e.*   |
| 0BA8A960                                  | 606EF4D9 | D3C1E4F0 | 000002FC | 0BA22838 | 0BA88D78 | 0BA89280 | 03274F21 | 25568557 | *->4RLAU0...-.y...y...y.k... e.e.*  |
| 0BA8A980                                  | F34C60D9 | D3C1E4F0 | 0BA23114 | 000004A6 | 0BA88D78 | 0BA89280 | 03274F21 | 25570461 | *3<-RLAU0.s....w.y...y.k... .../*   |
| 0BA8A9A0                                  | F24C60D9 | D3C9F0F0 | 0BA5068E | 00000B88 | 0BA88958 | 0BA88D78 | 03274F21 | 25570461 | *2<-RLI00.v.....y.yi...y... .../*   |
| 0BA8A9C0                                  | 606EF3D9 | E2E5F0F0 | 000004CA | 0BA80FA0 | 0BA88958 | 0BA88D78 | 03274F21 | 25570461 | *->3RSV00....y...yi...y... .../*    |
| 0BA8A9E0                                  | F24C60D9 | E2E5F0F0 | 0BA506D9 | 0000022A | 0BA88958 | 0BA88D78 | 03274F21 | 25570462 | *2<-RSV00...-.s...y...y.k... e.e.*  |
| 0BA8AA00                                  | 606EF3E4 | D9C9F1F0 | 0000050A | 0BA50234 | 0BA88958 | 0BA88D78 | 03274F21 | 25570462 | *->3URI10....v...yi...y... .....*   |
| 0BA8AA20                                  | D6D7C5D5 | C5D9F140 | 0BA86728 | 00000000 | 00000000 | 00000000 | 03274F21 | 25570462 | *OPENER1 .y..... .....*             |
| 0BA8AA40                                  | 00000000 | 00000000 | 00000000 | 00000000 | 00000000 | 00000000 | 00000000 | 00000000 | *..... .....*                       |
| LINES 0BA8AA60-0BA8AA7F SAME AS THE ABOVE |          |          |          |          |          |          |          |          |                                     |
| 0BA8AA80                                  | 606EF4E4 | D9C9F1F0 | 00000D7C | 0BA5023A | 0BA88D78 | 0BA89198 | 03274F21 | 25570680 | *->4URI10...@.v...y...yjq... .....* |
| 0BA8AAA0                                  | F34C60E4 | D9C9F1F0 | 0BA50FB0 | 00001162 | 0BA88D78 | 0BA89198 | 03274F21 | 25570683 | *3<-URI10.v.....y...yjq... .....c*  |
| 0BA8AAC0                                  | D6D7C5D5 | C5D9F240 | 0BA86728 | 00000000 | 00000000 | 00000000 | 03274F21 | 25570684 | *OPENER2 .y..... .....d*            |
| 0BA8AAE0                                  | 00000000 | 00000000 | 00000000 | 00000000 | 00000000 | 00024000 | 00024000 | 00023000 | *..... .....*                       |
| 0BA8AB00                                  | 00002000 | 00000020 | 00000000 | 00000000 | 00000000 | 00000000 | 00000000 | 00000012 | *..... .....*                       |
| 0BA8AB20                                  | F24C60E4 | D9C9F1F0 | 0BA5071A | 00000FF0 | 0BA88958 | 0BA88D78 | 03274F21 | 25570684 | *2<-URI10.v.....0.yi...y... .....d* |
| 0BA8AB40                                  | 606EF3E4 | D9C9F1F0 | 0000050A | 0BA50234 | 0BA88958 | 0BA88D78 | 03274F21 | 25570684 | *->3URI10....v...yi...y... .....d*  |
| 0BA8AB60                                  | D6D7C5D5 | C5D9F140 | 0BA86728 | 00000000 | 00000000 | 00000000 | 03274F21 | 25570684 | *OPENER1 .y..... .....d*            |
| 0BA8AB80                                  | 00000000 | 00000000 | 00000000 | 00000000 | 00000000 | 00000000 | 00000000 | 00000000 | *..... .....*                       |
| LINES 0BA8ABA0-0BA8ABBF SAME AS THE ABOVE |          |          |          |          |          |          |          |          |                                     |
| 0BA8ABC0                                  | 606EF4E4 | D9C9F1F0 | 00000D7C | 0BA5023A | 0BA88D78 | 0BA89198 | 03274F21 | 25570911 | *->4URI10...@.v...y...yjq... .....* |
| 0BA8ABE0                                  | F34C60E4 | D9C9F1F0 | 0BA50FB0 | 00001162 | 0BA88D78 | 0BA89198 | 03274F21 | 25570915 | *3<-URI10.v.....y...yjq... .....*   |
| 0BA8AC00                                  | D6D7C5D5 | C5D9F240 | 0BA86728 | 00000000 | 00000000 | 00000000 | 03274F21 | 25570915 | *OPENER2 .y..... .....*             |
| 0BA8AC20                                  | 00000000 | 00000000 | 00000000 | 00000000 | 00000000 | 00024000 | 00024000 | 00023000 | *..... .....*                       |
| 0BA8AC40                                  | 00002000 | 00000020 | 00000000 | 00000000 | 00000000 | 00000000 | 00000000 | 00000012 | *..... .....*                       |
| 0BA8AC60                                  | F24C60E4 | D9C9F1F0 | 0BA5071A | 00000FF0 | 0BA88958 | 0BA88D78 | 03274F21 | 25570916 | *2<-URI10.v.....0.yi...y... .....*  |
| 0BA8AC80                                  | 606EF3E4 | D9C9F1F0 | 0000050A | 0BA50234 | 0BA88958 | 0BA88D78 | 03274F21 | 25570916 | *->3URI10....v...yi...y... .....*   |
| 0BA8ACA0                                  | D6D7C5D5 | C5D9F140 | 0BA86728 | 00000000 | 00000000 | 00000000 | 03274F21 | 25570916 | *OPENER1 .y..... .....*             |
| 0BA8ACC0                                  | 00000000 | 00000000 | 00000000 | 00000000 | 00000000 | 00000000 | 00000000 | 00000000 | *..... .....*                       |
| LINES 0BA8ACE0-0BA8ACFF SAME AS THE ABOVE |          |          |          |          |          |          |          |          |                                     |
| 0BA8AD00                                  | D6D7C5D5 | C5D9F240 | 0BA86728 | 00000000 | 00000000 | 00000000 | 03274F21 | 25571270 | *OPENER2 .y..... .....*             |
| 0BA8AD20                                  | 00000000 | 00000000 | 00000000 | 00000000 | 00000000 | 00024000 | 00024000 | 00023000 | *..... .....*                       |
| 0BA8AD40                                  | 00002000 | 00000020 | 00000000 | 00000000 | 00000000 | 00000000 | 00000000 | 00000012 | *..... .....*                       |
| 0BA8AD60                                  | F24C60E4 | D9C9F1F0 | 0BA5071A | 00000FF0 | 0BA88958 | 0BA88D78 | 03274F21 | 25571271 | *2<-URI10.v.....0.yi...y... .....*  |
| 0BA8AD80                                  | 606EF3E4 | D9C9F2F0 | 000006AE | 0BA522CC | 0BA88958 | 0BA88D78 | 03274F21 | 25571276 | *->3URI20....v...yi...y... .....*   |
| 0BA8ADA0                                  | 606EF4E4 | C1D3D3F0 | 00000520 | 0BA81E00 | 0BA88D78 | 0BA89088 | 03274F21 | 25571451 | *->4UALL0...-.y...y...y.h... .....* |
| 0BA8ADC0                                  | F34C60E4 | C1D3D3F0 | 0BA527EC | 000002AA | 0BA88D78 | 0BA89088 | 03274F21 | 25571459 | *3<-UALL0.v.....y...y.h... .....*   |
| 0BA8ADE0                                  | F24C60E4 | D9C9F2F0 | 0BA508BE | 0000054A | 0BA88958 | 0BA88D78 | 03274F21 | 25571459 | *2<-URI20.v.....yi...y... .....*    |
| 0BA8AE00                                  | 606EF3C4 | C5D8F0F0 | 000006CC | 0BA0DA68 | 0BA88958 | 0BA88D78 | 03274F21 | 25571459 | *->3DEQ00.....yi...y... .....*      |
| 0BA8AE20                                  | F24C60C4 | C5D8F0F0 | 0BA508DC | 000007EE | 0BA88958 | 0BA88D78 | 03274F21 | 25571459 | *2<-DEQ00.v.....yi...y... .....*    |
| 0BA8AE40                                  | 606EF3E4 | D9C9F1F0 | 00001C16 | 0BA50234 | 0BA88958 | 0BA88D78 | 03274F21 | 25571640 | *->3URI10....v...yi...y... .....*   |
| 0BA8AE60                                  | D6D7C5D5 | C5D9F140 | 0BA86728 | 00000000 | 00000000 | 00000000 | 03274F21 | 25571640 | *OPENER1 .y..... .....*             |
| 0BA8AE80                                  | 00000000 | 00000000 | 00024000 | 00024000 | 20032806 | 00024000 | 00024000 | 00023000 | *..... .....*                       |
| 0BA8AEA0                                  | 00002000 | 00000020 | 00000000 | 00000000 | 00000000 | 00000000 | 00000000 | 00000012 | *..... .....*                       |
| 0BA8AEC0                                  | 606EF4E4 | D9C9F1F0 | 00000D7C | 0BA5023A | 0BA88D78 | 0BA89198 | 03274F21 | 25571886 | *->4URI10...@.v...y...yjq... ...f*  |
| 0BA8AEE0                                  | F34C60E4 | D9C9F1F0 | 0BA50FB0 | 00001162 | 0BA88D78 | 0BA89198 | 03274F21 | 25571892 | *3<-URI10.v.....y...yjq... ...k*    |
| 0BA8AF00                                  | D6D7C5D5 | C5D9F240 | 0BA86728 | 00000000 | 00000000 | 00000000 | 03274F21 | 25571892 | *OPENER2 .y..... ...k*              |
| 0BA8AF20                                  | 00000000 | 00000000 | 00024000 | 00024000 | 20032806 | 00024000 | 00024000 | 00023000 | *..... ...k*                        |
| 0BA8AF40                                  | 00002000 | 00000020 | 00000000 | 00000000 | 00000000 | 00000000 | 00000000 | 00000012 | *..... .....*                       |
| 0BA8AF60                                  | F24C60E4 | D9C9F1F0 | 0BA51E26 | 00000FF0 | 0BA88958 | 0BA88D78 | 03274F21 | 25571892 | *2<-URI10.v.....0.yi...y... ...k*   |
| 0BA8AF80                                  | 606EF3E4 | D9C9F1F0 | 00001C16 | 0BA50234 | 0BA88958 | 0BA88D78 | 03274F21 | 25572087 | *->3URI10....v...yi...y... ...g*    |

Figure 172. Example of Internal Trace Table Entries (Part 1 of 7)

|          |          |          |           |          |           |          |          |          |                                      |
|----------|----------|----------|-----------|----------|-----------|----------|----------|----------|--------------------------------------|
| 0BA8AFA0 | D6D7C5D5 | C5D9F140 | 0BA86728  | 00000000 | 00000000  | 00000000 | 03274F21 | 25572087 | *OPENER1 .y..... ....g*              |
| 0BA8AFC0 | 00000000 | 00000000 | 00024000  | 00024000 | 20032806  | 00024000 | 00024000 | 00023000 | *..... .....*                        |
| 0BA8AFE0 | 00002000 | 00000020 | 00000000  | 00000000 | 00000000  | 00000000 | 00000000 | 00000012 | *..... .....*                        |
| 0BA8B000 | 606EF4E4 | D9C9F1F0 | 000000D7C | 0BA5023A | 0BA88D78  | 0BA89198 | 03274F21 | 25572299 | *->4URI10...@.v...y...yjq... ....r*  |
| 0BA8B020 | F34C60E4 | D9C9F1F0 | 0BA50FB0  | 00001162 | 0BA88D78  | 0BA89198 | 03274F21 | 25572303 | *3<-URI10.v.....y...yjq... ....*     |
| 0BA8B040 | D6D7C5D5 | C5D9F240 | 0BA86728  | 00000000 | 00000000  | 00000000 | 03274F21 | 25572303 | *OPENER2 .y..... ....*               |
| 0BA8B060 | 00000000 | 00000000 | 00024000  | 00024000 | 20032806  | 00024000 | 00024000 | 00023000 | *..... .....*                        |
| 0BA8B080 | 00002000 | 00000020 | 00000000  | 00000000 | 00000000  | 00000000 | 00000000 | 00000012 | *..... .....*                        |
| 0BA8B0A0 | F24C60E4 | D9C9F1F0 | 0BA51E26  | 00000FF0 | 0BA88958  | 0BA88D78 | 03274F21 | 25572303 | *2<-URI10.v.....@.yi..y... ....*     |
| 0BA8B0C0 | 606EF3E4 | D9C9F3F0 | 000007BE  | 0BA53474 | 0BA88958  | 0BA88D78 | 03274F21 | 25572303 | *->3URI30....v...yi..y... ....*      |
| 0BA8B0E0 | F24C60E4 | D9C9F3F0 | 0BA509CE  | 00000EB2 | 0BA88958  | 0BA88D78 | 03274F21 | 25572341 | *2<-URI30.v.....yi..y... ....*       |
| 0BA8B100 | F14C60E4 | D9C9F1F0 | 0BA48DAA  | 00000942 | 0BA88010  | 0BA88958 | 03274F21 | 25572341 | *1<-URI10.u.....y...yi... ....*      |
| 0BA8B120 | 606EF2E4 | D9C9F5F0 | 00001C4C  | 0BA58C68 | 0BA88010  | 0BA88958 | 03274F21 | 25572341 | *->2URI50...<.v...y...yi... ....*    |
| 0BA8B140 | C4E2D7E2 | E3C1C3D2 | 8BA58E0C  | 0BA58C68 | 0BA88958  | 0BA88C78 | 03274F21 | 25572341 | *DSPSTACK.v...v...yi..y... ....*     |
| 0BA8B160 | C7C5E3C6 | C5C5C440 | 0BA86728  | 00000000 | 00000000  | 00000000 | 03274F21 | 25572341 | *GETFEED .y..... ....*               |
| 0BA8B180 | 00000000 | 00000280 | 00024000  | 00024000 | 20032806  | 00024000 | 00024000 | 00023000 | *..... .....*                        |
| 0BA8B1A0 | 00002000 | 80500000 | 00000000  | 0BAB059C | 00000280  | 00000100 | C7C5E340 | 40400000 | *.....&.....GET ..*                  |
| 0BA8B1C0 | F14C60E4 | D9C9F5F0 | 0BA4CA64  | 000002DE | 0BA88010  | 0BA88958 | 03274F21 | 25572341 | *1<-URI50.u.....y...yi... ....*      |
| 0BA8B1E0 | C7C5E3C6 | C5C5C440 | 0BA86728  | 00000000 | 00000000  | 00000000 | 03274F21 | 25572341 | *GETFEED .y..... ....*               |
| 0BA8B200 | 00000000 | 00000280 | 00024000  | 00024000 | 20032806  | 00024000 | 00024000 | 00023000 | *..... .....*                        |
| 0BA8B220 | 00002000 | 80500000 | 00000000  | 0BC72000 | 00000280  | 00000100 | C7C5E340 | 40400000 | *.....&.....G.....GET ..*            |
| 0BA8B240 | 606EF2C4 | C5D8F0F0 | 000042F4  | 0BA0DA68 | 0BA88010  | 0BA88958 | 03274F21 | 25572342 | *->2DEQ00...4....y...yi... ....*     |
| 0BA8B260 | F14C60C4 | C5D8F0F0 | 0BA4F10C  | 000007EE | 0BA88010  | 0BA88958 | 03274F21 | 25572343 | *1<-DEQ00.u1....y...yi... ....*      |
| 0BA8B280 | C4E2D7E4 | D9C9F0F0 | 0BA86728  | 00000000 | 00000000  | 00000000 | 03274F21 | 25572343 | *DSPURI00.y..... ....*               |
| 0BA8B2A0 | C5D5C4C9 | D9C3C1D9 | F1F201D8  | D6000600 | D7C8E8E2  | C9C3C1D3 | 40D6D7C5 | D5404040 | *ENDIRCAR12.QO...PHYSICAL OPEN *     |
| 0BA8B2C0 | 40404040 | 40404040 | 40404040  | 40404040 | 40404040  | 40404040 | 00000000 | 00000000 | *..... .....*                        |
| 0BA8B2E0 | F04C60E4 | D9C9F0F0 | 0BA85674  | 0000084E | 0BA01A30  | 0BA88010 | 03274F21 | 25572343 | *0<-URI00.y....+.y...y... ....*      |
| 0BA8B300 | C4E2D7C3 | D9E3D9F0 | 0A30DCB4  | 20212002 | 00000000  | 0BA86728 | 03274F21 | 27020292 | *DSPCRTR0.....y... ....k*            |
| 0BA8B320 | 00000000 | 00000000 | 00CE8E50  | 00000001 | 0BA86728  | 00000000 | 00000000 | 00000000 | *.....&.....y..... ....*             |
| 0BA8B340 | 00000000 | 00000000 | 00000000  | 0AEA0008 | 0A0AC8E0  | 80000000 | 00000000 | 00000000 | *..... .....H..... ....*             |
| 0BA8B360 | 606EF1E7 | C4D3F0F0 | 80009C6E  | 0BC76D90 | 0000061C8 | 0BA88010 | 03274F21 | 27042660 | *->1XDL00...>.G.../H.y... ....-*     |
| 0BA8B380 | 606EF2E4 | D9C9F0F0 | 00000240  | 0BA4AE18 | 0BA88010  | 0BA886B8 | 03274F21 | 27042696 | *->2URI00... .u...y...yf... ....o*   |
| 0BA8B3A0 | C4E2D7E4 | D9C9F0F0 | 0BA86728  | 00000000 | 00000000  | 00000000 | 03274F21 | 27042696 | *DSPURI00.y..... ....o*              |
| 0BA8B3C0 | D4D6C4C9 | D9C3C1D9 | F1F201D8  | D6000000 | 40D3D6C7  | C9C3C1D3 | 40D6D7C5 | D5404040 | *MODIRCAR12.QO... LOGICAL OPEN *     |
| 0BA8B3E0 | 40404040 | 40404040 | 40404040  | 40404040 | 40404040  | 40404040 | 00000000 | 00000000 | *..... .....*                        |
| 0BA8B400 | 606EF3D9 | E2E5F0F0 | 00001270  | 0BA80FA0 | 0BA886B8  | 0BA89000 | 03274F21 | 27042696 | *->3RSV00....y...yf.y... ....o*      |
| 0BA8B420 | F24C60D9 | E2E5F0F0 | 0BA4C088  | 0000022A | 0BA886B8  | 0BA89000 | 03274F21 | 27042698 | *2<-RSV00.u.h....yf..y... ....q*     |
| 0BA8B440 | 606EF3E4 | D9C9F3F0 | 00001340  | 0BA5347A | 0BA886B8  | 0BA89000 | 03274F21 | 27042698 | *->3URI30... .v...yf.y... ....q*     |
| 0BA8B460 | 606EF4E4 | D9C9F4E4 | 00001424  | 0BA522D8 | 0BA89000  | 0BA898A0 | 03274F21 | 27042801 | *->4URI20... .v...Q.y...yjq... ....* |
| 0BA8B480 | F34C60E4 | D9C9F2F0 | 0BA5489E  | 00000C2A | 0BA89000  | 0BA898A0 | 03274F21 | 27042801 | *3<-URI20.v.....y...yjq... ....*     |
| 0BA8B4A0 | F24C60E4 | D9C9F3F0 | 0BA4C158  | 00000F8E | 0BA886B8  | 0BA89000 | 03274F21 | 27042801 | *2<-URI30.uA....yf..y... ....*       |
| 0BA8B4C0 | 606EF3E4 | D9C9F5F0 | 00001C4C  | 0BA58C68 | 0BA886B8  | 0BA89000 | 03274F21 | 27042801 | *->3URI50...<.v...yf.y... ....*      |
| 0BA8B4E0 | C4E2D7E2 | E3C1C3D2 | 8BA58E0C  | 0BA58C68 | 0BA89000  | 0BA89320 | 03274F21 | 27042801 | *DSPSTACK.v...v...y...yl... ....*    |
| 0BA8B500 | C7C5E3C6 | C5C5C440 | 0BA86728  | 00000000 | 00000000  | 00000000 | 03274F21 | 27042801 | *GETFEED .y..... ....*               |
| 0BA8B520 | 00000000 | 00000280 | 00024000  | 00024000 | 20032806  | 00024000 | 00024000 | 00023000 | *..... .....*                        |
| 0BA8B540 | 00002000 | 80500000 | 0BA01138  | 0BAB059C | 00000280  | 00000100 | C7C5E340 | 40400000 | *.....&.....GET ..*                  |
| 0BA8B560 | F24C60E4 | D9C9F5F0 | 0BA4CA64  | 000002DE | 0BA886B8  | 0BA89000 | 03274F21 | 27042801 | *2<-URI50.u.....yf..y... ....*       |
| 0BA8B580 | C7C5E3C6 | C5C5C440 | 0BA86728  | 00000000 | 00000000  | 00000000 | 03274F21 | 27042801 | *GETFEED .y..... ....*               |
| 0BA8B5A0 | 00000000 | 00000280 | 00024000  | 00024000 | 20032806  | 00024000 | 00024000 | 00023000 | *..... .....*                        |
| 0BA8B5C0 | 00002000 | 80500000 | 0BA01138  | 0BC6E000 | 00000280  | 00000100 | C7C5E340 | 40400000 | *.....&.....F.....GET ..*            |
| 0BA8B5E0 | F14C60E4 | D9C9F0F0 | 0BC76FD0  | 0000084E | 0BA88010  | 0BA886B8 | 03274F21 | 27042802 | *1<-URI00.G?...+.y...yf... ....*     |
| 0BA8B600 | 606EF2E4 | D9C9F0F0 | 00000A84  | 0BA4AE18 | 0BA88010  | 0BA886B8 | 03274F21 | 27042802 | *->2URI00... .d.u...y...yf... ....*  |
| 0BA8B620 | C4E2D7E4 | D9C9F0F0 | 0BA86728  | 00000000 | 00000000  | 00000000 | 03274F21 | 27042802 | *DSPURI00.y..... ....*               |
| 0BA8B640 | D4D6C4C9 | D9C3C1D9 | F1F201D8  | D3002000 | C4C9D9C5  | C3E340D3 | D6C3C1E3 | C5404040 | *MODIRCAR12.QL...DIRECT LOCATE *     |
| 0BA8B660 | C8C2C8C4 | D6D1F0F1 | 00000000  | 00000000 | 18000000  | 00000000 | 00000000 | 00000000 | *HBHDOJ01..... ....*                 |
| 0BA8B680 | 606EF3E4 | D9C9F5F0 | 00001C4C  | 0BA58C68 | 0BA886B8  | 0BA89000 | 03274F21 | 27042802 | *->3URI50...<.v...yf.y... ....*      |
| 0BA8B6A0 | C4E2D7E2 | E3C1C3D2 | 8BA58E0C  | 0BA58C68 | 0BA89000  | 0BA89320 | 03274F21 | 27042802 | *DSPSTACK.v...v...y...yl... ....*    |
| 0BA8B6C0 | C7C5E3C6 | C5C5C440 | 0BA86728  | 00000000 | 00000000  | 00000000 | 03274F21 | 27042804 | *GETFEED .y..... ....*               |
| 0BA8B6E0 | 93080010 | 00000280 | 00024000  | 00024000 | 20032806  | 00024000 | 00024000 | 00023000 | *1..... .....*                       |
| 0BA8B700 | 00002000 | 80500000 | 0BA01138  | 0BAB059C | 00000280  | 00000100 | C7C5E340 | 40400000 | *.....&.....GET ..*                  |
| 0BA8B720 | F24C60E4 | D9C9F5F0 | 0BA4CA64  | 000002DE | 0BA886B8  | 0BA89000 | 03274F21 | 27042804 | *2<-URI50.u.....yf..y... ....*       |
| 0BA8B740 | C7C5E3C6 | C5C5C440 | 0BA86728  | 00000000 | 00000000  | 00000000 | 03274F21 | 27042804 | *GETFEED .y..... ....*               |
| 0BA8B760 | 93080010 | 00000280 | 00024000  | 00024000 | 20032806  | 00024000 | 00024000 | 00023000 | *1..... .....*                       |
| 0BA8B780 | 00002000 | 80500000 | 0BA01138  | 0BC6E000 | 00000280  | 00000100 | C7C5E340 | 40400000 | *.....&.....F.....GET ..*            |
| 0BA8B7A0 | C4E2D7E4 | D9C9F0F0 | 0BA86728  | 00000000 | 00000000  | 00000000 | 03274F21 | 27042804 | *DSPURI00.y..... ....*               |
| 0BA8B7C0 | C5D5C4C9 | D9C3C1D9 | F1F20004  | D3002000 | D9C5C3D6  | D9C440D5 | D6E340C6 | D6E4D5C4 | *ENDIRCAR12..L...RECORD NOT FOUND*   |
| 0BA8B7E0 | C8C2C8C4 | D6D1F0F1 | 00000000  | 00000000 | 18000000  | 00000000 | 00000000 | 00000000 | *HBHDOJ01..... ....*                 |
| 0BA8B800 | F14C60E4 | D9C9F0F0 | 0BC77814  | 0000084E | 0BA88010  | 0BA886B8 | 03274F21 | 27042804 | *1<-URI00.G....+.y...yf... ....*     |
| 0BA8B820 | 606EF2E4 | D9C9F0F0 | 00000A84  | 0BA4AE18 | 0BA88010  | 0BA886B8 | 03274F21 | 27042804 | *->2URI00... .d.u...y...yf... ....*  |

Figure 172. Example of Internal Trace Table Entries (Part 2 of 7)

|          |          |          |           |           |           |          |          |          |                                    |
|----------|----------|----------|-----------|-----------|-----------|----------|----------|----------|------------------------------------|
| 0BA8B840 | C4E2D7E4 | D9C9F0F0 | 0BA86728  | 00000000  | 00000000  | 00000000 | 03274F21 | 27042804 | *DSPURI00.y..... .....*            |
| 0BA8B860 | D4D6C4C9 | D9C3C1D9 | F1F201D8  | D3002000  | C4C9D9C5  | C3E340D3 | D6C3C1E3 | C5404040 | *MODIRCAR12.QL...DIRECT LOCATE *   |
| 0BA8B880 | C8C2C8C4 | D6D2F0F1 | 00000000  | 00000000  | 18000000  | 00000000 | 00000000 | 00000000 | *HBHDOK01..... .....*              |
| 0BA8B8A0 | 606EF3E4 | D9C9F5F0 | 00001C4C  | 0BA58C68  | 0BA886B8  | 0BA89000 | 03274F21 | 27042804 | *->3URI50.<.v...yf..y.... .....*   |
| 0BA8B8C0 | C4E2D7E2 | E3C1C3D2 | 8BA58E0C  | 0BA58C68  | 0BA89000  | 0BA89320 | 03274F21 | 27042804 | *DSPSTACK.v...v...y1... .....*     |
| 0BA8B8E0 | C7C5E3C6 | C5C5C440 | 0BA86728  | 00000000  | 00000000  | 00000000 | 03274F21 | 27042804 | *GETFEED .y..... .....*            |
| 0BA8B900 | 93080010 | 00000280 | 00024000  | 00024000  | 20032806  | 00024000 | 00024000 | 00023000 | *1..... .....*                     |
| 0BA8B920 | 00002000 | 80500000 | 0BA01138  | 0BAB059C  | 00000280  | 00000100 | C7C5E340 | 40400000 | *.....&.....GET ..*                |
| 0BA8B940 | F24C60E4 | D9C9F5F0 | 0BA4CA64  | 000002DE  | 0BA886B8  | 0BA89000 | 03274F21 | 27042804 | *2<-URI50.u.....yf..y.... .....*   |
| 0BA8B960 | C7C5E3C6 | C5C5C440 | 0BA86728  | 00000000  | 00000000  | 00000000 | 03274F21 | 27042804 | *GETFEED .y..... .....*            |
| 0BA8B980 | 93080010 | 00000280 | 00024000  | 00024000  | 20032806  | 00024000 | 00024000 | 00023000 | *1..... .....*                     |
| 0BA8B9A0 | 00002000 | 80500000 | 0BA01138  | 00000000  | 00000280  | 00000100 | C7C5E340 | 40400000 | *.....&.....GET ..*                |
| 0BA8B9C0 | C4E2D7E4 | D9C9F0F0 | 0BA86728  | 00000000  | 00000000  | 00000000 | 03274F21 | 27042804 | *DSPURI00.y..... .....*            |
| 0BA8B9E0 | C5D5C4C9 | D9C3C1D9 | F1F20004  | D3002000  | D9C5C3D6  | D9C440D5 | D6E340C6 | D6E4D5C4 | *ENDIRCAR12.L...RECORD NOT FOUND*  |
| 0BA8BA00 | C8C2C8C4 | D6D2F0F1 | 00000000  | 00000000  | 18000000  | 00000000 | 00000000 | 00000000 | *HBHDOK01..... .....*              |
| 0BA8BA20 | F14C60E4 | D9C9F0F0 | 0BC77814  | 00000084E | 0BA88010  | 0BA886B8 | 03274F21 | 27042804 | *1<-URI00.G.....+y...yf... .....*  |
| 0BA8BA40 | 606EF2E4 | D9C9F0F0 | 000007C8  | 0BA4AE18  | 0BA88010  | 0BA886B8 | 03274F21 | 27042804 | *->2URI00...H.u...y...yf... .....* |
| 0BA8BA60 | C4E2D7E4 | D9C9F0F0 | 0BA86728  | 00000000  | 00000000  | 00000000 | 03274F21 | 27042804 | *DSPURI00.y..... .....*            |
| 0BA8BA80 | D4D6C4C9 | D9C3C1D9 | F1F201D8  | C3002000  | 40D3D6C7  | C9C3C1D3 | 40C3D3D6 | E2C54040 | *MODIRCAR12.QC... LOGICAL CLOSE *  |
| 0BA8BAA0 | 40404040 | 40404040 | 40404040  | 40404040  | 40404040  | 40404040 | 00000000 | 00000000 | *..... .....*                      |
| 0BA8BAC0 | 606EF3C4 | C5D8F0F0 | 000042F4  | 0BA0DA68  | 0BA886B8  | 0BA89000 | 03274F21 | 27042805 | *->3DEQ00...4....yf..y.... .....*  |
| 0BA8BAE0 | F24C60C4 | C5D8F0F0 | 0BA4F10C  | 000007EE  | 0BA886B8  | 0BA89000 | 03274F21 | 27042805 | *2<-DEQ00.u1....yf..y.... .....*   |
| 0BA8BB00 | F14C60E4 | D9C9F0F0 | 0BC77558  | 00000084E | 0BA88010  | 0BA886B8 | 03274F21 | 27042805 | *1<-URI00.G.....+y...yf... .....*  |
| 0BA8BB20 | C4E2D7E2 | C4C3F0F0 | 000009C6E | 0000008B6 | 0000061C8 | 0BA88010 | 03274F21 | 27042805 | *0<-XDL00.>.../H.y.... .....*      |
| 0BA8BB40 | C3D9E3D9 | F0E7C9E3 | 0A30DCB4  | 20212002  | 00000000  | 0BA86728 | 03274F21 | 27042805 | *CRTR0XIT.....y.... .....*         |
| 0BA8BB60 | 00000000 | 0A313820 | 0A30DCB4  | 0A649490  | 00000008  | C4C6E2C2 | D9D3E2C2 | 00000100 | *.....m....DFSBRLSB...*            |
| 0BA8BB80 | 20212002 | 00000000 | 00000000  | 00000000  | 00000000  | 00000000 | 00000000 | 00000000 | *..... .....*                      |
| 0BA8BBA0 | C4E2D7C3 | D9E3D9F0 | 0A239B4E  | 44722002  | 00000000  | 0BA86728 | 03274F21 | 27050704 | *DSPCRT00...+.....y.... .....*     |
| 0BA8BBC0 | 00000000 | 00000000 | 00CE8E50  | 00000001  | 0BA86728  | 00000000 | 00000000 | 00000000 | *.....&.....y..... .....*          |
| 0BA8BBE0 | 00000000 | 00000000 | 00000000  | C9D4E2C7  | E2C7F140  | 00D1C024 | 00000000 | 00000000 | *.....IMSGSG1 .J..... .....*       |
| 0BA8BC00 | 606EF16F | 6F6F6F6F | 800098AC  | 0BA22010  | 0000061C8 | 0BA88010 | 03274F21 | 27050704 | *->1?????.q...s.../H.y.... .....*  |
| 0BA8BC20 | 606EF2E4 | D9C9F0F0 | 0000015A  | 0BA4AE18  | 0BA88010  | 0BA88068 | 03274F21 | 27050704 | *->2URI00...!u...y...y.... .....*  |
| 0BA8BC40 | C4E2D7E4 | D9C9F0F0 | 0BA86728  | 00000000  | 00000000  | 00000000 | 03274F21 | 27050704 | *DSPURI00.y..... .....*            |
| 0BA8BC60 | D4D6C4C9 | D9C3C1D9 | F1F201D8  | D6002000  | 40D3D6C7  | C9C3C1D3 | 40D6D7C5 | D5404040 | *MODIRCAR12.QO... LOGICAL OPEN *   |
| 0BA8BC80 | 40404040 | 40404040 | 40404040  | 40404040  | 40404040  | 40404040 | 00000000 | 00000000 | *..... .....*                      |
| 0BA8BCA0 | 606EF3D9 | E2E5F0F0 | 00001270  | 0BA80FA0  | 0BA88068  | 0BA889B0 | 03274F21 | 27050704 | *->3RSV00....y...y...yi... .....*  |
| 0BA8BCC0 | F24C60D9 | E2E5F0F0 | 0BA4C088  | 0000022A  | 0BA88068  | 0BA889B0 | 03274F21 | 27050705 | *2<-RSV00.u.h....y...yi... .....*  |
| 0BA8BCE0 | 606EF3E4 | D9C9F3F0 | 00001340  | 0BA5347A  | 0BA88068  | 0BA889B0 | 03274F21 | 27050705 | *->3URI30...v...y...yi... .....*   |
| 0BA8BD00 | F24C60E4 | D9C9F3F0 | 0BA4C158  | 000000F8E | 0BA88068  | 0BA889B0 | 03274F21 | 27050710 | *2<-URI30.uA.....y...yi... .....*  |
| 0BA8BD20 | 606EF3E4 | D9C9F5F0 | 00001C4C  | 0BA58C68  | 0BA88068  | 0BA889B0 | 03274F21 | 27050710 | *->3URI50.<.v...y...yi... .....*   |
| 0BA8BD40 | C4E2D7E2 | E3C1C3D2 | 8BA58E0C  | 0BA58C68  | 0BA889B0  | 0BA88CD0 | 03274F21 | 27050710 | *DSPSTACK.v...v...yi..y.... .....* |
| 0BA8BD60 | C7C5E3C6 | C5C5C440 | 0BA86728  | 00000000  | 00000000  | 00000000 | 03274F21 | 27050710 | *GETFEED .y..... .....*            |
| 0BA8BD80 | 00000000 | 00000280 | 00024000  | 00024000  | 20032806  | 00024000 | 00024000 | 00023000 | *..... .....*                      |
| 0BA8BDA0 | 00002000 | 80500000 | 0BA01138  | 0BAB059C  | 00000280  | 00000100 | C7C5E340 | 40400000 | *.....&.....GET ..*                |
| 0BA8BDC0 | F24C60E4 | D9C9F5F0 | 0BA4CA64  | 000002DE  | 0BA88068  | 0BA889B0 | 03274F21 | 27050710 | *2<-URI50.u.....y...yi... .....*   |
| 0BA8BDE0 | C7C5E3C6 | C5C5C440 | 0BA86728  | 00000000  | 00000000  | 00000000 | 03274F21 | 27050710 | *GETFEED .y..... .....*            |
| 0BA8BE00 | 00000000 | 00000280 | 00024000  | 00024000  | 20032806  | 00024000 | 00024000 | 00023000 | *..... .....*                      |
| 0BA8BE20 | 00002000 | 80500000 | 0BA01138  | 0BC72000  | 00000280  | 00000100 | C7C5E340 | 40400000 | *.....&.....G.....GET ..*          |
| 0BA8BE40 | F14C60E4 | D9C9F0F0 | 0BA2216A  | 00000084E | 0BA88010  | 0BA88068 | 03274F21 | 27050711 | *1<-URI00.s.....+y...y.... .....*  |
| 0BA8BE60 | 606EF2E4 | D9C9F0F0 | 00000196  | 0BA4AE18  | 0BA88010  | 0BA88068 | 03274F21 | 27050711 | *->2URI00...o.u...y...y.... .....* |
| 0BA8BE80 | C4E2D7E4 | D9C9F0F0 | 0BA86728  | 00000000  | 00000000  | 00000000 | 03274F21 | 27050711 | *DSPURI00.y..... .....*            |
| 0BA8BEA0 | D4D6C4C9 | D9C3C1D9 | F1F201D8  | D3002000  | C4C9D9C5  | C3E340D3 | D6C3C1E3 | C5404040 | *MODIRCAR12.QL...DIRECT LOCATE *   |
| 0BA8BEC0 | FFFFFFFF | FFFF0000 | C9D4E2C7  | E2C7F140  | 3A000000  | 00000000 | 00000000 | 00000000 | *.....IMSGSG1..... .....*          |
| 0BA8BEE0 | 606EF3E4 | D9C9F5F0 | 00001C4C  | 0BA58C68  | 0BA88068  | 0BA889B0 | 03274F21 | 27050711 | *->3URI50.<.v...y...yi... .....*   |
| 0BA8BF00 | C4E2D7E2 | E3C1C3D2 | 8BA58E0C  | 0BA58C68  | 0BA889B0  | 0BA88CD0 | 03274F21 | 27050711 | *DSPSTACK.v...v...yi..y.... .....* |
| 0BA8BF20 | C7C5E3C6 | C5C5C440 | 0BA86728  | 00000000  | 00000000  | 00000000 | 03274F21 | 27050712 | *GETFEED .y..... .....*            |
| 0BA8BF40 | 00000000 | 000000E0 | 00024000  | 00024000  | 20032806  | 00024000 | 00024000 | 00023000 | *..... .....*                      |
| 0BA8BF60 | 00002000 | 80500000 | 0BA01138  | 0BAB059C  | 000000E0  | 00000100 | C7C5E340 | 40400000 | *.....&.....GET ..*                |
| 0BA8BF80 | F24C60E4 | D9C9F5F0 | 0BA4CA64  | 000002DE  | 0BA88068  | 0BA889B0 | 03274F21 | 27050712 | *2<-URI50.u.....y...yi... .....*   |
| 0BA8BFA0 | C7C5E3C6 | C5C5C440 | 0BA86728  | 00000000  | 00000000  | 00000000 | 03274F21 | 27050712 | *GETFEED .y..... .....*            |
| 0BA8BFC0 | 00000000 | 000000E0 | 00024000  | 00024000  | 20032806  | 00024000 | 00024000 | 00023000 | *..... .....*                      |
| 0BA8BFE0 | 00002000 | 80500000 | 0BA01138  | 0BC6DC0F0 | 000000E0  | 00000100 | C7C5E340 | 40400000 | *.....&.....F.0.....GET ..*        |
| 0BA8C000 | C4E2D7E4 | D9C9F0F0 | 0BA86728  | 00000000  | 00000000  | 00000000 | 03274F21 | 27050712 | *DSPURI00.y..... .....*            |
| 0BA8C020 | C5D5C4C9 | D9C3C1D9 | F1F200E0  | D3002000  | D9C5C3D6  | D9C440E6 | C1E240C6 | D6E4D5C4 | *ENDIRCAR12.L...RECORD WAS FOUND*  |
| 0BA8C040 | FFFFFFFF | FFFF0000 | C9D4E2C7  | E2C7F140  | 3A000000  | 00000000 | 00000000 | 00000000 | *.....IMSGSG1..... .....*          |
| 0BA8C060 | F14C60E4 | D9C9F0F0 | 0BA221A6  | 00000084E | 0BA88010  | 0BA88068 | 03274F21 | 27050712 | *1<-URI00.s.w...+y...y.... .....*  |
| 0BA8C080 | 606EF2E4 | D9C9F0F0 | 0000024E  | 0BA4AE18  | 0BA88010  | 0BA88068 | 03274F21 | 27050712 | *->2URI00...+u...y...y.... .....*  |
| 0BA8C0A0 | C4E2D7E4 | D9C9F0F0 | 0BA86728  | 00000000  | 00000000  | 00000000 | 03274F21 | 27050712 | *DSPURI00.y..... .....*            |
| 0BA8C0C0 | D4D6C4C9 | D9C3C1D9 | F1F201D8  | C3002000  | 40D3D6C7  | C9C3C1D3 | 40C3D3D6 | E2C54040 | *MODIRCAR12.QC... LOGICAL CLOSE *  |

Figure 172. Example of Internal Trace Table Entries (Part 3 of 7)

|          |            |          |          |          |          |          |          |          |                                     |        |
|----------|------------|----------|----------|----------|----------|----------|----------|----------|-------------------------------------|--------|
| 0BA8C0E0 | 40404040   | 40404040 | 40404040 | 40404040 | 40404040 | 40404040 | 00000000 | 00000000 | *                                   | .....* |
| 0BA8C100 | 606EF3C4   | C5D8F0F0 | 000042F4 | 0BA0DA68 | 0BA88068 | 0BA889B0 | 03274F21 | 27050712 | *->3DEQ00...4.....y...yi... .....*  | .....* |
| 0BA8C120 | F24C60C4   | C5D8F0F0 | 0BA4F10C | 000007EE | 0BA88068 | 0BA889B0 | 03274F21 | 27050713 | *2<-DEQ00.u1.....y...yi... .....*   | .....* |
| 0BA8C140 | F14C60E4   | D9C9F0F0 | 0BA2225E | 0000084E | 0BA88010 | 0BA88068 | 03274F21 | 27050713 | *1<-URI00.s.;...+y...y... .....*    | .....* |
| 0BA8C160 | F04C606F   | 6F6F6F6F | 000098AC | 00000278 | 000061C8 | 0BA88010 | 03274F21 | 27050713 | *0<-?????...q...../H.y... .....*    | .....* |
| 0BA8C180 | C3D9E3D9   | F0E7C9E3 | 0A239B4E | 44722002 | 00000000 | 0BA86728 | 03274F21 | 27050713 | *CRTR0XIT...+.....y... .....*       | .....* |
| 0BA8C1A0 | 00000000   | 0A115E40 | 0A239B4E | 0A649490 | 00000000 | C4C6E2C2 | D9D3E2C2 | 00000100 | *.....;...+...m....DFSBRLSB...*     | .....* |
| 0BA8C1C0 | 44722002   | 00000000 | 00000000 | 00000000 | 00000000 | 00000000 | 00000000 | 00000000 | *..... .....*                       | .....* |
| 0BA8C1E0 | C4E2D7C3   | D9E3D9F0 | 0A23E56C | 17172002 | 00000000 | 0BA86728 | 03274F21 | 29025787 | *DSPCRTR0.V%.....y... ....g*        | .....* |
| 0BA8C200 | 00000000   | 00000000 | 00CE8E50 | 00000001 | 0BA86728 | 00000000 | 00000000 | 00000000 | *.....&.....y.....*                 | .....* |
| 0BA8C220 | 00000000   | 00000000 | 00000000 | 00000000 | 00000000 | 00000000 | 00000000 | 00000000 | *..... .....*                       | .....* |
| 0BA8C240 | 606EF1E2   | E2C9C7D5 | 8000972C | 0BA327B8 | 000061C8 | 0BA88010 | 03274F21 | 29025787 | *->1SSIGN...p..t.../H.y... ....g*   | .....* |
| 0BA8C260 | 606EF2E4   | D9C9F0F0 | 0000020C | 0BA4AE18 | 0BA88010 | 0BA88220 | 03274F21 | 29025787 | *->2URI00.....u...y...yb... ....g*  | .....* |
| 0BA8C280 | C4E2D7E4   | D9C9F0F0 | 0BA86728 | 00000000 | 00000000 | 00000000 | 03274F21 | 29025787 | *DSPURI00.y..... ....g*             | .....* |
| 0BA8C2A0 | D4D6C4C9   | D9C3C1D9 | F1F201D8 | D6002000 | 40D3D6C7 | C9C3C1D3 | 40D6D7C5 | D5404040 | *MODIRCAR12.QO... LOGICAL OPEN *    | .....* |
| 0BA8C2C0 | 40404040   | 40404040 | 40404040 | 40404040 | 40404040 | 40404040 | 00000000 | 00000000 | *..... .....*                       | .....* |
| 0BA8C2E0 | 606EF3D9   | E2E5F0F0 | 00001270 | 0BA80FA0 | 0BA88220 | 0BA88B68 | 03274F21 | 29025788 | *->3RSV00.....y...yb.y... ....h*    | .....* |
| 0BA8C300 | F24C60D9   | E2E5F0F0 | 0BA4C088 | 0000022A | 0BA88220 | 0BA88B68 | 03274F21 | 29025792 | *2<-RSV00.u.h.....yb.y... ....k*    | .....* |
| 0BA8C320 | 606EF3E4   | D9C9F3F0 | 00001340 | 0BA5347A | 0BA88220 | 0BA88B68 | 03274F21 | 29025792 | *->3URI30... .v.:.yb.y... ....k*    | .....* |
| 0BA8C340 | F24C60E4   | D9C9F3F0 | 0BA4C158 | 00000F8E | 0BA88220 | 0BA88B68 | 03274F21 | 29025797 | *2<-URI30.uA.....yb.y... ....p*     | .....* |
| 0BA8C360 | 606EF3E4   | D9C9F5F0 | 00001C4C | 0BA58C68 | 0BA88220 | 0BA88B68 | 03274F21 | 29025797 | *->3URI50...<.v...yb.y... ....p*    | .....* |
| 0BA8C380 | C4E2D7E2   | E3C1C3D2 | 8BA58E0C | 0BA58C68 | 0BA88B68 | 0BA88E88 | 03274F21 | 29025797 | *DSPSTACK.v...v...y...y.h... ....p* | .....* |
| 0BA8C3A0 | C7C5E3C6   | C5C5C440 | 0BA86728 | 00000000 | 00000000 | 00000000 | 03274F21 | 29025797 | *GETFEED .y..... ....p*             | .....* |
| 0BA8C3C0 | 00000000   | 00000000 | 00024000 | 00024000 | 20032806 | 00024000 | 00024000 | 00023000 | *..... .....*                       | .....* |
| 0BA8C3E0 | 00002000   | 80500000 | 0BA01138 | 0BAB059C | 00000280 | 00000100 | C7C5E340 | 40400000 | *.....&.....GET ..*                 | .....* |
| 0BA8C400 | F24C60E4   | D9C9F5F0 | 0BA4CA64 | 000002DE | 0BA88220 | 0BA88B68 | 03274F21 | 29025797 | *2<-URI50.u.....yb.y... ....p*      | .....* |
| 0BA8C420 | C7C5E3C6   | C5C5C440 | 0BA86728 | 00000000 | 00000000 | 00000000 | 03274F21 | 29025797 | *GETFEED .y..... ....p*             | .....* |
| 0BA8C440 | 00000000   | 00000280 | 00024000 | 00024000 | 20032806 | 00024000 | 00024000 | 00023000 | *..... .....*                       | .....* |
| 0BA8C460 | 00002000   | 80500000 | 0BA01138 | 0BC6E000 | 00000280 | 00000100 | C7C5E340 | 40400000 | *.....&.....F.....GET ..*           | .....* |
| 0BA8C480 | F14C60E4   | D9C9F0F0 | 0BA329C4 | 0000084E | 0BA88010 | 0BA88220 | 03274F21 | 29025797 | *1<-URI00.t.D...+y...yb... ....p*   | .....* |
| 0BA8C4A0 | 606EF2E4   | D9C9F0F0 | 00000CA0 | 0BA4AE18 | 0BA88010 | 0BA88220 | 03274F21 | 29025797 | *->2URI00...u...y...yb... ....p*    | .....* |
| 0BA8C4C0 | C4E2D7E4   | D9C9F0F0 | 0BA86728 | 00000000 | 00000000 | 00000000 | 03274F21 | 29025797 | *DSPURI00.y..... ....p*             | .....* |
| 0BA8C4E0 | D4D6C4C9   | D9C3C1D9 | F1F201D8 | D3002000 | C4C9D9C5 | C3E340D3 | D6C3C1E3 | C5404040 | *MODIRCAR12.QL...DIRECT LOCATE *    | .....* |
| 0BA8C500 | FFFFFFFFFF | FFFF0000 | C9D4E2C7 | E2C7F140 | 3A000000 | 00000000 | 00000000 | 00000000 | *.....IMSGSG1 ..*                   | .....* |
| 0BA8C520 | 606EF3E4   | D9C9F5F0 | 00001C4C | 0BA58C68 | 0BA88220 | 0BA88B68 | 03274F21 | 29025798 | *->3URI50...<.v...yb.y... ....q*    | .....* |
| 0BA8C540 | C4E2D7E2   | E3C1C3D2 | 8BA58E0C | 0BA58C68 | 0BA88B68 | 0BA88E88 | 03274F21 | 29025798 | *DSPSTACK.v...v...y...y.h... ....q* | .....* |
| 0BA8C560 | C7C5E3C6   | C5C5C440 | 0BA86728 | 00000000 | 00000000 | 00000000 | 03274F21 | 29025798 | *GETFEED .y..... ....q*             | .....* |
| 0BA8C580 | 00000000   | 00000000 | 00024000 | 00024000 | 20032806 | 00024000 | 00024000 | 00023000 | *..... .....*                       | .....* |
| 0BA8C5A0 | 00002000   | 80500000 | 0BA01138 | 0BAB059C | 000000E0 | 00000100 | C7C5E340 | 40400000 | *.....&.....GET ..*                 | .....* |
| 0BA8C5C0 | F24C60E4   | D9C9F5F0 | 0BA4CA64 | 000002DE | 0BA88220 | 0BA88B68 | 03274F21 | 29025798 | *2<-URI50.u.....yb.y... ....q*      | .....* |
| 0BA8C5E0 | C7C5E3C6   | C5C5C440 | 0BA86728 | 00000000 | 00000000 | 00000000 | 03274F21 | 29025799 | *GETFEED .y..... ....r*             | .....* |
| 0BA8C600 | 00000000   | 00000000 | 00024000 | 00024000 | 20032806 | 00024000 | 00024000 | 00023000 | *..... .....*                       | .....* |
| 0BA8C620 | 00002000   | 80500000 | 0BA01138 | 0BC6DCF0 | 000000E0 | 00000100 | C7C5E340 | 40400000 | *.....&.....F.0.....GET ..*         | .....* |
| 0BA8C640 | C4E2D7E4   | D9C9F0F0 | 0BA86728 | 00000000 | 00000000 | 00000000 | 03274F21 | 29025799 | *DSPURI00.y..... ....r*             | .....* |
| 0BA8C660 | C5D5C4C9   | D9C3C1D9 | F1F200E0 | D3002000 | D9C5C3D6 | D9C440E6 | C1E240C6 | D6E4D5C4 | *ENDIRCAR12.L...RECORD WAS FOUND*   | .....* |
| 0BA8C680 | FFFFFFFF   | FFFF0000 | C9D4E2C7 | E2C7F140 | 3A000000 | 00000000 | 00000000 | 00000000 | *.....IMSGSG1 ..*                   | .....* |
| 0BA8C6A0 | F14C60E4   | D9C9F0F0 | 0BA33458 | 0000084E | 0BA88010 | 0BA88220 | 03274F21 | 29025799 | *1<-URI00.t.....+y...yb... ....r*   | .....* |
| 0BA8C6C0 | 606EF2E4   | D9C9F0F0 | 00000E18 | 0BA4AE18 | 0BA88010 | 0BA88220 | 03274F21 | 29025799 | *->2URI00...u...y...yb... ....r*    | .....* |
| 0BAAA240 | C4E2D7E4   | D9C9F0F0 | 0BA86728 | 00000000 | 00000000 | 00000000 | 03274F21 | 34301939 | *DSPURI00.y..... ....*              | .....* |
| 0BAAA260 | D4D6C4C9   | D9C3C1D9 | F1F200A0 | D4002000 | C3C8C1D5 | C7C540D6 | D3C440D9 | C5C3D9C4 | *MODIRCAR12.M...CHANGE OLD RECRD*   | .....* |
| 0BAAA280 | 00000000   | 00000000 | 00000000 | 00000000 | 07200327 | 4F212904 | 50000002 | 8D000000 | *..... ....&.....*                  | .....* |
| 0BAAA2A0 | 606EF5E4   | D9C9F5F0 | 00001C4C | 0BA58C68 | 0BAD3010 | 0BAD3958 | 03274F21 | 34301939 | *->5URI50...<.v..... ....*          | .....* |
| 0BAAA2C0 | C4E2D7E2   | E3C1C3D2 | 8BA58E0C | 0BA58C68 | 0BAD3958 | 0BAD3C78 | 03274F21 | 34301939 | *DSPSTACK.v...v..... ....*          | .....* |
| 0BAAA2E0 | C7C5E3C6   | C5C5C440 | 0BA86728 | 00000000 | 00000000 | 00000000 | 03274F21 | 34301939 | *GETFEED .y..... ....*              | .....* |
| 0BAAA300 | 00000000   | 00000080 | 00024000 | 00024000 | 20032816 | 00024000 | 00024000 | 00023000 | *..... .....*                       | .....* |
| 0BAAA320 | 00002000   | 80500000 | 0BA01138 | 0BAB059C | 00000080 | 00000100 | C7C5E340 | 40400000 | *.....&.....GET ..*                 | .....* |
| 0BAAA340 | F44C60E4   | D9C9F5F0 | 0BA4CA64 | 000002DE | 0BAD3010 | 0BAD3958 | 03274F21 | 34301939 | *4<-URI50.u..... ....*              | .....* |
| 0BAAA360 | C7C5E3C6   | C5C5C440 | 0BA86728 | 00000000 | 00000000 | 00000000 | 03274F21 | 34301939 | *GETFEED .y..... ....*              | .....* |
| 0BAAA380 | 00000000   | 00000080 | 00024000 | 00024000 | 20032816 | 00024000 | 00024000 | 00023000 | *..... .....*                       | .....* |
| 0BAAA3A0 | 00002000   | 80500000 | 0BA01138 | 0BC732A0 | 00000080 | 00000100 | C7C5E340 | 40400000 | *.....&.....G.....GET ..*           | .....* |
| 0BAAA3C0 | 606EF5E4   | D9C9F5F0 | 00003F98 | 0BA58C68 | 0BAD3010 | 0BAD3958 | 03274F21 | 34301939 | *->5URI50...q.v..... ....*          | .....* |
| 0BAAA3E0 | C4E2D7E2   | E3C1C3D2 | 8BA58E0C | 0BA58C68 | 0BAD3958 | 0BAD3C78 | 03274F21 | 34301939 | *DSPSTACK.v...v..... ....*          | .....* |
| 0BAAA400 | 606EF6E4   | D9C9F5F0 | 00001B66 | 0BA58C68 | 0BAD3958 | 0BAD3D20 | 03274F21 | 34301939 | *->6URI50...v..... ....*            | .....* |
| 0BAAA420 | C4E2D7E2   | E3C1C3D2 | 8BA58E0C | 0BA58C68 | 0BAD3D20 | 0BAD4040 | 03274F21 | 34301940 | *DSPSTACK.v...v..... ....*          | .....* |
| 0BAAA440 | C7C5E3C6   | C5C5C440 | 0BA86728 | 00000000 | 00000000 | 00000000 | 03274F21 | 34301940 | *GETFEED .y..... ....*              | .....* |
| 0BAAA460 | 00000000   | 00000080 | 00024000 | 00024000 | 20032816 | 00024000 | 00024000 | 00023000 | *..... .....*                       | .....* |
| 0BAAA480 | 00002000   | 80500000 | 0BA01138 | 0BAD3ADC | 00000080 | 00000100 | C7C5E340 | 40400000 | *.....&.....GET ..*                 | .....* |
| 0BAAA4A0 | F54C60E4   | D9C9F5F0 | 0BA5A7CE | 000002DE | 0BAD3958 | 0BAD3D20 | 03274F21 | 34301940 | *5<-URI50.vx..... ....*             | .....* |
| 0BAAA4C0 | C7C5E3C6   | C5C5C440 | 0BA86728 | 00000000 | 00000000 | 00000000 | 03274F21 | 34301972 | *GETFEED .y..... ....*              | .....* |

Figure 172. Example of Internal Trace Table Entries (Part 4 of 7)

|           |          |          |          |          |          |          |          |          |                                    |
|-----------|----------|----------|----------|----------|----------|----------|----------|----------|------------------------------------|
| 0BAAA4E0  | 00000000 | 000000A0 | 00024000 | 00024000 | 20032816 | 00024000 | 00024000 | 00023000 | *.....                             |
| 0BAAA500  | 00002000 | 80500000 | 0BA01138 | 0BACFFA0 | 000000A0 | 00000101 | D7E4E340 | 40400000 | *....&.....PUT ..*                 |
| 0BAAA520  | 606EF6E4 | D9C9F5F0 | 00001B66 | 0BA58C68 | 0BAD3958 | 0BAD3D20 | 03274F21 | 34301972 | *->6URI50....v.....                |
| 0BAAA540  | C4E2D7E2 | E3C1C3D2 | 8BA58E0C | 0BA58C68 | 0BAD3D20 | 0BAD4040 | 03274F21 | 34301972 | *DSPSTACK.v...v.....               |
| 0BAAA560  | C7C5E3C6 | C5C5C440 | 0BA86728 | 00000000 | 00000000 | 00000000 | 03274F21 | 34301973 | *GETFEED .y.....                   |
| 0BAAA580  | 00000000 | 00000080 | 00024000 | 00024000 | 20032816 | 00024000 | 00024000 | 00023000 | *.....                             |
| 0BAAA5A0  | 00002000 | 80500000 | 0BA013B8 | 0BAD3ADC | 00000080 | 00000200 | C7C5E340 | 40400000 | *....&.....GET ..*                 |
| 0BAAA5C0  | F54C60E4 | D9C9F5F0 | 0BA5A7CE | 000002DE | 0BAD3958 | 0BAD3D20 | 03274F21 | 34301973 | *5<-URI50.vx.....                  |
| 0BAAA5E0  | C7C5E3C6 | C5C5C440 | 0BA86728 | 00000000 | 00000000 | 00000000 | 03274F21 | 34302023 | *GETFEED .y.....                   |
| 0BAAA600  | 00000000 | 000000A0 | 00024000 | 00024000 | 20032816 | 00024000 | 00024000 | 00023000 | *.....                             |
| 0BAAA620  | 00002000 | 80500000 | 0BA013B8 | 0BAEDFA0 | 000000A0 | 00000201 | D7E4E340 | 40400000 | *....&.....PUT ..*                 |
| 0BAAA640  | F14C60E4 | D9C9F5F0 | 0BA4EDB0 | 000002DE | 0BAD3010 | 0BAD3958 | 03274F21 | 34302023 | *4<-URI50.u.....                   |
| 0BAAA660  | F34C60E4 | D9C9F0F0 | 0BA60F70 | 0000084E | 0BA89560 | 0BAD3010 | 03274F21 | 34302023 | *3<-URI00.w.....+yn-...            |
| 0BAAA680  | F24C60E4 | D9E3F1F0 | 0BA5F5E4 | 000012AA | 0BA88C48 | 0BA89560 | 03274F21 | 34302023 | *2<-URT10.v5U....y...yn-...        |
| 0BAAA6A0  | 606EF3E4 | D9C9F0F0 | 000003D2 | 0BA4AE18 | 0BA88C48 | 0BA89560 | 03274F21 | 34302023 | *->3URI00....K.u...y...yn-...      |
| 0BAAA6C0  | C4E2D7E4 | D9C9F0F0 | 0BA86728 | 00000000 | 00000000 | 00000000 | 03274F21 | 34302023 | *DSPUR100.y.....                   |
| 0BAAA6E0  | D4D6C4C9 | D9C3C1D9 | F1F201D8 | C3002000 | 40D3D6C7 | C9C3C1D3 | 40C3D3D6 | E2C54040 | *MODIRCAR12.QC... LOGICAL CLOSE *  |
| 0BAAA700  | 40404040 | 40404040 | 40404040 | 40404040 | 40404040 | 40404040 | 00000000 | 00000000 | *.....                             |
| 0BAAA720  | F24C60E4 | D9C9F0F0 | 0BA5CECA | 0000084E | 0BA88C48 | 0BA89560 | 03274F21 | 34302023 | *2<-URI00.v.....+y...yn-...        |
| 0BAAA740  | F4C60E4  | D9C9F0F0 | 0BA0657E | 000003FC | 0BAD8010 | 0BA88C48 | 03274F21 | 34302024 | *1<-URT00...=.y...y.....           |
| 0BAAA760  | 606EF2E4 | D9E4D7C4 | 00000EF0 | 0BA6F880 | 0BA88010 | 0BA88C48 | 03274F21 | 34302024 | *->2URUPD...0.w8.y...y.....        |
| 0BAAA780  | 606EF3E4 | D9C9F0F0 | 000006CA | 0BA4AE18 | 0BA88C48 | 0BA892F0 | 03274F21 | 34302024 | *->3URI00....u...y...yk0...        |
| 0BAAA7A0  | C4E2D7E4 | D9C9F0F0 | 0BA86728 | 00000000 | 00000000 | 00000000 | 03274F21 | 34302024 | *DSPUR100.y.....                   |
| 0BAAA7C0  | D4D6C4C9 | D9C3C1D9 | F1F201D8 | D3002000 | C4C9D9C5 | C3E340D3 | D6C3C1E3 | C5404040 | *MODIRCAR12.QL...DIRECT LOCATE *   |
| 0BAAA7E0  | 00000000 | 00000000 | 00000000 | 00000000 | 01000000 | 00000000 | 00000000 | 00000000 | *.....                             |
| 0BAAA800  | 606EF4E4 | D9C9F5F0 | 00001C4C | 0BA58C68 | 0BA892F0 | 0BA89C38 | 03274F21 | 34302024 | *->4URI50...<.v...yk0.y.....       |
| 0BAAA820  | C4E2D7E2 | E3C1C3D2 | 8BA58E0C | 0BA58C68 | 0BA89C38 | 0BA89F58 | 03274F21 | 34302024 | *DSPSTACK.v...v...y...y.....       |
| 0BAAA840  | C7C5E3C6 | C5C5C440 | 0BA86728 | 00000000 | 00000000 | 00000000 | 03274F21 | 34302024 | *GETFEED .y.....                   |
| 0BAAA860  | 00000000 | 00000280 | 00024000 | 00024000 | 20032816 | 00024000 | 00024000 | 00023000 | *.....                             |
| 0BAAA880  | 00002000 | 80500000 | 0BA01138 | 0BAB059C | 00000280 | 00000100 | C7C5E340 | 40400000 | *....&.....GET ..*                 |
| 0BAAA8A0  | F34C60E4 | D9C9F5F0 | 0BA4CA64 | 000002DE | 0BA892F0 | 0BA89C38 | 03274F21 | 34302024 | *3<-URI50.u.....yk0.y.....         |
| 0BAAA8C0  | C7C5E3C6 | C5C5C440 | 0BA86728 | 00000000 | 00000000 | 00000000 | 03274F21 | 34302024 | *GETFEED .y.....                   |
| 0BAAA8E0  | 00000000 | 00000280 | 00024000 | 00024000 | 20032816 | 00024000 | 00024000 | 00023000 | *.....                             |
| 0BAAA900  | 00002000 | 80500000 | 0BA01138 | 0BC6A000 | 00000280 | 00000100 | C7C5E340 | 40400000 | *....&.....F.....GET ..*           |
| 0BAAA920  | C4E2D7E4 | D9C9F0F0 | 0BA86728 | 00000000 | 00000000 | 00000000 | 03274F21 | 34302024 | *DSPUR100.y.....                   |
| 0BAAA940  | C5D5C4C9 | D9C3C1D9 | F1F20280 | D3002000 | D9C5C3D6 | D9C440E6 | C1E240C6 | D6E4D5C4 | *ENDIRCAR12..L...RECORD WAS FOUND* |
| 0BAAA960  | 00000000 | 00000000 | 00000000 | 00000000 | 01000000 | 00000000 | 00000000 | 00000000 | *.....                             |
| 0BAAA980  | F24C60E4 | D9C9F0F0 | 0BA6FF4A | 0000084E | 0BA88C48 | 0BA892F0 | 03274F21 | 34302024 | *2<-URI00.w.....+y...yk0...        |
| 0BAAA9A0  | 606EF3E4 | D9C9F0F0 | 0000071E | 0BA4AE18 | 0BA88C48 | 0BA89C38 | 03274F21 | 34302024 | *->3URI00....u...y...yk0...        |
| 0BAAA9C0  | C4E2D7E4 | D9C9F0F0 | 0BA86728 | 00000000 | 00000000 | 00000000 | 03274F21 | 34302024 | *DSPUR100.y.....                   |
| 0BAAA9E0  | D4D6C4C9 | D9C3C1D9 | F1F20280 | D4002000 | C3C8C1D5 | C7C540D6 | D3C440D9 | C5C3D9C4 | *MODIRCAR12..M...CHANGE OLD RECRD* |
| 0BAAAA00  | 00000000 | 00000000 | 00000000 | 00000000 | 01000000 | 00000000 | 00000000 | 00000000 | *.....                             |
| 0BAAAA20  | 606EF4E4 | D9C9F5F0 | 00001C4C | 0BA58C68 | 0BA892F0 | 0BA89C38 | 03274F21 | 34302024 | *->4URI50...<.v...yk0.y.....       |
| 0BAAAA40  | C4E2D7E2 | E3C1C3D2 | 8BA58E0C | 0BA58C68 | 0BA89C38 | 0BA89F58 | 03274F21 | 34302024 | *DSPSTACK.v...v...y...y.....       |
| 0BAAAA60  | C7C5E3C6 | C5C5C440 | 0BA86728 | 00000000 | 00000000 | 00000000 | 03274F21 | 34302024 | *GETFEED .y.....                   |
| 0BAAAA80  | 00000000 | 00000280 | 00024000 | 00024000 | 20032816 | 00024000 | 00024000 | 00023000 | *.....                             |
| 0BAAAAA0  | 00002000 | 80500000 | 0BA01138 | 0BAB059C | 00000280 | 00000100 | C7C5E340 | 40400000 | *....&.....GET ..*                 |
| 0BAAAAC0  | F34C60E4 | D9C9F5F0 | 0BA4CA64 | 000002DE | 0BA892F0 | 0BA89C38 | 03274F21 | 34302025 | *3<-URI50.u.....yk0.y.....         |
| 0BAAAAE0  | C7C5E3C6 | C5C5C440 | 0BA86728 | 00000000 | 00000000 | 00000000 | 03274F21 | 34302025 | *GETFEED .y.....                   |
| 0BAAAAB0  | 00000000 | 00000280 | 00024000 | 00024000 | 20032816 | 00024000 | 00024000 | 00023000 | *.....                             |
| 0BAAAAB20 | 00002000 | 80500000 | 0BA01138 | 0BC6A000 | 00000280 | 00000100 | C7C5E340 | 40400000 | *....&.....F.....GET ..*           |
| 0BAAAAB40 | 606EF4E4 | D9C9F5F0 | 00003F98 | 0BA58C68 | 0BA892F0 | 0BA89C38 | 03274F21 | 34302025 | *->4URI50....q.v...yk0.y.....      |
| 0BAAAAB60 | C4E2D7E2 | E3C1C3D2 | 8BA58E0C | 0BA58C68 | 0BA89C38 | 0BA89F58 | 03274F21 | 34302025 | *DSPSTACK.v...v...y...y.....       |
| 0BAAAAB80 | 606EF5E4 | D9C9F5F0 | 00001B66 | 0BA58C68 | 0BA89C38 | 0BAD3010 | 03274F21 | 34302025 | *->5URI50....v...y.....            |
| 0BAAAABA0 | C4E2D7E2 | E3C1C3D2 | 8BA58E0C | 0BA58C68 | 0BAD3010 | 0BAD3330 | 03274F21 | 34302025 | *DSPSTACK.v...v...y...y.....       |
| 0BAAAABC0 | C7C5E3C6 | C5C5C440 | 0BA86728 | 00000000 | 00000000 | 00000000 | 03274F21 | 34302025 | *GETFEED .y.....                   |
| 0BAAAABE0 | 00000000 | 00000280 | 00024000 | 00024000 | 20032816 | 00024000 | 00024000 | 00023000 | *.....                             |
| 0BAAAAC00 | 00002000 | 80500000 | 0BA01138 | 0BA89DBC | 00000280 | 00000100 | C7C5E340 | 40400000 | *....&.....y.....GET ..*           |
| 0BAAAAC20 | F44C60E4 | D9C9F5F0 | 0BA5A7CE | 000002DE | 0BA89C38 | 0BAD3010 | 03274F21 | 34302025 | *4<-URI50.vx.....y.....            |
| 0BAAAAC40 | C7C5E3C6 | C5C5C440 | 0BA86728 | 00000000 | 00000000 | 00000000 | 03274F21 | 34302112 | *GETFEED .y.....                   |
| 0BAAAAC60 | 00000000 | 00000280 | 00024000 | 00024000 | 20032816 | 00024000 | 00024000 | 00023000 | *.....                             |
| 0BAAAAC80 | 00002000 | 80500000 | 0BA01138 | 0BACFFA0 | 00000280 | 00000101 | D7E4E340 | 40400000 | *....&.....PUT ..*                 |
| 0BAAAACA0 | 606EF5E4 | D9C9F5F0 | 00001B66 | 0BA58C68 | 0BA89C38 | 0BAD3010 | 03274F21 | 34302113 | *->5URI50....v...y.....            |
| 0BAAAACC0 | C4E2D7E2 | E3C1C3D2 | 8BA58E0C | 0BA58C68 | 0BAD3010 | 0BAD3330 | 03274F21 | 34302113 | *DSPSTACK.v...v...y...y.....       |
| 0BAAAACE0 | C7C5E3C6 | C5C5C440 | 0BA86728 | 00000000 | 00000000 | 00000000 | 03274F21 | 34302113 | *GETFEED .y.....                   |
| 0BAAAAD00 | 00000000 | 00000280 | 00024000 | 00024000 | 20032816 | 00024000 | 00024000 | 00023000 | *.....                             |
| 0BAAAAD20 | 00002000 | 80500000 | 0BA013B8 | 0BA89DBC | 00000280 | 00000200 | C7C5E340 | 40400000 | *....&.....y.....GET ..*           |
| 0BAAAAD40 | F44C60E4 | D9C9F5F0 | 0BA5A7CE | 000002DE | 0BA89C38 | 0BAD3010 | 03274F21 | 34302113 | *4<-URI50.vx.....y.....            |
| 0BAAAAD60 | C7C5E3C6 | C5C5C440 | 0BA86728 | 00000000 | 00000000 | 00000000 | 03274F21 | 34302133 | *GETFEED .y.....                   |

Figure 172. Example of Internal Trace Table Entries (Part 5 of 7)

|          |          |          |          |          |          |          |          |          |                                    |
|----------|----------|----------|----------|----------|----------|----------|----------|----------|------------------------------------|
| 0BAAAD80 | 00000000 | 00000280 | 00024000 | 00024000 | 20032816 | 00024000 | 00024000 | 00023000 | *.....*                            |
| 0BAAADA0 | 00002000 | 80500000 | 0BA013B8 | 0BAEDFA0 | 00000280 | 00000201 | D7E4E340 | 40400000 | *....&.....PUT **                  |
| 0BAAADC0 | F34C60E4 | D9C9F5F0 | 0BA4EDB0 | 000002DE | 0BA892F0 | 0BA89C38 | 03274F21 | 34302133 | *3<-URI50.u.....yk0.y....*         |
| 0BAAADE0 | F24C60E4 | D9C9F0F0 | 0BA6FF9E | 0000084E | 0BA88C48 | 0BA892F0 | 03274F21 | 34302133 | *2<-URI00.w.....+y...yk0...*       |
| 0BAAAE00 | F14C60E4 | D9E4D7C4 | 0BA066C8 | 00000202 | 0BA88010 | 0BA88C48 | 03274F21 | 34302133 | *1<-URUPD...H.....y...y....*       |
| 0BAAAE20 | 606EF2E4 | D9C9F0F0 | 00000362 | 0BA4AE18 | 0BA88010 | 0BA88C48 | 03274F21 | 34302133 | *->2URI00.....u...y...y....*       |
| 0BAAAE40 | C4E2D7E4 | D9C9F0F0 | 0BA86728 | 00000000 | 00000000 | 00000000 | 03274F21 | 34302134 | *DSPURI00.y.....*                  |
| 0BAAAE60 | D4D6C4C9 | D9C3C1D9 | F1F201D8 | C3002000 | 40D3D6C7 | C9C3C1D3 | 40C3D3D6 | E2C54040 | *MODIRCAR12.QC... LOGICAL CLOSE *  |
| 0BAAAE80 | 40404040 | 40404040 | 40404040 | 40404040 | 40404040 | 40404040 | 00000000 | 00000000 | *.....*                            |
| 0BAAAEA0 | 606EF3E4 | D9C9F3F0 | 000020B0 | 0BA53474 | 0BA88C48 | 0BA89590 | 03274F21 | 34302134 | *->3URI30.....v...y...yn....*      |
| 0BAAAEEO | F24C60E4 | D9C9F3F0 | 0BA4CEC8 | 00000EB2 | 0BA88C48 | 0BA89590 | 03274F21 | 34302168 | *2<-URI30.u.H.....y...yn....*      |
| 0BAAAEFO | 606EF3C4 | C5D8F0F0 | 000042F4 | 0BA0DA68 | 0BA88C48 | 0BA89590 | 03274F21 | 34302168 | *->3DEQ00...4.....y...yn....*      |
| 0BAAAF00 | F24C60C4 | C5D8F0F0 | 0BA4F10C | 000007EE | 0BA88C48 | 0BA89590 | 03274F21 | 34302170 | *2<-DEQ00.u1.....y...yn....*       |
| 0BAAAF20 | F14C60E4 | D9C9F0F0 | 0BA05B3A | 0000084E | 0BA88010 | 0BA88C48 | 03274F21 | 34302170 | *1<-URI00...\$....+y...y....*      |
| 0BAAAF40 | F04C60C1 | D3C4F0F0 | 000096AA | 00000524 | 000061C8 | 0BA88010 | 03274F21 | 34302170 | *0<-ALD00...o...../H.y....*        |
| 0BAAAF60 | C3D9E3D9 | F0E7C9E3 | 09F8A56C | 07052002 | 00001000 | 0BA86728 | 03274F21 | 34302170 | *CRTROXIT.8v%.....y...y....*       |
| 0BAAAF80 | 00000000 | 09FC1900 | 09F8A56C | 0A649490 | 00000000 | C4C6E2C2 | D9D3E2C2 | 00000100 | *.....8v%.m.....DFSBRLSB....*      |
| 0BAAAFAO | 07052002 | 00001000 | 00000000 | 00000000 | 00000000 | 00000000 | 00000000 | 00000000 | *.....*                            |
| 0BAAAFCO | C4E2D7C3 | D9E3D9F0 | 09F8A56C | 070B2002 | 00001000 | 0BA86728 | 03274F21 | 34302257 | *DSPCRTR0.8v%.....y...y....*       |
| 0BAAAFEO | 00000000 | 00000000 | 00CE8E50 | 00000001 | 0BA86728 | 09F89050 | 0A6F2038 | 00000000 | *.....&.....y...y...8.&?....*      |
| 0BAAB000 | 00000000 | 00000000 | 00000000 | 0A0DF728 | 09F8905C | 09F8906C | 00D34748 | 00000000 | *.....7..8.*.8%.L.....*            |
| 0BAAB020 | 606EF1C1 | D3C4F0F0 | 800096C6 | 0BA057D8 | 000061C8 | 0BA88010 | 03274F21 | 34302257 | *->1ALD00...oF...Q.../H.y....*     |
| 0BAAB040 | 606EF2E4 | D9C9F0F0 | 00002A4E | 0BA4AE18 | 0BA88010 | 0BA88870 | 03274F21 | 34302257 | *->2URI00...+u...y...yh....*       |
| 0BAAB060 | C4E2D7E4 | D9C9F0F0 | 0BA86728 | 00000000 | 00000000 | 00000000 | 03274F21 | 34302257 | *DSPURI00.y.....*                  |
| 0BAAB080 | D4D6C4C9 | D9C3C1D9 | F1F201D8 | D6002000 | 40D3D6C7 | C9C3C1D3 | 40D6D7C5 | D5404040 | *MODIRCAR12.QO... LOGICAL OPEN *   |
| 0BAAB0A0 | 40404040 | 40404040 | 40404040 | 40404040 | 40404040 | 40404040 | 00000000 | 00000000 | *.....*                            |
| 0BAAB0C0 | 606EF3D9 | E2E5F0F0 | 00001270 | 0BA80FA0 | 0BA88870 | 0BA891B8 | 03274F21 | 34302257 | *->3RSV00.....y...yh...yj....*     |
| 0BAAB0E0 | F24C60D9 | E2E5F0F0 | 0BA4C088 | 0000022A | 0BA88870 | 0BA891B8 | 03274F21 | 34302310 | *2<-RSV00.u.h.....yh...yj....*     |
| 0BAAB100 | 606EF3E4 | D9C9F3F0 | 00001340 | 0BA53474 | 0BA88870 | 0BA891B8 | 03274F21 | 34302310 | *->3URI30...v...y...yh...yj....*   |
| 0BAAB120 | F24C60E4 | D9C9F3F0 | 0BA4C158 | 00000F8E | 0BA88870 | 0BA891B8 | 03274F21 | 34302317 | *2<-URI30.uA.....y...yh...yj....*  |
| 0BAAB140 | 606EF3E4 | D9C9F5F0 | 00001C4C | 0BA58C68 | 0BA88870 | 0BA891B8 | 03274F21 | 34302317 | *->3URI50...<.v...y...yh...yj....* |
| 0BAAB160 | C4E2D7E2 | E3C1C3D2 | 8BA58E0C | 0BA58C68 | 0BA891B8 | 0BA894D8 | 03274F21 | 34302317 | *DSPSTACK.v...v...yj...ymQ....*    |
| 0BAAB180 | C7C5E3C6 | C5C5C440 | 0BA86728 | 00000000 | 00000000 | 00000000 | 03274F21 | 34302317 | *GETFEED .y.....*                  |
| 0BAAB1A0 | 00000000 | 00000280 | 00024000 | 00024000 | 20032817 | 00024000 | 00024000 | 00023000 | *.....*                            |
| 0BAAB1C0 | 00002000 | 80500000 | 0BA01138 | 0BA8059C | 00000280 | 00000100 | C7C5E340 | 40400000 | *....&.....GET **                  |
| 0BAAB1E0 | F24C60E4 | D9C9F5F0 | 0BA4CA64 | 000002DE | 0BA88870 | 0BA891B8 | 03274F21 | 34302317 | *2<-URI50.u.....yh...yj....*       |
| 0BAAB200 | C7C5E3C6 | C5C5C440 | 0BA86728 | 00000000 | 00000000 | 00000000 | 03274F21 | 34302317 | *GETFEED .y.....*                  |
| 0BAAB220 | 00000000 | 00000280 | 00024000 | 00024000 | 20032817 | 00024000 | 00024000 | 00023000 | *.....*                            |
| 0BAAB240 | 00002000 | 80500000 | 0BA01138 | 0BC74000 | 00000280 | 00000100 | C7C5E340 | 40400000 | *....&.....G.....GET **            |
| 0BAAB260 | F14C60E4 | D9C9F0F0 | 0BA08226 | 0000084E | 0BA88010 | 0BA88870 | 03274F21 | 34302318 | *1<-URI00...b...+y...yh....*       |
| 0BAAB280 | 606EF2E4 | D9C9F0F0 | 00002B9A | 0BA4AE18 | 0BA88010 | 0BA88870 | 03274F21 | 34302318 | *->2URI00.....u...y...yh....*      |
| 0BAAB2A0 | C4E2D7E4 | D9C9F0F0 | 0BA86728 | 00000000 | 00000000 | 00000000 | 03274F21 | 34302318 | *DSPURI00.y.....*                  |
| 0BAAB2C0 | D4D6C4C9 | D9C3C1D9 | F1F201D8 | D3011000 | C3D3C1E2 | E24040D3 | D6C3C1E3 | C540D5E7 | *MODIRCAR12.LL...CLASS LOCATE NX*  |
| 0BAAB2E0 | C4C2D6E5 | D3C6D7C3 | 00000000 | 00000000 | 20000000 | 00000000 | 00000000 | 00000000 | *DBOVLFPCL.....*                   |
| 0BAAB300 | 606EF3E4 | D9C9F5F0 | 00001C4C | 0BA58C68 | 0BA88870 | 0BA891B8 | 03274F21 | 34302318 | *->3URI50...<.v...y...yh...yj....* |
| 0BAAB320 | C4E2D7E2 | E3C1C3D2 | 8BA58E0C | 0BA58C68 | 0BA891B8 | 0BA894D8 | 03274F21 | 34302318 | *DSPSTACK.v...v...yj...ymQ....*    |
| 0BAAB340 | C7C5E3C6 | C5C5C440 | 0BA86728 | 00000000 | 00000000 | 00000000 | 03274F21 | 34302319 | *GETFEED .y.....*                  |
| 0BAAB360 | 00000000 | 000000FE | 00024000 | 00024000 | 20032817 | 00024000 | 00024000 | 00023000 | *.....*                            |
| 0BAAB380 | 00002000 | 80500000 | 0BA01138 | 0BA8059C | 000000FE | 00000100 | C7C5E340 | 40400000 | *....&.....GET **                  |
| 0BAAB3A0 | F24C60E4 | D9C9F5F0 | 0BA4CA64 | 000002DE | 0BA88870 | 0BA891B8 | 03274F21 | 34302319 | *2<-URI50.u.....yh...yj....*       |
| 0BAAB3C0 | C7C5E3C6 | C5C5C440 | 0BA86728 | 00000000 | 00000000 | 00000000 | 03274F21 | 34302319 | *GETFEED .y.....*                  |
| 0BAAB3E0 | 00000000 | 000000FE | 00024000 | 00024000 | 20032817 | 00024000 | 00024000 | 00023000 | *.....*                            |
| 0BAAB400 | 00002000 | 80500000 | 0BA01138 | 0BC73420 | 000000FE | 00000100 | C7C5E340 | 40400000 | *....&.....G.....GET **            |
| 0BAAB420 | C4E2D7E4 | D9C9F0F0 | 0BA86728 | 00000000 | 00000000 | 00000000 | 03274F21 | 34302319 | *DSPURI00.y.....*                  |
| 0BAAB440 | C5D5C4C9 | D9C3C1D9 | F1F200FE | D3011000 | D9C5C3D6 | D9C440E6 | C1E240C6 | D6E4D5C4 | *ENDIRCAR12.LL...RECORD WAS FOUND* |
| 0BAAB460 | C4C2D6E5 | D3C6D7C3 | E5D3D6E2 | C1D4F0F1 | 20000000 | 00000000 | 00000000 | 00000000 | *DBOVLFPCLVLOSAM01.....*           |
| 0BAAB480 | F14C60E4 | D9C9F0F0 | 0BA08372 | 0000084E | 0BA88010 | 0BA88870 | 03274F21 | 34302319 | *1<-URI00...c...+y...yh....*       |
| 0BAAB4A0 | 606EF2E4 | D9C9F0F0 | 00002B9A | 0BA4AE18 | 0BA88010 | 0BA88870 | 03274F21 | 34302319 | *->2URI00.....u...y...yh....*      |
| 0BAAB4C0 | C4E2D7E4 | D9C9F0F0 | 0BA86728 | 00000000 | 00000000 | 00000000 | 03274F21 | 34302319 | *DSPURI00.y.....*                  |
| 0BAAB4E0 | D4D6C4C9 | D9C3C1D9 | F1F201D8 | D3011000 | C3D3C1E2 | E24040D3 | D6C3C1E3 | C540D5E7 | *MODIRCAR12.LL...CLASS LOCATE NX*  |
| 0BAAB500 | C4C2D6E5 | D3C6D7C3 | E5D3D6E2 | C1D4F0F1 | 20000000 | 00000000 | 00000000 | 00000000 | *DBOVLFPCLVLOSAM01.....*           |
| 0BAAB520 | 606EF3E4 | D9C9F5F0 | 00001C4C | 0BA58C68 | 0BA88870 | 0BA891B8 | 03274F21 | 34302319 | *->3URI50...<.v...y...yh...yj....* |
| 0BAAB540 | C4E2D7E2 | E3C1C3D2 | 8BA58E0C | 0BA58C68 | 0BA891B8 | 0BA894D8 | 03274F21 | 34302319 | *DSPSTACK.v...v...yj...ymQ....*    |
| 0BAAB560 | C7C5E3C6 | C5C5C440 | 0BA86728 | 00000000 | 00000000 | 00000000 | 03274F21 | 34302320 | *GETFEED .y.....*                  |
| 0BAAB580 | 00000000 | 000000B0 | 00024000 | 00024000 | 20032817 | 00024000 | 00024000 | 00023000 | *.....*                            |
| 0BAAB5A0 | 00002000 | 80500000 | 0BA01138 | 0BA8059C | 000000B0 | 00000100 | C7C5E340 | 40400000 | *....&.....GET **                  |
| 0BAAB5C0 | F24C60E4 | D9C9F5F0 | 0BA4CA64 | 000002DE | 0BA88870 | 0BA891B8 | 03274F21 | 34302320 | *2<-URI50.u.....yh...yj....*       |
| 0BAAB5E0 | C7C5E3C6 | C5C5C440 | 0BA86728 | 00000000 | 00000000 | 00000000 | 03274F21 | 34302320 | *GETFEED .y.....*                  |
| 0BAAB600 | 00000000 | 000000B0 | 00024000 | 00024000 | 20032817 | 00024000 | 00024000 | 00023000 | *.....*                            |

Figure 172. Example of Internal Trace Table Entries (Part 6 of 7)

```

0BAAB620 00002000 80500000 0BA01138 0BC70000 000000B0 00000100 C7C5E340 40400000 *.....&.....G.....GET ...*
0BAAB640 606EF3E4 D9C9F5F0 00001C4C 0BA58C68 0BA88870 0BA891B8 03274F21 34302320 *->3URI50...<.v...yh..yj...|.....*
0BAAB660 C4E2D7E2 E3C1C3D2 8BA58E0C 0BA58C68 0BA891B8 0BA894D8 03274F21 34302321 *DSPSTACK.v...v...yj..ymQ...|.....*
0BAAB680 C7C5E3C6 C5C5C440 0BA86728 00000000 00000000 00000000 03274F21 34302321 *GETFEED .y.....|.....*
0BAAB6A0 00000000 000000428 00024000 00024000 20032817 00024000 00024000 00023000 *.....|.....*
0BAAB6C0 00002000 80500000 0BA01138 0BA8059C 00000428 00000100 C7C5E340 40400000 *.....&.....G.....GET ...*
0BAAB6E0 F24C60E4 D9C9F5F0 0BA4CA64 000002DE 0BA88870 0BA891B8 03274F21 34302321 *2<-URI50.u.....yh..yj...|.....*
0BAAB700 C7C5E3C6 C5C5C440 0BA86728 00000000 00000000 00000000 03274F21 34302321 *GETFEED .y.....|.....*
0BAAB720 00000000 00000428 00024000 00024000 20032817 00024000 00024000 00023000 *.....|.....*
0BAAB740 00002000 80500000 0BA01138 0BC700B0 00000428 00000100 C7C5E340 40400000 *.....&.....G.....GET ...*
0BAAB760 C4E2D7E4 D9C9F0F0 0BA86728 00000000 00000000 00000000 03274F21 34302321 *DSPURI00.y.....|.....*
0BAAB780 C5D5C4C9 D9C9C1D9 F1F20428 D3011000 D9C5C3D6 D9C440E6 C1E240C6 D6E4D5C4 *ENDIRCAR12..L...RECORD WAS FOUND*
0BAAB7A0 C4C5C4C2 C4C4F0F1 C4C4F0F1 C1D9F040 20000000 00000000 00000000 00000000 *DEBDD01DD01AR0 .....*
0BAAB7C0 F14C60E4 D9C9F0F0 0BA08372 0000084E 0BA88010 0BA88870 03274F21 34302321 *1<-URI00..c...+.y...yh...|.....*
0BAAB7E0 606EF2E4 D9C9F0F0 00002C98 0BA4AE18 0BA88010 0BA88870 03274F21 34302321 *->2URI00...q.u...y...yh...|.....*
0BAAB800 C4E2D7E4 D9C9F0F0 0BA86728 00000000 00000000 00000000 03274F21 34302321 *DSPURI00.y.....|.....*
0BAAB820 D4D6C4C9 D9C3C1D9 F1F201D8 C3011000 40D3D6C7 C9C3C1D3 40C3D3D6 E2C54040 *MODIRCAR12.QC... LOGICAL CLOSE *
0BAAB840 40404040 40404040 40404040 40404040 40404040 40404040 00000000 00000000 *.....*
0BAAB860 606EF3C4 C5D8F0F0 000042F4 0BA0DA68 0BA88870 0BA891B8 03274F21 34302321 *->3DEQ00...4.....yh..yj...|.....*
0BAAB880 F24C60E4 C5D8F0F0 0BA4F10C 000007EE 0BA88870 0BA891B8 03274F21 34302322 *2<-DEQ00.u1.....yh..yj...|.....*
0BAAB8A0 F14C60E4 D9C9F0F0 0BA08470 0000084E 0BA88010 0BA88870 03274F21 34302322 *1<-URI00..d...+.y...yh...|.....*
0BAAB8C0 F04C60C1 D3C4F0F0 000096C6 00000524 000061C8 0BA88010 03274F21 34302322 *0<-ALD00..oF...../H.y...|.....*
0BAAB8E0 C3D9E3D9 F0E7C9E3 09F8A56C 070B2002 00001000 0BA86728 03274F21 34302322 *CRTR0XIT.8v%.....y...|.....*
0BAAB900 00000000 09FC1900 09F8A56C 0A649490 00000000 C4C6E2C2 D9D3E2C2 00000100 *.....8v%.m.....DFSBRLSB...*
0BAAB920 070B2002 00001000 00000000 00000000 00000000 00000000 00000000 00000000 *.....*
0BAAB940 C4E2D7C3 D9E3D9F0 00D26F10 17882002 00000000 0BA86728 03274F21 37275946 *DSPCRTR0.K?.h.....y...|.....*
0BAAB960 00000000 00000000 00000000 00000001 0BA86728 00000000 00000000 00000000 *.....y.....*
0BAAB980 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 *.....*
0BAAB9A0 606EF1D7 E3D2D6E5 8000997E 0BA1D570 000061C8 0BA88010 03274F21 37275946 *->1PTKOV...r=..N.../H.y...|.....*
0BAAB9C0 606EF2E4 D9C9F0F0 00000218 0BA4AE18 0BA88010 0BA88260 03274F21 37275947 *->2URI00.....u...y...yb-...|.....*
0BAAB9E0 C4E2D7E4 D9C9F0F0 0BA86728 00000000 00000000 00000000 03274F21 37275947 *DSPURI00.y.....|.....*
0BAABA00 D4D6C4C9 D9C3C1D9 F1F201D8 D6011000 40D3D6C7 C9C3C1D3 40D6D7C5 D5404040 *MODIRCAR12.QO... LOGICAL OPEN *
0BAABA20 40404040 40404040 40404040 40404040 40404040 40404040 00000000 00000000 *.....*
0BAABA40 606EF3D9 E2E5F0F0 00001270 0BA80FA0 0BA88260 0BA888A8 03274F21 37275947 *->3RSV00.....y...yb-.y.y...|.....*
0BAABA60 C2C7D5C3 C1C2D5F0 0BA86728 00000000 00000000 00000000 03274F21 37276626 *BGN CABN0.y.....|.....*
0BAABA80 C4E2D7C3 C1C2D5F0 0BA86728 00000000 00000000 00000000 03274F21 37277116 *DSCABN0.y.....|.....*
0BAABAA0 E3D9C1C3 C5D5E7E3 40404040 40404040 40404040 40404040 40404040 *TRACENXT *
0BAABAC0 40404040 40404040 40404040 40404040 40404040 40404040 40404040 * *

```

Figure 172. Example of Internal Trace Table Entries (Part 7 of 7)

**Note:** Lines 0BAABAE0-0BAAFF1F same as the above.

## 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 173 shows the format of these records.

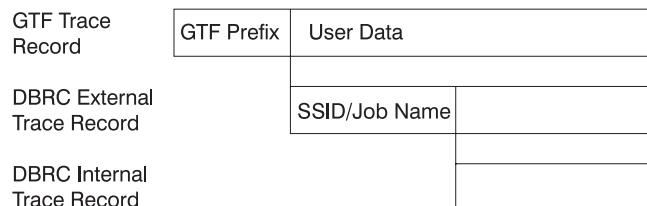


Figure 173. Format of Trace Records

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 *z/OS MVS 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 AMDUSRF2 on this subcommand. For detailed information about using IPCS, see *OS/390 MVS IPCS User's Guide*.

- | The following topics provide additional information:
- | • “Examples of Output”
- | • “Samples of JCL to Create Trace Output” on page 471

## 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 174:

- 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 174. DBRC External Trace Output for DBRC Router Processing

In Figure 175 on page 471 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.7.
+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
DBRCJOB1 TIME=99085F2248398254 DSPURI00 GDB=00012D78 FUNC=LOCATE FLAG=0020
RECKEY=FFFFFFFFFFFFFFFFC9D4E2F1404040403F00000000000000000000000000000000
```

Figure 175. 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:

```
//PRTUSR F2 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=IMSB LD.I710TS25.CRESLIB
/* DD DISP=SHR,DSN=IMSTESTG.IMS710.TSTRES
/* DD DISP=SHR,DSN=IMSTESTG.IMSQA.ACPLIB
/* DD DISP=SHR,DSN=IMSTESTG.IMSQA.PGMLIB
//JOB CAT 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 *
/* the Batch IPCS feature.                                       *
//*****
/*ROUTE PRINT THISCPU/IMSM3405
//JOBLIB  DD DSN=IMSTESTL.TNUC0,DISP=SHR
//        DD DISP=SHR,DSN=IMSB LD.I710TS25.CRESLIB
//        DD DISP=SHR,DSN=IMSTESTG.IMS710.TSTRES
//        DD DISP=SHR,DSN=IMSTESTG.IMSQA.ACPLIB
//        DD DISP=SHR,DSN=IMSTESTG.IMSQA.PGMLIB
//JOB CAT 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(AMDUSRF2)
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)
/*
```

---

## DBRC API Return and Reason Codes

This topic contains the internal return and reason codes needed to diagnose DBRC API problems.

### Return and Reason Codes for the TYPE=BACKOUT Query Request

Table 113 on page 473 contains the return and reason codes for TYPE=BACKOUT query requests.

Table 113. Return and Reason Codes for TYPE=BACKOUT Query Requests

| Return Codes | Reason Codes | Meaning                                                                                      |
|--------------|--------------|----------------------------------------------------------------------------------------------|
| X'0000002C'  | X'D8000001'  | DBRC internal error. RECON open failure.                                                     |
| X'0000002C'  | X'D8700001'  | DBRC internal error. Failure attempting to locate the first or the specified backout record. |
| X'0000002C'  | X'D8700002'  | DBRC internal error. Failure attempting to locate the next backout record.                   |

**Return and Reason Codes for TYPE=DB Query Requests**

Table 114 contains the internal return and reason codes associated with the TYPE=DB query requests.

Table 114. Return and Reason Codes for TYPE=DB Query Requests

| Return Codes | Reason Codes | Meaning                                                                                                              |
|--------------|--------------|----------------------------------------------------------------------------------------------------------------------|
| X'0000002C'  | X'D8000001'  | DBRC internal error. RECON open failure.                                                                             |
| X'0000002C'  | X'D8200001'  | DBRC internal error. DB record locate failure processing DBLIST.                                                     |
| X'0000002C'  | X'D8200002'  | DBRC internal error. DB record locate failure processing single database request.                                    |
| X'0000002C'  | X'D8210001'  | DBRC internal error. Failure attempting to locate the first DBDS record.                                             |
| X'0000002C'  | X'D8210002'  | DBRC internal error. Failure attempting to locate the specified DBDS record.                                         |
| X'0000002C'  | X'D8210003'  | DBRC internal error. Failure attempting to locate the next DBDS record.                                              |
| X'0000002C'  | X'D8210004'  | DBRC internal error. Locate failure attempting to locate the first Area Auth record.                                 |
| X'0000002C'  | X'D8210005'  | DBRC internal error. Failure attempting to locate the first ALLOC record.                                            |
| X'0000002C'  | X'D8210006'  | DBRC internal error. Failure attempting to locate the next ALLOC record.                                             |
| X'0000002C'  | X'D8210007'  | DBRC internal error. Failure attempting to locate the first IC record.                                               |
| X'0000002C'  | X'D8210008'  | DBRC internal error. Failure attempting to locate the next IC record.                                                |
| X'0000002C'  | X'D8210009'  | DBRC internal error. Failure attempting to locate the first REORG record.                                            |
| X'0000002C'  | X'D821000A'  | DBRC internal error. Failure attempting to locate the next REORG record.                                             |
| X'0000002C'  | X'D821000B'  | DBRC internal error. Failure attempting to locate the first RECOV record.                                            |
| X'0000002C'  | X'D821000C'  | DBRC internal error. Failure attempting to locate the next RECOV record.                                             |
| X'0000002C'  | X'D8220001'  | DBRC internal error. Failure attempting to locate the first HALDB partition record.                                  |
| X'0000002C'  | X'D8220002'  | DBRC internal error. Failure attempting to locate the DB record associated with the HALDB partition being processed. |
| X'0000002C'  | X'D8220003'  | DBRC internal error. Failure attempting to locate the next HALDB partition record.                                   |

**Return and Reason Codes for the TYPE=xxxxGROUP Query Request**

Table 115 contains the return and reason codes for TYPE=xxxxGROUP query requests.

*Table 115. Return and Reason Codes for TYPE=xxxxGROUP Query Requests*

| Return Codes | Reason Codes | Meaning                                                                                                                          |
|--------------|--------------|----------------------------------------------------------------------------------------------------------------------------------|
| X'0000002C'  | X'D8000001'  | DBRC internal error. RECON open failure.                                                                                         |
| X'0000002C'  | X'D8300001'  | DBRC internal error. Failure attempting to locate a specific group record or the first group record of the requested group type. |
| X'0000002C'  | X'D8300002'  | DBRC internal error. Failure attempting to locate the next group record of the requested group type.                             |
| X'0000002C'  | X'D8310001'  | DBRC internal error. Failure attempting to locate a CA record.                                                                   |

**Return and Reason Codes for the TYPE=LOG Query Request**

Table 116 contains the return and reason codes for TYPE=BACKOUT query requests.

*Table 116. Return and Reason Codes for TYPE=LOG Query Requests*

| Return Codes | Reason Codes | Meaning                                                                                                                           |
|--------------|--------------|-----------------------------------------------------------------------------------------------------------------------------------|
| X'0000002C'  | X'D8000001'  | DBRC internal error. RECON open failure.                                                                                          |
| X'0000002C'  | X'D8400001'  | DBRC internal error. Failure attempting to locate the previous or next log record of the requested log type - PRILOG or PRITSLDS. |
| X'0000002C'  | X'D8400002'  | DBRC internal error. Failure attempting to locate the specified log record of the requested log type - PRILOG or PRITSLDS.        |
| X'0000002C'  | X'D8400003'  | DBRC internal error. Failure attempting to locate the corresponding SECTSLDS record.                                              |
| X'0000002C'  | X'D8400004'  | DBRC internal error. Failure attempting to locate the LOGALL record that corresponds to the PRILOG record.                        |
| X'0000002C'  | X'D8400005'  | DBRC internal error. The LOGALL record that corresponds to the PRILOG record does not exist.                                      |
| X'0000002C'  | X'D8400006'  | DBRC internal error. Failure attempting to locate the corresponding SECLOG record.                                                |
| X'0000002C'  | X'D8400007'  | DBRC internal error. Failure attempting to locate the corresponding PRISLDS record.                                               |
| X'0000002C'  | X'D8400008'  | DBRC internal error. No PRISLDS record exists for the online log.                                                                 |
| X'0000002C'  | X'D8400009'  | DBRC internal error. Failure attempting to locate the corresponding SECSLDS record.                                               |

**Return and Reason Codes for the TYPE=OLDS Query Request**

The following table contains the return and reason codes for TYPE=OLDS query requests.

*Table 117. Return and Reason Codes for TYPE=OLDS Query Requests*

| Return Codes | Reason Codes | Meaning                                                                                      |
|--------------|--------------|----------------------------------------------------------------------------------------------|
| X'0000002C'  | X'D8000001'  | DBRC internal error. RECON open failure.                                                     |
| X'0000002C'  | X'D8500001'  | DBRC internal error. Failure attempting to locate the first or the specified PRIOLDS record. |
| X'0000002C'  | X'D8500002'  | DBRC internal error. Failure attempting to locate the corresponding SECOLDS record.          |

Table 117. Return and Reason Codes for TYPE=OLDS Query Requests (continued)

| Return Codes | Reason Codes | Meaning                                                                    |
|--------------|--------------|----------------------------------------------------------------------------|
| X'0000002C'  | X'D8500003'  | DBRC internal error. Failure attempting to locate the next PRIOLDS record. |

**TYPE=RECON Query Requests**

Table 118 contains the internal return and reason codes associated with the TYPE=RECON query requests.

Table 118. Return and Reason Codes for TYPE=RECON Query Requests

| Return Codes | Reason Codes | Meaning                                                                    |
|--------------|--------------|----------------------------------------------------------------------------|
| X'0000002C'  | X'D8000001'  | DBRC internal error. RECON open failure.                                   |
| X'0000002C'  | X'D8100001'  | DBRC internal error. Failure attempting to locate the RECON header record. |

**Return and Reason Codes for the TYPE=SUBSYS Query Request**

Table 119 contains the return and reason codes for TYPE=SUBSYS query requests.

Table 119. Return and Reason Codes for TYPE=SUBSYS Query Requests

| Return Codes | Reason Codes | Meaning                                                                                                                                 |
|--------------|--------------|-----------------------------------------------------------------------------------------------------------------------------------------|
| X'0000002C'  | X'D8000001'  | DBRC internal error. RECON open failure.                                                                                                |
| X'0000002C'  | X'D8600001'  | DBRC internal error. Failure attempting to locate the first or the specified subsystem record of the requested type, active or tracker. |
| X'0000002C'  | X'D8600002'  | DBRC internal error. Failure attempting to locate the next subsystem record of the requested type, active or tracker.                   |



## 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 section provides service aids and tips that can help you analyze problems in a Database Control (DBCTL) environment. It discusses:

- “DRA Dumps”
- “Analyzing DRA Problems” on page 478

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 9: 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 a z/OS 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:

```
hhssuuu
```

where hh indicates the type of dump.

Table 120 lists the values for *hh* and tells which dump the DRA creates for different types of failures.

Table 120. 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 z/OS 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: <ul style="list-style-type: none"> <li>• IMS abend codes U0775, U0777, U2478, U2479, U3303</li> <li>• z/OS nonretryable abend codes (for example, S222, S13E)</li> <li>• DRA return codes (See <i>IMS Version 9: Messages and Codes, Volume 1</i> for DRA return codes and their meanings.)</li> </ul> |

The following topics provide additional information:

- “SDUMP” on page 478
- “SNAPs” on page 478
- “Recovery Tokens” on page 478

## 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 154. The ODF does not format DRA storage. You can use IPCS to format the z/OS 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 is shown in Table 121:

Table 121. Recovery Token Format

|                                       |                                                              |
|---------------------------------------|--------------------------------------------------------------|
| CCTL subsystem ID<br>8 bytes (EBCDIC) | Unique UOR ID (created by the CCTL)<br>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.

The following topics provide additional information:

- "Procedure"
- "Notes on Dumping" on page 479

## 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 9: Customization Guide*. For information about the codes passed between the DRA and the CCTL, see *IMS Version 9: 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 9: Messages and Codes, Volume 1*.
    - If yes, take a z/OS 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 section in this manual to analyze the problem. The keyword procedures in Chapter 4, “Selecting the Keywords,” on page 31 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 section provides Fast Path Tracker Trace Entries (“Fast Path Tracker Trace Entries” on page 483) and Database Tracker Trace Entries (“X'D4’: Database Tracker Trace Entries (D4)” on page 489) that might help you analyze problems in a Remote Site Recovery (RSR) Environment.

Included in this section are:

- “Determining Last Non-MSD Message Recorded”
- “Determining Last MSD Message Recorded” on page 483
- “Fast Path Tracker Trace Entries” on page 483
- “Log Router Trace Data” on page 492

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 MSD links might result in resynchronization errors.

The MTO is notified of both non-MSD errors and MSD errors. as follows:

- Message DFS2948 notifies the MTO of non-MSD errors.
- Either message DFS3211 or message DFS3212 notifies the MTO of MSD errors.

Use the remote takeover message information in conjunction with the received log data to determine the last terminal or MSD message recorded by the tracking process. Then input or output any messages that were lost.

---

### Determining Last Non-MSD Message Recorded

#### Non-MSD, Non-Fast Path Messages

For a non-MSD, non-Fast Path message, use the following procedure to determine the last input or output message recorded using 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 is 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 using 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 is 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 is 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 can result in the loss or duplication of messages. This can 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 might 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

Use Table 122 through Table 139 on page 489 to analyze the Fast Path Tracker Trace entries.

### Trace Entry: Fast Path Tracker Log Router Interface (9E)

#### 9E01

Table 122. 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:

```

DT210 Entry      9E018A65      LSN      streamID      OFRID
                   |          |          |          |
                   000023AB    00000001    00000000
    
```

```

00000090 0094122F 1141138F 8613CD64
|         |
milestone prilog time
index
    
```

**Trace Entry: Fast Path Tracker Log Router Interface (9F)**

**9F22**

Table 123. 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:**

```

DT300 Entry      enqueueer  open area  EMAC
                 |         |         |
                 9F22B879  04F9E5E2  00000003  0476A3C0
                 00000001  00000002  00000000  A0AFD862
                 |         |
                 streamID  USID
    
```

**Trace Entry: Fast Path Tracker Log Router Interface (9F)**

**9F22**

Table 124. 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:**

```

DT300 Entry      enqueueer  open area  EMAC
                 |         |         |
                 9F22B879  04F9E5E2  00000003  0476A3C0
                 00000001  00000002  00000000  A0AFD862
                 |         |
                 streamID  USID
    
```

**9F41**

Table 125. 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

Table 125. Trace Record 9F41 - DBFDT180 Area Status Change Module Entry (continued)

| 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:**

```

DT180 Entry      9F41D6C0  00000001  00000007  041843C0
                  00B3C060  00000000  00000000  AC97BB2C
                  |
                  PST
                stop_req | ndtrk_fail | EMAC
    
```

**9F44**

Table 126. 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 127. 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 128. 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 129. Trace Record 9F50 - DBFDT350 IPOST

**Module:** DBFDT350 IPOST







**Example:**

```

                                Log Record ID
                                |
DT350 ERQE2    9F558AA5  00000000  000023B3  00000000
                00000000  00000000  00000000  8613F10D
    
```

**9F70**

Table 135. 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 136. 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 137. Trace Record 9F72 - DBFDT400 EMAC

**Module:** DBFDT400 EMAC

**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 138. 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 |

Table 138. Trace Record 9F73 - DBFDT400 Read (continued)

|    |         |   |                |
|----|---------|---|----------------|
| 8  | Fixed   | 4 | DMHRSRBA       |
| 12 | Address | 4 | DMHRDMAC       |
| 16 | Fixed   | 4 | IOTIERQE Count |
| 20 | Address | 4 | IOTIEMAC       |

**Example:**

```

DT400 Read      9F7374E5  02F553A0  0316A860  00014000
                 028F7E28  00000002  02867040  8FA4571B
                 |          |          |          |
                 DMAC      IOTIERQE count  EMAC
    
```

**9F74**

Table 139. 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:**

```

DT400 Write      9F7474DC  02F553A0  0316AD38  00013000
                 028F7E28  02867040  00000000  8FA06CE5
                 |          |          |          |
                 DMAC      EMAC
    
```

**X'D4': Database Tracker Trace Entries (D4)**

Table 140 shows the database tracking trace entries for the X'D4' trace entry.

Table 140. Database Tracking Trace Entries for X'D4' Trace Entry

| Word 1, first half           | Word 1, second half | Word 2 | Word 3 | Word 4 | Word 5                                     | Word 6                                           | Word 7 |
|------------------------------|---------------------|--------|--------|--------|--------------------------------------------|--------------------------------------------------|--------|
| X'0001': DRQE queued on DRWQ | Stream id           | TDBC   | DRQE   | DRWQ   | Log sequence number (LSN), right half only | Prilog start time. In the format: yyyyddFhhmsstt |        |

Table 140. Database Tracking Trace Entries for X'D4' Trace Entry (continued)

| Word 1, first half                   | Word 1, second half            | Word 2               | Word 3          | Word 4                      | Word 5                                     | Word 6                                      | Word 7 |
|--------------------------------------|--------------------------------|----------------------|-----------------|-----------------------------|--------------------------------------------|---------------------------------------------|--------|
| X'0002': DRQE queued on TDBC         | Stream id                      | TDBC                 | DRQE            | DRWQ                        | Log sequence number (LSN), right half only | Prilog start time. Format: yyyydddFhhmssstt |        |
| X'0003': DRQE freed without tracking | Stream id                      | TDBC                 | DRQE            |                             | Log sequence number (LSN), right half only | Prilog start time. Format: yyyydddFhhmssstt |        |
| X'0004': DBTI                        | X'0001': Dispatched for work   | PST                  | DTT             | DTTPCTL                     |                                            |                                             |        |
| X'0005': DFSDT240 AWE                | AWE function <sup>1</sup>      | TDBC                 | AWE             |                             |                                            |                                             |        |
| X'0006': DFSDT300 AWE                | AWE function <sup>2</sup>      | TDBC                 | AWE             | Return code                 |                                            |                                             |        |
| X'0007': Shutdown                    | X'0030': DFSDT300              |                      |                 |                             |                                            |                                             |        |
|                                      | X'0040': DFSDT400              | PST                  | DTT             |                             |                                            |                                             |        |
|                                      | X'0050': DFSDT500              | PST                  |                 |                             |                                            |                                             |        |
| X'0008': DB stop                     | Function <sup>3</sup>          | Reason               | TDBC            | ECB                         |                                            |                                             |        |
| X'0009': Milestone                   | X'0000': Prepare for milestone | Type code            | New index       |                             |                                            |                                             |        |
|                                      | X'0001': Begin milestone       | Type code            | New index       |                             |                                            |                                             |        |
|                                      | X'0002': End milestone         | Type code            |                 |                             |                                            |                                             |        |
|                                      | X'0003': BQEL transfer done    | PST                  | PSTFNCTN        | Milestone index transferred |                                            |                                             |        |
|                                      | X'0004': Buffer purge done     | PST                  | PSTFNCTN        | Milestone index purged      |                                            |                                             |        |
| X'000A': End stream                  | Stream type                    | Stream id            | Milestone index |                             |                                            |                                             |        |
| X'000B': Load balance                | X'0000': DTT statistics        | DTT                  | Busy percent    | DTTWAIT                     | DTTWORK                                    | DTTPCTL                                     |        |
|                                      | X'0001': Summary               | Average busy percent | Active DBTIs    | Backlog                     |                                            |                                             |        |
|                                      | X'0002': DRWQ assign           | New DTT              | DRWQ            | Q busy percent              | Old DTT                                    |                                             |        |

Table 140. Database Tracking Trace Entries for X'D4' Trace Entry (continued)

| Word 1, first half                                | Word 1, second half                     | Word 2  | Word 3 | Word 4   | Word 5 | Word 6 | Word 7 |
|---------------------------------------------------|-----------------------------------------|---------|--------|----------|--------|--------|--------|
|                                                   | X'0003':<br>DRWQ<br>assigns<br>complete | Old DTT |        |          |        |        |        |
| X'000C': OFR                                      | X'0000':<br>DFSLROFR<br>called          | OFR id  | OFRL   | DB count |        |        |        |
|                                                   | X'0001':<br>Restart OFR                 | OFR id  | TDBC   | TDBCT    | Flags  |        |        |
|                                                   | X'0002': Begin<br>OFR                   | OFR id  | TDBC   | TDBCT    | Flags  |        |        |
|                                                   | X'0003': End<br>OFR                     | OFR id  | TDBC   | TDBCT    | Flags  |        |        |
|                                                   | X'0004': Begin<br>OFR ignored           | OFR id  | TDBC   | TDBCT    | Flags  |        |        |
|                                                   | X'0005':<br>Restart OFR<br>ignored      | OFR id  | TDBC   | TDBCT    | Flags  |        |        |
| X'000F':<br>Wait/post/resume<br>for routed record | Event <sup>4</sup>                      | TDBC    | ECB    |          |        |        |        |

Notes to Table 140 on page 489:

1. AWE functions for DFSDT240 (trace code X'0005'):

- X'0015' Open/authorize/NUSID for database succeeded
- X'0017' Stream does not apply (open/authorize/NUSID failed)
- X'0018' Process TDBC queue
- X'001A' Add DDIR through online change
- X'001E' Database might need OFR
- X'0020' OLR output data set creation succeeded
- X'0021' OLR output data set creation failed

2. AWE functions for DFSDT300 (trace code X'0006'):

- X'0001' Initialize DLI/SAS queue server
- X'0002' Terminate DLI/SAS queue server
- X'0003' Open/authorize/NUSID for database
- X'0004' Close database
- X'0005' End database tracking, written to disk
- X'0006' End database tracking, initial call
- X'0007' Stream complete, initial call
- X'0008' Stream complete, written to disk
- X'0009' Load balancing
- X'000A' OFR needed

- | X'000B' OFR complete, initial call
- | X'0010' OFR complete, written to disk
- | X'001F' Create OLR output data sets
- | X'0020' OLR cursor active, written to disk
- | X'0021' OLR cursor inactive
- | 3. Functions for DFSDT180, database stop (trace code X'0008'):
  - | X'0000' Initiate database stop
  - | X'0001' Database stop complete
  - | X'0002' Database might need OFR
  - | X'0003' Log the TDBC state
  - | X'0004' Call DFSLRETR to record tracking suspend point
  - | X'0005' Database started
- | 4. Events for wait/post/resume for routed record (X'000F'):
  - | X'0000' Wait because of OFR-pending.
  - | X'0001' Posted because database is up to date already; OFR not needed.
  - | X'0002' Posted because OFR was started.
  - | X'0003' Posted because stop is complete.
  - | X'0004' Posted at end-OFR (should not occur).
  - | X'0005' Posted because database is up to date at restart (should not occur).
  - | X'0006' Posted for unknown reason when starting database (should not occur).
  - | X'0007' Posted because of error starting OFR.
  - | X'0009' Resumption after being posted.

---

## Buffer Handler Trace Entries at Database Tracker

| Trace entries are written for buffer handler calls at the database tracker just as they are at the active site. | These entries are described in “X'DB' through X'FA' Trace Entry” on page 295. However, there are the | following differences in these trace entries written by the database tracker:

- | • Word 3 contains the RBA or RBN of the data in the CI or block rather than PSTDSGA.
- | • Word 6 contains the right half of the log sequence number (LSN) rather than PSTISAMW.

---

## Log Router Trace Data

The log router (LRTT) trace entries are documented in Table 141 through Table 205 on page 518. Field lengths are in bytes.

### Trace Entry: Log Router Data Set Services (370x)

**3701**

*Table 141. 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) |

Table 141. Trace Record 3701 - Data Set Services Control Routine Entry (continued)

|    |           |   |                                      |
|----|-----------|---|--------------------------------------|
| 5  | Fixed     | 1 | AWLGDSFL (DSS Request Code)          |
| 6  | Fixed     | 1 | AWLGDSTP (Data Set Type)             |
|    | 1... ..   | 1 | Tracking_SLDS (AWLGDTRK)             |
|    | .1.. ..   |   | Archive SLDS (AWLGDARC)              |
|    | .1. ....  |   | Archive RLDS (AWLGDRLD)              |
|    | ...1 1111 |   |                                      |
| 7  | Fixed     | 1 | Request Priority (AWLGDPR1)          |
| 8  | Address   | 4 | LTDCB address (AWLGD LTD)            |
| 12 | Address   | 4 | LDSD address (AWLGD LDS)             |
| 16 | Bit       | 4 | DSS Flags (LGBDSSFLAGS)              |
|    | 1... ..   | 1 | LGB_CBTE_ALTERED                     |
|    | .1.. ..   |   | LGBDSS_DUAL_TRACKING_SLDS            |
|    | .1. ....  |   | LGBDSS_DUAL_ARCHIVE_SLDS             |
|    | ...1 ..   |   | LGBDSS_DUAL_ARCHIVE_RLDS             |
|    | ... 1..   |   | LGB_ARCHIVE_SLDS                     |
|    | ... .1..  |   | LGB_ARCHIVE_RLDS                     |
|    | ... ..1.  |   | LGB_INITIALIZEDSS                    |
|    | ... ...1  |   | LGB_TERMINATINGDSS                   |
|    | 1... ..   |   | LGB_DSS_DATASETS_RETURNED            |
|    | .1.. ..   |   | LGB_DSS_RESTART_INIT                 |
|    | ..11 1111 |   | *                                    |
| 20 | Fixed     | 4 | LGB_DATASET_NUMBER                   |
| 24 | Bit       | 2 | Data set Action Flags (AWLGD SAC)    |
|    | 1... ..   |   | Delete data set (AWLGD SDE)          |
|    | .1.. ..   |   | Input/Output (AWLGD SIO)             |
|    | ..1. .... |   | Last active data set (AWLGD LST)     |
|    | ...1 ..   |   | Allocate for restart (AWLGD ARS)     |
|    | ... 1...  |   | 4906 delete record (AWLGD4906)       |
|    | ... .1..  |   | Delete for restart (AWLGD RST)       |
|    | ... ..1.  |   | End stream notification (AWLGD EST)  |
|    | ... ...1  |   | Create prealloc data set (AWLGD LGB) |
| 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 142. Trace Record 3702 - Create Data Set Routine Invoke DY A

Module: DFSLRDCR Data Set Create Routine

Explanation: Invoke DY A 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 (AWLGD LDS) |

### 3703

Table 143. 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)    |

Table 143. Trace Record 3703 - Create Data Set Routine Exit (continued)

|    |           |   |  |                               |
|----|-----------|---|--|-------------------------------|
|    | 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 (AWLGDLTG)      |
| 24 | Address   | 4 |  | LDSD address (AWLGDLDG)       |

**3704**

Table 144. 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 |
| 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 (AWLGDLTG)      |
| 20     | Address | 4      | LDSD address (AWLGDLDG)       |
| 24     | Address | 4      | R13                           |

**3705**

Table 145. 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 (AWLGDSDG)                   |
|        | .1.. .... |        | Input/Output (AWLGDGSI)                      |
|        | ..1. .... |        | Last active data set (AWLGDGSI)              |
|        | ...1 .... |        | Allocate for restart (AWLGDARS)              |
|        | .... 1... |        | 4906 delete record (AWLGD4906)               |
|        | .... .1.. |        | Delete for restart (AWLGDARS)                |
|        | .... ..1. |        | End stream notification (AWLGDGSI)           |
|        | .... ...1 |        | Create prealloc data set (AWLGDGSI)          |
| 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 (AWLGDLTG)                     |
| 16     | Address   | 4      | LDSD address (AWLGDLDG)                      |
| 20     | Address   | 4      | AWE address                                  |
| 24     | Address   | 4      | R13                                          |

**3707**

Table 146. 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) |



Table 146. Trace Record 3707 - Deallocate/Delete Data Set Routine Exit (continued)

|    |            |   |                                      |
|----|------------|---|--------------------------------------|
| 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 (AWLGDLTDCB)           |
| 12 | Address    | 4 | LDS address (AWLGDLDSD)              |
| 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 (AWLGDSD)            |
|    | .1.. ....  |   | Input/Output (AWLGDIO)               |
|    | ..1. ....  |   | Last active data set (AWLGDLAST)     |
|    | ...1 ....  |   | Allocate for restart (AWLGDARS)      |
|    | .... 1...  |   | 4906 delete record (AWLGD4906)       |
|    | .... .1..  |   | Delete for restart (AWLGDARST)       |
|    | .... ..1.  |   | End stream notification (AWLGDDEST)  |
|    | .... ...1  |   | Create prealloc data set (AWLGDALGB) |
| 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                      |

Trace Entry: Log Router Record Router (3709/370E/370F/371x)

3709

Table 147. 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 LRRMRG End Mrg

| Offset | Type  | Length | Description                      |
|--------|-------|--------|----------------------------------|
| 4      | Char  | 8      | Stream subsystem ID              |
| 12     | Char  | 1      | mr_b_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 148. 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                |

Table 148. Trace Record 370E - Received Last Buffer of the Active Stream (continued)

|    |            |   |                          |
|----|------------|---|--------------------------|
|    | 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  |   | *                        |
|    | 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      | 2 | lpd_feedback             |

**370F**

Table 149. 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 150. 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   |

Table 150. Trace Record 3710 - Active Stream Tracker RSR04\_PTKO (continued)

|    |         |   |                        |
|----|---------|---|------------------------|
|    | 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 151. 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          |
| 5      | Char  | 1      | rsr04sub           |
| 8      | Char  | 4      | r04_hipritoken     |
| 12     | Char  | 4      | lrb_record_ID(5-8) |
| 16     | Char  | 8      | r04_pri1gts(1-8)   |
| 24     | Fixed | 31     | lpd_stream_id      |

**Trace Entry: Log Router I/O (373x)**

**3731**

Table 152. 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. |         |        | *                          |
| 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_BFR_CK_OK          |
| .... .111 |         |        |                            |

Table 152. Trace Record 3731 - Stream Archiver Controller Entry (continued)

|    |           |   |                         |
|----|-----------|---|-------------------------|
| 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   |
| 24 | Character | 8 | SAA_PRILOG_TIME         |

**3732**

Table 153. 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                |

Table 153. Trace Record 3732 - Stream Archiver Controller Exit (continued)

|    |           |   |                       |
|----|-----------|---|-----------------------|
|    | .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 154. 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                                                            |
| 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 155. 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** LRSDDS 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_BUF_CK_OK          |
|        | .... .111 |        |                            |
| 12     | Address   | 4      | SAA_LDSD                   |

Table 155. Trace Record 3734 - Stream Archiver Switch Data Set (continued)

|    |           |   |                 |
|----|-----------|---|-----------------|
| 16 | Address   | 4 | SAA_LTDCB       |
| 20 | Character | 4 | AWLG_CSW_LSN    |
| 24 | Character | 4 | LRB_RECORD_ID   |
| 28 | Fixed     | 4 | Switch feedback |

**3736**

Table 156. 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          |
| 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 157. 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             |

Table 157. Trace Record 3737 - Log Router Log Truncation Exit (continued)

|    |                     |   |                          |
|----|---------------------|---|--------------------------|
|    | 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             |
|    | .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 158. 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 159. 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   |

Table 159. Trace Record 373A - Log Router Log Reader First Read 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 | 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 160. 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               |



Table 160. Trace Record 373B - Log Router Log Reader Buffer Return (continued)

|    |           |   |                       |
|----|-----------|---|-----------------------|
|    | 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 161. 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                     |
|        | 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 162. 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   |

Table 162. Trace Record 373D - Log Router Log Reader Exit (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     | 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 163. 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 164. 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 165. 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 166. 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 167. 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 168. 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    |

Table 168. Trace Record 3750 - Initiate Online Forward Recovery (OFR) (continued)

|    |       |   |                  |
|----|-------|---|------------------|
| 20 | Fixed | 4 | DBRC return code |
|----|-------|---|------------------|

**3751**

Table 169. 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      | OFR address       |
| 8      | Address | 4      | OFR address       |
| 12     | Fixed   | 4      | OFR identifier    |
| 16     | Address | 4      | ECB address       |
| 20     | Fixed   | 4      | current OFR count |

**3752**

Table 170. 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      | OFR address                                            |
| 8      | Address   | 4      | OFR address                                            |
| 12     | Address   | 4      | buffer address if AWLGFUNC=002E, AWE address otherwise |
| 16     | Fixed     | 2      | AWLGFUNC                                               |
| 18     | Bit       | 1      | OFR_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 171. 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      | OFR address                                            |
| 8      | Address | 4      | OFR address                                            |
| 12     | Address | 4      | buffer address if AWLGFUNC=002E, AWE address otherwise |
| 16     | Fixed   | 2      | AWLGFUNC                                               |
| 18     | Bit     | 2      | OFR_flags                                              |
| 20     | Fixed   | 4      | OFR identifier                                         |

**3754**

*Table 172. 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          |
| 28     | Character | 4      | LDSD_prilog_time(5:8) |

### 3756

*Table 173. 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 174. Trace Record 3757 - Log Descriptors Obtained from DBRC*

**Module:** DFSLRORM - Online Forward RecoveryRead 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 175. 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 176. 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 177. 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          |
|        | .... ...1 |        | AAB_AR_DUAL         |

### 3761

Table 178. Trace Record 3761 - DFSLRARC Auto Archive Controller Exit

**Module:** DFSLRARC Auto Archive Controller

**Explanation:** Record cut on exit from DFSLRARC (Level - Medium)

**Trace Subcode** LRARC Exit

| Offset | Type  | Length | Description   |
|--------|-------|--------|---------------|
| 4      | Fixed | 2      | Feedback Code |

### 3762

Table 179. 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      | LDSO_FLRID          |
| 16     | Fixed     | 4      | AAB_LDSO_LIST       |
| 20     | Character | 4      | LDSO_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_LDSO_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 180. 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 |
|--------|------|--------|-------------|
|--------|------|--------|-------------|

Table 180. Trace Record 3762 - DFSLRARP Auto Archive Processor Entry (continued)

|    |           |    |                       |
|----|-----------|----|-----------------------|
| 4  | Fixed     | 2  | AWLGFUNC='2E'x        |
| 6  | Character | 2  | *                     |
| 8  | Fixed     | 4  | AAB address           |
| 12 | Character | 4  | LRB_RECORD_ID         |
| 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 181. 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      | *              |



Table 181. Trace Record 3762 - DFSLRARP Auto Archive Processor Entry (continued)

|    |           |   |                       |
|----|-----------|---|-----------------------|
| 16 | Address   | 4 | AWLG_RTBBUFP          |
| 20 | Character | 4 | *                     |
| 24 | Bit       | 4 | AWE's flags           |
|    | 1... .... |   | AWLG_RTB_TRK          |
|    | .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 182. 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 |

Table 182. Trace Record 3762 - DFSLRARP Auto Archive Processor Entry (continued)

|    |           |   |                       |
|----|-----------|---|-----------------------|
|    | .1. ....  |   | AWLG_ADS_ARCHIVE_SLDS |
|    | ..1. .... |   | AWLG_ADS_ARCHIVE_RLDS |
|    | ...1 1111 |   | *                     |
| 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           |

**3763**

Table 183. 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 184. 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 |

Table 184. Trace Record 3764 - DFSLRARP After Create Log Reader (continued)

|    |           |   |                  |
|----|-----------|---|------------------|
| 16 | Fixed     | 4 | AAB_READ_RETQ    |
| 20 | Character | 4 | LDSO_FLRID       |
| 24 | Character | 4 | LDSO_LLRID       |
| 28 | Fixed     | 4 | AAB_READ_Routine |

**3765**

Table 185. 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_LDSO_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 186. 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   |

*Table 186. Trace Record 3770 - Isolated Log Transport Control Routine Entry (continued)*

|    |      |    |                |
|----|------|----|----------------|
| 12 | Char | 16 | AWE parameters |
|----|------|----|----------------|

**3771***Table 187. 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 188. 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 189. 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 190. 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 191. 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)

*Table 191. Trace Record 3775 - Isolated Log Transport Query Response (continued)*

| Trace Subcode LRICM Query |       |        |                   |
|---------------------------|-------|--------|-------------------|
| Offset                    | Type  | Length | Description       |
| 4                         | Fixed | 4      | PRILOG token      |
| 8                         | Fixed | 4      | High PRILOG token |
| 12                        | Fixed | 4      | DBRC rc           |

### 3776

*Table 192. 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 193. 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 194. 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 195. 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

Table 195. Trace Record 3779 - Isolated Log Transport DS Received (continued)

| Offset | Type  | Length | Description |
|--------|-------|--------|-------------|
| 4      | Fixed | 2      | Request ID  |
| 6      | Fixed | 2      | Reserved    |
| 8      | Fixed | 4      | SRA Address |
| 12     | Fixed | 4      | STB Address |

**377A**

Table 196. 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 197. 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      | Time stamp                  |

**3781**

Table 198. 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 198. Trace Record 3781 - Milestone Complete (continued)

|    |          |   |                             |
|----|----------|---|-----------------------------|
|    | ..1..... |   | Timer pop                   |
| 13 | Char     | 3 | Spares                      |
| 16 | Fixed    | 4 | LGB restart milestone index |
| 24 | Char     | 8 | Time stamp                  |

**3782**

Table 199. Trace Record 3782 - Unplan Takeover Process Phase 1 Entry

**Module:** DFSLRTRK0 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      | Time stamp                  |

**3783**

Table 200. Trace Record 3783 - Unplan Takeover Process Phase 2 Entry

**Module:** DFSLRTRK0 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      | Time stamp                  |

**3784**

Table 201. 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 202. 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           | 16     | lgb_takeover_flags                                 |
|        | 1.....        |        | On= plan takeover requested                        |
|        | .1.....       |        | On= plan takeover progress                         |
|        | ..1....       |        | On= unplan takeover requested                      |
|        | ...1...       |        | On= unplan takeover progress                       |
|        | ....1..       |        | Takeover reversed                                  |
|        | .....1.       |        | On= unplan takeover pending                        |
|        | .....1.       |        | On= 'takeover start' notify to dbrs has been done  |
|        | .....1        |        | On= truncation complete                            |
|        | ..... 1.....  |        | On= NOREVERSE was specified on /RTA UNPLAN command |
|        | ..... .1..... |        | On= unhardened buffers were on                     |

Table 202. Trace Record 3785 - Log Router Master ITASK Request Done (continued)

|    |                |   |                                                                |
|----|----------------|---|----------------------------------------------------------------|
|    | ..... ..1..... |   | On if this is the restart after a takeover processed for uptko |
|    | ..... ..11111  |   | Spares ( last 5 bits reserved)                                 |
| 8  | Fixed          | 4 | Current milestone index                                        |
| 12 | Char           | 8 | Time stamp                                                     |
| 16 | Fixed          | 4 | Current milestone index                                        |
| 24 | Char           | 8 | Time stamp                                                     |

**3786**

Table 203. 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      | Time stamp  |

**3787**

Table 204. 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 205. 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                                  |
| 12     | Bit            | 16     | lgb_flags                                           |
|        | 1.....         |        | On if we are terminating                            |
|        | .1.....        |        | Transport mgr has terminated                        |
|        | ..1.....       |        | On if we are identified to TM                       |
|        | ...1....       |        | On if /STO SERVGRP entered                          |
|        | ....1...       |        | On if /STA SERVGRP entered                          |
|        | .....1..       |        | On during log router restart                        |
|        | .....1.        |        | On if LR termination is waiting for OFR to shutdown |
|        | .....1         |        | On if initialization complete                       |
|        | ..... 1.....   |        | On if no restart mpb from log                       |
|        | ..... .1.....  |        | On if quiesce itasks for ptk                        |
|        | ..... ..1..... |        | On if this is the restart after a takeover          |
|        | ..... ..1....  |        | Position set from ds                                |
|        | ..... ..1...   |        | On if set from 51 pos                               |
|        | ..... ..1..    |        | On if xrc tracking requested                        |
|        | ..... ..11     |        | Spares (last 2 bits reserved)                       |
| 14     | Bit            | 16     | lgb_takeover_flags                                  |



Table 205. Trace Record 3788 - Create Active Stream Begin Takeover (continued)

|  |          |          |                                                                    |
|--|----------|----------|--------------------------------------------------------------------|
|  | 1.....   | .....    | 0n= plan takeover requested                                        |
|  | .1.....  | .....    | 0n= plan takeover progress                                         |
|  | ..1..... | .....    | 0n= unplan takeover requested                                      |
|  | ...1.... | .....    | 0n= unplan takeover progress                                       |
|  | ....1... | .....    | Takeover reversed                                                  |
|  | .....1.. | .....    | 0n= unplan takeovr pending                                         |
|  | .....1.  | .....    | 0n= 'takeover start' notify to<br>dbrs has been done               |
|  | .....1   | .....    | 0n= truncation complete                                            |
|  | .....    | 1.....   | 0n= NOREVERSE was specified on<br>/RTA UNPLAN command              |
|  | .....    | .1.....  | 0n= unhardened buffers were                                        |
|  | .....    | .1.....  | 0n if this is the restart after<br>ga takeover processed for uptko |
|  | .....    | ...11111 | Spares ( last 5 bits reserved)                                     |

## X'4930': Database Tracker FSE Error Log Record Format

The log record layout for X'4930' is shown in Table 206.

Table 206. X'4930' Log Record Layout

| 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) |



---

## Chapter 16. CQS Diagnosis

This section describes diagnostic information that helps you analyze problems in CQS.

### **In this section:**

- “Diagnosing a CQS Related Problem”
- “CQS Structure Rebuild Problems” on page 525
- “CQS Trace records” on page 526
- “CQS Log Records” on page 529
- “Printing CQS Log Records” on page 531

---

### **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.

For a CQS environment, related problems might include:

- IMS WAIT problems
- CQS WAIT/HANG problems
- CQS checkpoint problems
- CQS restart problems
- CQS structure rebuild problems

Implement normal operating procedures to preserve the following documentation near the time of error:

- Additional manual dump intervention
- z/OS Log Stream (for IMS shared queue related problems)
- Most recent SRDS (Structure Recovery Data Set) for each dumped structure

For a CQS WAIT/HANG problem, obtain dumps and syslogs of CQS address spaces in the sysplex such as:

- One dump and syslog from the master CQS
- One dump and syslog from the non-master CQS
- One dump and syslog from the error CQS
- One dump and syslog from the normal CQS

There is only one master CQS and, most likely, one error CQS in a sysplex. Thus, the maximum number of CQS dumps and syslogs to be taken is three. If the sysplex contains less than three CQS address spaces, then dumps and syslogs of all CQS address spaces are needed.

Before obtaining the syslog, issue the following z/OS DISPLAY commands to have the sysplex information written into the syslog:

```
D CF,CFNAME=cfname
D XCF,CF
D XCF,STRUCTURE,STRNAME=strname
```

For a CQS loop problem, obtain two dumps.

1. Obtain a z/OS SVC dump of the CQS and its associated IMS control region address space using the following command:

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

| In this command:

| *j1* is the CQS job name

| *j2* is the IMS control region job name

| 2. Save the IMS log data sets that are created during the error period.

| 3. Save the current z/OS log data sets that are created. The current z/OS log data sets for the CQS log stream can be copied using the IEBGENER utility. There are no archived z/OS 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 “Setting Up CQS, OM, RM, and SCI Tracing” on page 10 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.

| The following topics provide additional information:

- | • “CQS Additional Manual Intervention for Dump Creation”
- | • “CQS Structure Dump Contents”
- | • “CQS - z/OS Log Stream” on page 524
- | • “CQS Structure Recovery Data Set” on page 524
- | • “CQS Checkpoint Problems” on page 524

## | **CQS Additional Manual Intervention for Dump Creation**

| CQS environment additional dump considerations include:

- | • Structure dumps
- | • CQS regions and other CQS clients with their related CQS regions
- | • CQS regions and other CQS clients with their related CQS regions from other IMSplex members
- | • z/OS Logger

## | **CQS Structure Dump Contents**

| CQS structure dumps should include:

- | • The primary structures
- | • The overflow structures
- | • The associated lock entries

## | **CQS Structure Dump Example**

| Here is an example of the STRLIST for a dump:

```
| DUMP COMM=(MSGQ STRUCTURE DUMP)
| R nn,
| STRLIST=(STRNAME=imsmsgq01,LOCKE,(LISTNUM=ALL,ADJ=CAPTURE,EDATA=UNSER),
| STRNAME=imsmsgq01oflw,LOCKE,(LISTNUM=ALL,ADJ=CAPTURE,EDATA=UNSER)),END
```

| Where:

| *imsmsgq01*

| The main structure name.

| *imsmsgq01oflw*

| The overflow structure name.

| When an IMS structure dump is necessary, it is possible that the z/OS Logger function could be involved with the problem. The following example will dump the z/OS Logger address spaces, the logger structure, and the IMS CF structure.

```
| DUMP COMM=(CQS/LOGR STRUCTURE DUMP)
| R vv,STRLIST=(STRNAME=imsmsgq01,LOCKE,
|           (LISTNUM=ALL,ADJ=CAPTURE,EDATA=UNSER),CONT
| R ww,STRNAME=imsmsgq01oflw,LOCKE,
|           (LISTNUM=ALL,ADJ=CAPTURE,EDATA=UNSER),CONT
| R xx,STRNAME=mvslogqmsg01,LOCKE,ACC=NOLIM,
|           (LISTNUM=ALL,EDATA=UNSER,ADJ=CAPTURE)),CONT
| R yy,JOBNAME=(IXGLOGR),DSPNAME=('IXGLOGR'.SYSLOGR0),CONT
| R zz,SDATA=(COUPLE,ALLNUC,LPA,LSQA,PSA,RGN,SQA,TRT,CSA,GRSQ,XESDATA),END
```

| Where:

| *imsmsgq01*  
| The main structure name.

| *imsmsgq01oflw*  
| The overflow structure name.

| *mvslogqmsg01*

### | **CQS - IEADMCxx Example with Structures**

| Create three SYS1.PARMLIB members called IEADMCIA, IEADMCIB, IEADMCIC:

```
| JOBNAME=(j1,j2,j3,j4,j5),SDATA=(CSA,PSA,RGN,SQA,SUM,TRT,GRSQ),
| REMOTE=(SYSLIST=*('j1','j2','j3','j4','j5'),SDATA)
```

| Where:

| *j1* IMS Control Region Jobname.

| *j2* IMS DLI Region Jobname.

| *j3* DBRC Region Jobname.

| *j4* IRLM Region Jobname.

| *j5* IMS CQS Region.

```
| JOBNAME=(j6,j7,j8,j9,j10),SDATA=(CSA,PSA,RGN,SQA,SUM,TRT,XESDATA),
| REMOTE=(SYSLIST=*('j6','j7','j8','j9','j10'),SDATA)
```

| Where:

| *j6* APPC Region.

| *j7* APPC Scheduler.

| *j8* VTAM.

| *j9* Other CQS Client Region.

| *j10* Other CQS Region.

```
| JOBNAME=(IXGLOGR),DSPNAME=('IXGLOGR'.SYSLOGR0),
| SDATA=(COUPLE,ALLNUC,LPA,PSA,RGN,SQA,TRT,CSA,GRSQ,XESDATA),
| STRLIST=(STRNAME=imsmsgq01,LOCKE,(LISTNUM=ALL,ADJ=CAPTURE,EDATA=UNSER),
| STRNAME=imsmsgq01oflw,LOCKE,(LISTNUM=ALL,ADJ=CAPTURE,EDATA=UNSER),
| STRNAME=mvslogqmsg01,LOCKE,ACC=NOLIM,(LISTNUM=ALL,EDATA=UNSER,ADJ=CAPTURE))
```

| Where:

| *imsmsgq01*  
| The main structure name.

| *imsmsgq01oflw*  
| The overflow structure name.

| *mvslogqmsg01*  
|       The associated logger structure.

### | **CQS - IEADMCxx DUMP Activation**

| To request a dump from the IEADMCIA, IEADDMCIB and IEADMCIC parmlib members, enter the following z/OS command:

| DUMP TITLE=(DUMP OF IMSplex and Partners),PARMLIB=(IA,IB,IC)

| Three dump data sets are created on the z/OS image from which the command is entered. Two dump data sets are created on each image in the sysplex matching the REMOTE specifications for the JOBNAMEs.

| **Recommendation:** Provide the z/OS logger address space from the system experiencing problems to z/OS logger support.

### | **CQS - z/OS Log Stream**

| The merged z/OS log stream can be used to examine CQS log records. IEBGENER can be used along with the default log stream subsystem exit routine, IXGSEXIT, to copy the log records at time of failure for later analysis.

### | **CQS - z/OS Log Stream, JCL Example**

```
| //CQSCPYLG JOB USERID,USERID,MSGLEVEL=1,CLASS=K
| //*****
| /* This job copies a CQS log stream to a dataset (max 32K / record) *
| /*
| /* - Replace the DSN on the SYSUT1 card with your CQS logstream *
| /* name. *
| /*
| /* - Replace the DSN on the SYSUT2 card with your desired output *
| /* dataset name. You may also need to adjust the space *
| /* allocations, depending on the size of your logstream. *
| //*****
| //STEP1 EXEC PGM=IEBGENER,REGION=1024K
| //SYSPRINT DD SYSOUT=*
| //SYSUDUMP DD SYSOUT=*
| //SYSIN DD DUMMY
| //SYSUT1 DD DSN=SYSLOG.MSGQ01.LOG,
| // SUBSYS=(LOGR,IXGSEXIT),
| // DCB=(BLKSIZE=32760)
| //SYSUT2 DD DSN=CQS.LOG.COPY,
| // DISP=(NEW,KEEP,DELETE),
| // VOL=SER=USER05,
| // SPACE=(CYL,(2,10)),
| // UNIT=SYSDA
```

### | **CQS Structure Recovery Data Set**

| Save the most recent CQS SRDS (Structure Recovery Data Set) for each structure dumped. Use the IDCAMS REPRO command if the LRECL is acceptable (less than 32761).

### | **CQS Checkpoint Problems**

| There are two types of CQS checkpoints: system checkpoint and structure checkpoint. Most problems are of structure checkpoint type because it is a sysplex-wide operation with shared resources (SRDS data set, structures on the CF), and it needs cooperation through z/OS IXLUSYNC between all CQSs within the sysplex. Sometimes, another CQS process (initialization, termination, rebuild, overflow threshold, or overflow scan) can interfere with the checkpoint process and cause it to fail.

### | **CQS System Checkpoint Messages**

| 1. CQS0030I for a successful CQS system checkpoint.

2. CQS0035E for a failed CQS system checkpoint. If the CQS system checkpoint failed with CQS0035E, refer to “CQS Messages” in *IMS Version 9: Messages and Codes, Volume 1* for the details of the failure and the recommended system programmer action.

### CQS Structure Checkpoint Messages

The syslog of a successful CQS structure checkpoint will contain five CQS messages in the following order:

```
CQS0220I CQS cqsname START STRUCTURE CHECKPOINT FOR STRUCTURE strname
CQS0200I STRUCTURE strname QUIESCED FOR STRUCTURE CHECKPOINT
CQS0201I STRUCTURE strname RESUME AFTER STRUCTURE CHECKPOINT
CQS0030I SYSTEM CHECKPOINT COMPLETE, STRUCTURE strname LOGTOKEN logtoken
CQS0221I CQS COMPLETE STRUCTURE CHECKPOINT FOR STRUCTURE strname
```

If the CQS structure checkpoint failed with a CQS0222E message, refer to the “CQS Messages” section in *IMS Version 9: Messages and Codes, Volume 1* for the details of the failure and the recommended system programmer action.

If message CQS0222E is not displayed and not all five of the normal structure checkpoint messages appeared on the console, then CQS has most likely encountered a WAIT/HANG serious problem.

If CQS encountered a WAIT/HANG problem, refer to “Diagnosing CQS-Related Problems” on page 24 for guidelines about obtaining dumps and syslogs. Also, the CQS log records, the structure dump of the related structure, and the SRDS (Structure Recovery Data Sets) are helpful in diagnosing the problem. After collecting all the documents, you can stop the CQS in error and restart it to resolve the WAIT/HANG problem.

---

## CQS Structure Rebuild Problems

The most common structure rebuild problem is a rebuild failure. Some environmental situations can occur that cause rebuild to fail. Other types of rebuild problems are much more rare, such as rebuild hanging, rebuild not being initiated when required, work hanging after a successful rebuild, rebuild losing data objects, and rebuild duplicating data objects. Follow these general steps to address any rebuild failure you encounter:

- **Collect SYSLOGs**

Collect the syslog for each LPAR that is running a CQS that is sharing queues. Evaluate each syslog for the following information:

- How the rebuild was initiated (operator command, structure failure, CF failure, link failure).
- How the rebuild was stopped (operator command or CQS).
- Rebuild master (CQS0240I message).
- Rebuild type (COPY or RECOVERY in CQS0240I message).
- Structure quiesced or resumed messages:
  - CQS0200I STRUCTURE *strname* QUIESCED FOR *reason*
  - CQS0201I STRUCTURE *strname* RESUMED AFTER *reason*
- Structure status change messages (CQS0202I).
- Structure rebuild messages:
  - CQS0240I CQS *cqsname* STARTED STRUCTURE *copy/recovery* FOR STRUCTURE *strname*
  - CQS0241I CQS *cqsname* COMPLETED STRUCTURE *copy/recovery* FOR STRUCTURE *strname*
  - CQS0242E CQS FAILED STRUCTURE *copy/recovery/rebuild* FOR STRUCTURE *strname*
  - CQS0243E CQS *cqsname* UNABLE TO PARTICIPATE IN REBUILD FOR STRUCTURE *strname*
  - CQS0244E STRUCTURE RECOVERY REQUIRED AFTER RECOVERY FAILURE FOR STRUCTURE *strname*
  - CQS0245E STRUCTURE *strname* REBUILD ERROR

- **Consult the CQS Restart and Rebuild Error Reason Codes table**  
If rebuild failed with an error message such as the CQS0242E message, consult the topic titled “CQS Restart and Rebuild Reason Codes” in the *IMS Version 9: Messages and Codes, Volume 1* for details on the system programmer action to take.
- **Check rebuild status**  
Check the rebuild status by issuing the following command on every LPAR where a CQS participating in the rebuild resides:  
`D XCF,STRUCTURE,STRNAME=strname`  
If the output indicates that rebuild is waiting for a particular event, a CQS might not be responding to a rebuild event because it is hung or in a loop, which hangs the rebuild. Consider dumping the CQS address space and canceling the CQS that is not responding to the rebuild event, to see if that enables the rebuild to continue.
- **Analyze if structure still viable**  
If a structure copy initiated by an operator failed, no action needs to be taken to restore access to the structure. The structure is still viable and you still have access. Analyze why the structure copy failed, to determine whether you need to take action to prevent a subsequent rebuild failure.
- **Restore link, if applicable**  
If a structure rebuild was initiated because of a link failure and the structure rebuild failed, try to restore the link to restore access to the structure. The structure is still viable. Analyze why the structure rebuild failed, to determine whether you need to take action to prevent a subsequent rebuild failure.
- **Contact IBM**  
If you are unable to resolve the problem, do the following:
  - Copy the SYSLOG, including the `D XCF,STRUCTURE,STRNAME=strname` output from every LPAR.
  - Dump all the CQS address spaces, especially the rebuild master CQS address space. The CQS0240I message should indicate the rebuild master name.
  - Retain the CQS log records. The CQS log might contain important log records pertaining to data objects put on the structure, moved on the structure, or deleted from the structure. The CQS log might also contain important log records pertaining to rebuild, such as:
    - Rebuild begin log record (4301).
    - Rebuild end log record (4302).
    - Rebuild failed log record (4303).
    - Rebuild lost UOW list log record (4304).
    - Request log records (03xx, 07xx, 08xx, 0Bxx, 0Dxx).
  - Retain the IMS log records.
  - Take a structure dump if you suspect a rebuild hang. The structure dump might contain important information about structure locks.
  - Call the IBM Support Center for help.

---

## CQS Trace records

You can analyze CQS trace records in a formatted dump to help you determine what function encountered an error, and whether a problem is environmental or internal. Trace record eye-catchers in a formatted dump can provide a clue about what function resulted in an error. You might be able to take action to correct environmental problems right away. Internal IBM problems should be referred to IBM with appropriate documentation, such as system console logs and dumps.



| CQS trace records are written to one or more of the trace tables shown in Table 207.

| *Table 207. Trace Tables Containing CQS Trace Records*

| Table Name | Number of Tables             | Table Description                    |
|------------|------------------------------|--------------------------------------|
| ERR        | 1                            | Errors                               |
| CQS        | 1                            | CQS activity, including errors       |
| STR        | 1 per structure (EMHQ, MSGQ) | Structure activity, including errors |

| Each CQS trace record contains 32 bytes. The first byte is the trace code and the second byte is the trace subcode. Many trace records contain a structure ID that identifies which structure the trace record applies to: the MSGQ primary structure, the MSGQ overflow structure, the EMHQ primary structure, or the EMHQ overflow structure. Trace records that apply to a client request contain a client ID that identifies the client that issued the request. The last 8 bytes are the STCK time stamp of when the trace record was written. The mapping of the rest of the bytes in the trace record is unique to the trace code and subcode.

| CQS trace records are mapped by macros that use the naming convention CQSTRxxx, where xxx represents the function being traced. For example, CQSTRPUT maps trace records associated with the CQSPUT request. Trace record mapping is based upon the trace code and the trace subcode.

| Look up the CQS trace code in Table 208 to locate the CQS macro that maps the trace record. Table 208 lists the CQS trace codes, the macro that maps the trace code, and a description of the trace macro.

| *Table 208. CQS Trace Codes and Mapping Macros*

| Trace Code | Macro    | Description              |
|------------|----------|--------------------------|
| 3          | CQSTRCON | CQSCONN request          |
| 4          | CQSTRDSC | CQSDISC request          |
| 5          | CQSTRRSY | CQSRSYNC request         |
| 6          | CQSTRINF | CQSINFRM request         |
| 7          | CQSTRPUT | CQSPUT request           |
| 8          | CQSTRRD  | CQSREAD request          |
| 9          | CQSTRBRW | CQSBRWSE request         |
| 0A         | CQSTRUNL | CQSUNLCK request         |
| 0B         | CQSTRMOV | CQSMOVE request          |
| 0C         | CQSTRRCV | CQSRECVR request         |
| 0D         | CQSTRDEL | CQSDDEL request          |
| E          | CQSTRQRY | CQSQUERY request         |
| 0F         | CQSTRCHK | CQSCHKPT request         |
| 10         | CQSTRSHT | CQSSHUT request          |
| 11         | CQSTRUPD | CQSUPD request           |
| 30         | CQSTRICQ | CQS initialization       |
| 31         | CQSTRTCQ | CQS termination          |
| 32         | CQSTRYCH | System checkpoint        |
| 40         | CQSTRIST | Structure initialization |
| 41         | CQSTRSTS | Structure service        |
| 42         | CQSTRTCH | Structure checkpoint     |
| 43         | CQSTRRBL | Rebuild                  |

| *Table 208. CQS Trace Codes and Mapping Macros (continued)*

| <b>Trace Code</b> | <b>Macro</b> | <b>Description</b>          |
|-------------------|--------------|-----------------------------|
| 44                | CQSTROFL     | Overflow                    |
| 45                | CQSTRSTE     | Structure event             |
| 50                | CQSTRLOG     | Log services                |
| 51                | CQSTRTBL     | Table services              |
| 52                | CQSTRDYA     | Dynamic allocation services |
| 53                | CQSTRDSS     | Data set services           |
| 54                | CQSTRDSP     | Data space services         |
| 55                | CQSTRLRR     | Log record router           |
| 56                | CQSTRXCF     | XCF interface               |
| 57                | CQSTRCMD     | Command                     |
| 60                | CQSTRSTT     | Statistics                  |
| 70                | CQSTRINT     | CQS client interface        |

| Trace codes for CQS requests are defined in the CQSRQTYP macro. Trace codes for other CQS functions are defined in the CQSCODES macro. CQS trace records in a formatted dump might contain eye-catchers that provide a clue about what function encountered an error, such as “overflow”, “rbl”, “str chkpt”, and “duplex.”

| CQS request trace records sometimes contain a return code, reason code, and completion code from the request. CQS request return codes, reason codes, and completion codes are mapped by macros that use the naming convention CQSRRxxx, where xxx represents the function being traced. For example, CQSRRPUT maps return codes, reason codes, and completion codes associated with the CQSPUT request. Look up the macro that defines the return codes, reason codes, and completion codes for the CQS request in Table 209.

| *Table 209. CQS Mapping Macros and Request Trace Records*

| <b>Macro</b> | <b>CQS request macro for return codes, reason codes, completion codes</b> |
|--------------|---------------------------------------------------------------------------|
| CQSRRCON     | CQSCONN                                                                   |
| CQSRRDSC     | CQSDISC                                                                   |
| CQSRRRSY     | CQSRSYNC                                                                  |
| CQSRRINF     | CQSINFRM                                                                  |
| CQSRRPUT     | CQSPUT                                                                    |
| CQSRRD       | CQSREAD                                                                   |
| CQSRRBRW     | CQSBRWSE                                                                  |
| CQSRRUNL     | CQSUNLCK                                                                  |
| CQSRRMOV     | CQSMOVE                                                                   |
| CQSRRRCV     | CQSRECVR                                                                  |
| CQSRRDEL     | CQSDEL                                                                    |
| CQSRRQRY     | CQSQUERY                                                                  |
| CQSRRCHK     | CQSCHKPT                                                                  |
| CQSRRSHT     | CQSSHUT                                                                   |
| CQSRRUPD     | CQSUPD                                                                    |

| CQS trace records in formatted dumps contain eye-catchers that identify the trace code and the trace subcode.

| Here is an example of a CQS trace record with eye-catchers:

```
| INFRM: INF DONE FOR Q      06090101 05E3F3F2 F7F0D3C1 40404040  
|                          40404040 05541160 AF975E81 59426906
```

| The trace code is in the first byte (X'06'), which the CQSRQTYP macro documents as the CQSINFRM request. The eye-catcher for this is INFRM. The CQSTRINF macro maps the trace records for trace code X'06'.

| The trace subcode is in the second byte (X'09'), which the CQSTRINF macro documents as "inform done for queue." The eye-catcher for this is INF DONE FOR Q.

| The CQSTRINF macro documents byte 3 for trace code X'06' as containing the structure ID (X'01'). Structure ID X'01' indicates the primary MSGQ structure.

| The CQSTRINF macro documents byte 4 for trace subcode X'06' as containing the client ID (X'01'). Client ID X'01' represents the client that issued the CQSINFRM request. The CQSTRINF macro documents words 2, 3, 4, and 5 for trace subcode X'06' as containing the name of the queue for which the inform was done. This queue name is for queue type 05 (the IMS transaction queue). The queue name is T3270LA (X'E3F3F2F7F0D3C1').

| The CQSTRINF macro documents word 6 for trace subcode X'06' as the ECB of the task that wrote this trace record (X'05541160').

| The CQSTRINF macro documents words 7 and 8 as the STCK time of when the trace record was written.

---

## CQS Log Records

CQS writes records to the z/OS log stream that contains all CQS log records from all CQs 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 210 on page 530 lists:

- The log record type and subtype
- The macro that maps the record
- The events that cause the record to be written

Table 210. 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. |
| 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.        |

Table 210. CQS Log Records (continued)

| Type  | Sub type | Mapping Macro | Conditions for Writing the Log Record                                                                                                                              |
|-------|----------|---------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|       | 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.                                                                                                                                           |
| 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'0C'    |               | Internal BPE service statistics were written at the end of system checkpoint.                                                                                      |

## Printing CQS Log Records

To print the CQS log records from the z/OS 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 a z/OS system log. This JCL causes the z/OS 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.SDFSRESL
//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.SDFSRESL, 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')).

### Limiting Log Data to a Specified Time Range

You can limit the log records you print to those in a particular interval of time by using the FROM and TO parameters on the SUBSYS statement. For example, the following DD card:

```
//SYSUT1 DD DSN=SYSLOG.MSGQ01.LOG,
//          SUBSYS=(LOGR,IXGSEXIT,
//          'FROM=(2001/042,11:00:00),TO=(2001/042,12:00:00)'),
//          DCB=(BLKSIZE=32760)
```

would pass log records only from 11:00 to 12:00 on day 42 of the year 2001 to the DFSERA10 program. Dates and times specified in this manner are in GMT, and the seconds field of the time values is optional. If you want to use local dates and times, add the LOCAL keyword to the statement:

```
//SYSUT1 DD DSN=SYSLOG.MSGQ01.LOG,
//          SUBSYS=(LOGR,IXGSEXIT,
//          'FROM=(2001/042,11:00:00),TO=(2001/042,12:00:00),LOCAL'),
//          DCB=(BLKSIZE=32760)
```

---

## Copying CQS Log Records for Diagnostics

IBM service will sometimes require a copy of a range of CQS log records for problem determination. You can use the IEBGENER utility program to copy some or all of the CQS log for a structure to a BSAM data set for sending to IBM service. The copy made by IEBGENER is a binary image of the log records. The following JCL is a job that will copy CQS log records between 15:10 and 15:30 local time on day 89 of 2001 to a data set named CQS.LOG.COPY:

```
//CQSCPYLG JOB MSGLEVEL=1,CLASS=K
//*****
//* THIS JOB COPIES A CQS LOG STREAM TO A DATASET (MAX 32K / RECORD) *
//*****
//STEP1 EXEC PGM=IEBGENER
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSIN DD DUMMY
//SYSUT1 DD DSN=SYSLOG.MSGQ01.LOG,
//          SUBSYS=(LOGR,IXGSEXIT,
//          'FROM=(2001/089,15:10),TO=(2001/089,15:30),LOCAL'),
//          DCB=(BLKSIZE=32760)
//SYSUT2 DD DSN=CQS.LOG.COPY,
```

```
//          DISP=(NEW,KEEP,DELETE),  
//          VOL=SER=EDSDMP,  
//          SPACE=(CYL,(10,10)),  
//          UNIT=SYSDA
```

If you copy CQS log records using IEBGENER, be aware of the following:

- The copied records cannot be used by CQS in any way (such as restart or recovery). They are for diagnostic purposes only.
- CQS log records that are greater than 32K bytes in length are truncated. The SUBSYS exit supports a maximum of a 32K record size.

**Related Reading:** For a complete description of the IMS File Select and Formatting Print utility (DFSERA10), see *IMS Version 9: Utilities Reference: System*. For a complete description of the z/OS logger subsystem exit (IXGSEXIT) usage and parameters, see the *MVS Diagnosis: Tools and Service Aids*.





## Chapter 17. CSL Diagnosis

This section describes diagnostic information to help you analyze problems in CSL.

### **In this section:**

- “CSL Trace Records”
- “RM Trace Record Example” on page 537

### **CSL Trace Records**

You can analyze CSL address space trace records (for example OM, RM, or SCI) in a formatted dump to help you determine whether the problem is environmental or internal. Trace record eye-catchers in a formatted dump can provide a clue about what function resulted in an error. You might be able to take action to correct environmental problems right away. Internal IBM problems should be referred to IBM with appropriate documentation, such as system console logs and dumps.

OM trace records are written to one or more of the trace tables shown in Table 211.

*Table 211. Trace Tables for OM Trace Records*

| Table Name | Number of Tables | Table Description                  |
|------------|------------------|------------------------------------|
| CSL        | 1                | CSL activity, including errors     |
| ERR        | 1                | Errors                             |
| OM         | 1                | OM activity                        |
| PLEX       | 1 per IMSplex    | IMSplex activity, including errors |

RM trace records are written to one or more of the trace tables shown in Table 212.

*Table 212. Trace Tables for RM Trace Records*

| Table Name | Number of Tables | Table Description                  |
|------------|------------------|------------------------------------|
| CSL        | 1                | CSL activity, including errors     |
| ERR        | 1                | Errors                             |
| PLEX       | 1 per IMSplex    | IMSplex activity, including errors |
| RM         | 1                | RM activity, including errors      |

SCI trace records are written to one or more of the trace tables shown in Table 213.

*Table 213. Trace Tables for SCI Trace Records*

| Table Name | Number of Tables | Table Description                  |
|------------|------------------|------------------------------------|
| CSL        | 1                | CSL activity, including errors     |
| ERPL       | 1                | Parmlist errors                    |
| ERR        | 1                | Errors                             |
| INTF       | 1                | SCI interface activity             |
| INTP       | 1                | Interface parmlist                 |
| PLEX       | 1 per IMSplex    | IMSplex activity, including errors |
| SCI        | 1                | SCI activity, including errors     |

Each CSL trace record contains 32 bytes (the ERPL and INTP tables in the SCI address space contain records that are 256 bytes). The first byte is the trace code, which indicates the function that wrote the trace record. Examples of trace code functions include address space initialization, address space termination, the CSLOMCMC request, the CSLRMUPD request, and the CSLSCRQS request. The second byte is the trace subcode, which indicates the category of the trace record. Examples of trace subcode categories include begin request, end request, CQS error, and SCI error. Most trace records include a 2-byte module identifier of the module that wrote the trace record. The last 8 bytes are the STCK time stamp of when the trace record was written. Trace record mapping of the rest of the fields is unique to the trace subcode.

CSL address space trace codes and other common codes used in trace records are mapped by a macro following the naming convention of CSLxCODE macro, where x represents the CSL address space as shown in Table 214.

Table 214. CSL Address Space Trace Code Mapping Macros

| Codes Macro Name | Description                                     |
|------------------|-------------------------------------------------|
| CSLOCODE         | OM codes                                        |
| CSLRCODE         | RM codes                                        |
| CSLSCODE         | SCI codes                                       |
| CSLZCODE         | CSL codes common to multiple CSL address spaces |

CSL address space trace records are mapped by a macro following the naming convention of CSLxTRC macro, where x represents the CSL address space as shown in Table 215.

Table 215. CSL Address Space Trace Record Mapping Macros

| Trace Macro Name | Description                                             |
|------------------|---------------------------------------------------------|
| CSLOTRC          | OM trace records                                        |
| CSLRTRC          | RM trace records                                        |
| CSLSTRC          | SCI trace records                                       |
| CSLZTRC          | CSL trace records common to multiple CSL address spaces |

Trace record mapping is based upon the trace subcode, which identifies the category of trace record. One particular trace subcode can apply to many trace codes. Each trace record mapping also includes a pictorial representation in a comment block. Use the trace subcode to locate the trace record mapping in the CSLxTRC macro. Some trace codes are unique to a particular CSL address space, others are common to more than one CSL address space.

The CSLxCODE macro includes 2-byte module identifier codes that are used in trace records and messages when it is necessary to identify a CSL module. The module identifier represents the module that wrote the trace record. Module identifier codes are defined as follows:

- CSL address spaces**  
X'0001'-X'6FFF'
- CSLZ modules**  
X'7000'-X'77FF'
- BPE modules**  
X'7800'-X'7FFF'
- Not used**  
X'8000'

**Reserved for BPE tracing**

X'8001'-X'FFFF'

CSL request trace records sometimes contain a return code, reason code, and completion code from the request. CSL request return codes, reason codes, and completion codes are mapped by macros following the naming convention CSLxRR, where x represents the CSL address space as shown in Table 216.

*Table 216. CSL Request Return, Reason, and Completion Codes Mapping Macros*

| Macro  | Description                                          |
|--------|------------------------------------------------------|
| CSLORR | OM return codes, reason codes, and completion codes  |
| CSLRRR | RM return codes, reason codes, and completion codes  |
| CSLSRR | SCI return codes, reason codes, and completion codes |

CSL trace records in formatted dumps contain eye-catchers identifying the trace code, the trace subcode, and the module that wrote the trace record. The EPL and INTF trace tables in SCI contain parm lists, tables, and other data areas going across the SCI interface. These data areas are formatted in the SCI dump.

**RM Trace Record Example**

Here is an example of an RM trace record with eye-catchers:

```
CEVTX:*CQS SERVICE ERR RCQE 60110000 03000042 0000000C 00000304 00000008 0BC60C20 B6B1AF09 07C68F08
```

The trace code is in the first byte (X'60'), which the CSLRCODE macro documents as CQS Event Exit. The eye-catcher for this is CEVTX. The trace subcode is in the second byte (X'11'), which the CSLRTRC macro documents as a miscellaneous CQS service error. The eye-catcher for this is \*CQS SERVICE ERR. The asterisk at the beginning of the eye-catcher indicates an error.

The CSLRTRC macro documents byte 5 for trace subcode X'11' as containing the service request X'03', which is the CQSCONN request.

The CSLRTRC macro documents byte 8 for trace subcode X'11' as containing the module ID X'0042', which the CSLRCODE macro defines as module CSLRCQE0. The module name is included in the eye-catcher on RCQE. The CSLRTRC macro documents word 3 for trace subcode X'11' as containing the CQSCONN return code X'0000000C'. The CQSRRCON macro defines the CQSCONN return codes, reason codes, and completion codes. The CQSRRCON macro defines return code X'0000000C' as a list error.

The CSLRTRC macro documents word 4 for trace subcode X'11' as containing the CQSCONN reason code X'00000304'. The CQSRRCON macro defines reason code X'00000304' as no requests successful.

The CSLRTRC macro documents word 5 for trace subcode X'11' as containing the CQSCONN completion code X'00000008'. CQSRRCON macro defines completion code X'00000008' as no resource structure is defined. RM was unable to connect to the resource structure because it is not defined. This is probably an environmental problem where the resource structure was not correctly defined to CQS.

The CSLRTRC macro documents word 6 for trace subcode X'11' as containing the ECB address (X'0BC60C20').



---

## **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 the IMS keyword dictionary shown in Table 217 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 freeform and SDB formats are provided throughout the procedures in Chapter 4, “Selecting the Keywords,” on page 31.

**Related Information:** For more information about SDB formats, see *Software Service General Information Manual*.

Table 217. IMS Keyword Dictionary

| Category/Keyword Examples                                                                   | RETAIN Formats Keyword                         | SDB                                                      |
|---------------------------------------------------------------------------------------------|------------------------------------------------|----------------------------------------------------------|
| <b>Abends:</b><br>System 0C4<br>User 0845                                                   | ABEND0C4<br>ABENDU0845                         | AB/S00C4<br>AB/U0845                                     |
| <b>Access Methods:</b><br>OSAM<br>VSAM                                                      | OSAM<br>VSAM                                   | RIDS/OSAM<br>RIDS/VSAM                                   |
| <b>Automatic Operator Interface</b>                                                         | AOI                                            | RIDS/AOI                                                 |
| <b>APARS</b>                                                                                | PL12345                                        | PTFS/PL12345                                             |
| <b>Checkpoint Processing:</b><br>Checkpoint<br>Extended Checkpoint                          | CHKPT<br>XCHKPT                                | PCSS/CHKPT<br>PCSS/XCHKPT                                |
| <b>CICS Interface</b>                                                                       | CICSDLI                                        | PCSS/CICSDLI                                             |
| <b>IMS Commands:</b> <sup>1</sup><br>/ASSIGN<br>/CHECKPOINT<br>/ERESTART<br>/TRACE<br>/STOP | CMDASS<br>CMDCHE<br>CMDERE<br>CMDTRA<br>CMDSTO | PCSS/ASS<br>PCSS/CHE<br>PCSS/ERE<br>PCSS/TRA<br>PCSS/STO |
| <b>DBRC Commands:</b> <sup>2</sup><br>INIT.RECON<br>CHANGE.PRILOG                           | INITRECON<br>CHANGEPRILOG                      | PCSS/INITRECON<br>PCSS/CHANGEPRIL                        |
| <b>Condition Code</b>                                                                       | CC08 (HEX)                                     | PRCS/00000008                                            |
| <b>Control Blocks:</b><br>Data Control Block<br>Database Descriptor                         | DCB<br>DBD                                     | FLDS/DCB<br>FLDS/DBD                                     |
| <b>Database Organization</b>                                                                | HDAM                                           | PCSS/HDAM                                                |
| <b>Database Pre-Open</b>                                                                    | PRE-OPEN                                       | RIDS/PREOPEN                                             |
| <b>Data Sharing Environment</b>                                                             | DATA SHARING                                   | RIDS/DATASHARE                                           |

5. This is a sample of IMS keywords and is not intended to be a complete list.

Table 217. IMS Keyword Dictionary (continued)

| Category/Keyword Examples                                                                         | RETAIN Formats Keyword                     | SDB                                                                 |
|---------------------------------------------------------------------------------------------------|--------------------------------------------|---------------------------------------------------------------------|
| <b>Devices:</b><br>3270<br>LU TYPE1                                                               | D/T3270<br>SLU1                            | DEVS/3270<br>DEVS/SLU1                                              |
| <b>DL/I Address Space</b>                                                                         | DLISAS                                     | PCSS/DLISAS                                                         |
| <b>DSECTS</b>                                                                                     | IDSPWRK                                    | FLDS/IDSPWRK                                                        |
| <b>Emergency Restart Processing</b>                                                               | ERE                                        | RIDS/ERE                                                            |
| <b>Error Codes (DBRC)</b>                                                                         | EC0182062                                  | PRCS/00182062                                                       |
| <b>Extended Restart</b>                                                                           | XRST                                       | PCSS/XRST                                                           |
| <b>Fast Path:</b><br>Fast Path Area<br>Second CI<br>Main Storage Database<br>Sequential Dependent | FASTPATH<br>FPAREA<br>DMAC<br>MSDB<br>SDEP | RIDS/FASTPATH<br>PCSS/FPAREA<br>FLDS/DMAC<br>PCSS/MSDB<br>PCSS/SDEP |
| <b>Feedback Code</b>                                                                              | FDBK0C (HEX)                               | PRCS/0000000C                                                       |
| <b>Fields:</b><br>PSTUSID                                                                         | PSTUSID                                    | FLDS/PSTUSID                                                        |
| <b>Function Sub-Function</b>                                                                      | SYS CHKRT                                  | RIDS/SYS<br>RIDS/CHKRT                                              |
| <b>Function Codes</b>                                                                             | FC0291                                     | OPCS/0291                                                           |
| <b>System Definition:</b><br>ACB<br>NUCLEUS                                                       | ACBGEN<br>NUC                              | PCSS/ACB<br>PCSS/NUC                                                |
| <b>IRLM</b>                                                                                       | IRLM                                       | RIDS/IRLM                                                           |
| <b>Labels:</b><br>LOOPNEXT<br>FREEMAIN                                                            | LOOPNEXT<br>FREEMAIN                       | RIDS/LOOPNEXT<br>RIDS/FREEMAIN                                      |
| <b>Log Records:</b><br>TYPE 18<br>TYPE 67FF                                                       | TYPE18<br>TYPE67FF                         | PCSS/TYPE18<br>PCSS/TYPE67FF                                        |
| <b>Macros:</b><br>RWOS<br>TERMINAL                                                                | RWOS<br>TERMINAL                           | PCSS/RWOS<br>PCSS/TERMINAL                                          |
| <b>Master Terminal Operator</b>                                                                   | MTO                                        | PCSS/MTO                                                            |
| <b>Messages:</b><br>DFS045I<br>IEC030I                                                            | MSGDFS045I<br>MSGIEC030I                   | MS/DFS045I<br>MS/IEC030I                                            |
| <b>Modules:</b><br>DFSPCC20                                                                       | DFSPCC20                                   | RIDS/DFSPCC20                                                       |
| <b>Online Change</b>                                                                              | OLCHG                                      | PCSS/OLCHG                                                          |
| <b>Online Data Set</b>                                                                            | OLDS                                       | PCSS/OLDS                                                           |
| <b>Online Image Copy</b>                                                                          | OLIC                                       | RIDS/OLIC                                                           |
| <b>Parameters:</b><br>ERROPT=ACCEPT                                                               | ERROPT=ACCEPT                              | PCSS/ERROPT<br>PCSS/ACCEPT                                          |



Table 217. IMS Keyword Dictionary (continued)

| Category/Keyword Examples                                                                       | RETAIN Formats Keyword               | SDB                                            |
|-------------------------------------------------------------------------------------------------|--------------------------------------|------------------------------------------------|
| <b>Processing Options:</b><br>PROCOPT=GO                                                        | PROCOPT=GO                           | PCSS/PROCOPT<br>PCSS/GO                        |
| <b>Publication Numbers:</b><br>SY26-3991-2                                                      | SY26399102                           | PUBS/SY26399102                                |
| <b>Reason Codes</b>                                                                             | RSN08 (HEX)                          | PRCS/00000008                                  |
| <b>Registers:</b><br>General purpose registers<br>Control registers<br>Floating point registers | REG13 (DECIMAL)<br>CREG10<br>FPREG01 | REGS/GR13<br>REGS/CR10<br>REGS/FP01            |
| <b>Restart Processing</b>                                                                       | RSTRT                                | RIDS/RSTRT                                     |
| <b>Release Levels:</b><br>IMS Version 9 Database Manager<br>IMS Version 9 Transaction Manager   | AR901<br>AR902                       | LVLS/901<br>LVLS/902                           |
| <b>Return Codes:</b><br>Return code 12 (X'0C')                                                  | RC0C                                 | PRCS/0000000C                                  |
| <b>RSR Environment:</b><br>RSR                                                                  | IMSRSR                               | RIDS/IMSRSR                                    |
| <b>Sense Codes:</b><br>Sense 080B                                                               | SNS080B                              | PRCS/0000080B                                  |
| <b>Status Codes:</b><br>Status code GE<br>Status blank BLANK                                    | STATUSGE<br>STATUS4040               | PRCS/000000GE<br>PRCS/00004040                 |
| <b>Subcode</b>                                                                                  | SUBCODE101                           | PRCS/00000101                                  |
| <b>SVC Numbers</b>                                                                              | SVC255 (DECIMAL)                     | OPCS/SVCFE                                     |
| <b>Trace Entry Function Code</b>                                                                | TRACEE6 (DL/I)<br>TRACE03 (DISP)     | PCSS/TRACEE6<br>PCSS/TRACE03                   |
| <b>XRF Environments:</b><br>XRF<br>Takeover<br>Alternate                                        | IMSXRF<br>TAKEOVER<br>ALTERNATE      | RIDS/IMSXRF<br>PCSS/TAKEOVER<br>PCSS/ALTERNATE |

**Notes:**

1. IMS commands begin with the special character “/”, which is not searchable in RETAIN. Therefore, the convention is 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. 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.

| <b>Keyword</b> | <b>Environment</b>                          | <b>Keyword</b> | <b>Environment</b>       |
|----------------|---------------------------------------------|----------------|--------------------------|
| D/CICS         | CICS                                        | D/TRKREC       | Track Recovery           |
| D/CONVPROC     | Conversational Processing                   | D/TWX          | Teletype                 |
| D/FP           | Fast Path                                   | D/UCF          | Utility Control Facility |
| D/GSAM         | GSAM                                        | D/VSAM         | VSAM                     |
| D/HDAM         | HDAM                                        | D/VTAM         | VTAM                     |
| D/HIDAM        | HIDAM                                       | D/1050         | 1050 Device Type         |
| D/HISAM        | HISAM                                       | D/2260         | 2260 Device Type         |
| D/HSAM         | HSAM                                        | D/2740         | 2740 Device Type         |
| D/IRLM         | MS/VS Resource Lock Manager                 | D/2741         | 2741 Device Type         |
| D/LU6          | VTAM LU 6<br>(Intersystem Communication)    | D/2770         | 2770 Device Type         |
| D/MFS          | Message Format Services                     | D/2780         | 2780 Device Type         |
| D/MSC          | Multiple System Coupling                    | D/2980         | 2980 Device Type         |
| D/MVS          | z/OS                                        | D/3270         | 3270 Large Screen        |
| D/None         | No dependencies                             | D/3270L        | 3270 Local               |
| D/NTO          | Network Terminal Option                     | D/3270R        | 3270 Remote              |
| D/OSAM         | OSAM                                        | D/3274         | 3274 Device Type         |
| D/SB           | Sequential Buffering                        | D/3275         | 3275 Device Type         |
| D/SECINDX      | Secondary Index                             | D/3276         | 3276 Device Type         |
| D/SHISAM       | Simple HISAM                                | D/3278         | 3278 Device Type         |
| D/SLU1         | VTAM Type SLU 1                             | D/3279         | 3279 Device Type         |
| D/SLU2         | VTAM Type SLU 2                             | D/3284         | 3284 Device Type         |
| D/SLU4         | VTAM Type SLU 4                             | D/3286         | 3286 Device Type         |
| D/SLU P        | VTAM Type SLU P                             | D/3287         | 3287 Device Type         |
| D/SYSGEN       | PTFs that should be applied prior to SYSGEN | D/3350         | 3350 Device Type         |
| D/SYS3         | System/3                                    | D/3375         | 3375 Device Type         |
| D/SYS7         | System/7                                    | D/3380         | 3380 Device Type         |
|                |                                             | D/3600         | 3600 Device Types        |
|                |                                             | D/3790         | 3790 Device Types        |



## Appendix C. AIBREASN Codes for Queue Control Facility (QCF)/Message Requeuer (MRQ) Errors

This topic 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.

- In this section:
- “AIBREASN Codes Set by DFSQMRQ0”
  - “AIB Return Codes Set by DFSQMRQ0” on page 553

For more information about diagnosing problems with the Message Requeuer or how the Message Requeuer (MRQ) program product communicates with certain functions in the IMS Transaction Manager and System Services, see “Diagnosing Problems in the Queue Control Facility/Message Requeuer” on page 341.

### AIBREASN Codes Set by DFSQMRQ0

Table 218 lists the AIBREASN Codes Set by DFSQMRQ0.

Table 218. 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'001C' | ENTRY   | INVALID CMD/GCMD/ISRT CALL         |
| X'0020' | ENTRY   | INVALID MRQ FUNCTION               |
| X'0024' | ENTRY   | INVALID DFSQMR10 FUNCTION          |
| X'0028' | ENTRY   | INVALID SPANNED COMMAND            |
| X'002C' | ENTRY   | INVALID BUILD MRQ PREFIX CALL      |
| X'0030' | ENTRY   | ERROR DURING INIT QC FUN CALL      |
| 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    |

Table 218. AIBREASN Codes Set by DFSQMRQ0 (continued)

| Code    | Routine | Error Message                      |
|---------|---------|------------------------------------|
| 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    |
| 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 DFSPPOOL 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   |

Table 218. AIBREASN Codes Set by DFSQMRQ0 (continued)

| Code    | Routine      | Error Message                         |
|---------|--------------|---------------------------------------|
| 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'10F4' | ISRT         | ERROR CREATING DYNAMIC LNB            |
| X'10F8' | INIT         | CAN'T FIND SYS PREFIX SEG MSGSIPEX    |
| X'10FC' | INIT         | ERROR LOADING MODULE DFSTSPC0         |
| X'1100' | ISRT         | ISRT - /MSV CMD MESSAGE CANCELED      |
| X'1104' | INIT         | QMRQWORK SHOULD NOT OCCUR ERROR       |
| X'1108' | MSGPROC      | CAN'T FIND TMR PREFIX MSGMSC          |
| X'110C' | MSGPROC      | DFSSQQRV INVALID QUEUE TYPE           |
| X'1110' | INIT         | INVALID INCLUDE/EXCLUDE ENTRY         |
| X'1114' | INIT         | INVALID MRQWORK INIT CALL             |
| X'1118' | ISRT         | QBUFF DIDN'T FIT IN DEP RGN COMM AREA |
| X'111C' | ISRT         | DFSRAC6 ERROR GETTING UTOKEN FOR APPC |
| X'1120' | ISRT         | QUEUE BUFFER FLAG ERROR               |
| X'1124' | ISRT         | SEGMENT FLAG ERROR                    |
| X'1128' | ISRT         | INVALID QUEUE BUFFER DETECTED         |
| X'112C' | INIT/CLEANUP | ISWITCH FAILURE                       |
| X'1130' | INSERT       | ISRT QBUFF THRESHOLD EXCEEDED         |
| X'2000' | PURGE        | PURGE PCB NOT MODIFIABLE              |
| X'2004' | PURGE        | PURGE PCB DEST INVALID                |
| 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 SYS PREFIX MSGSSEGM        |
| X'4008' | CPYPRFX      | CAN'T FIND TMR PREFIX MSGMSC          |

Table 218. AIBREASN Codes Set by DFSQMRQ0 (continued)

| Code    | Routine  | Error Message                                                         |
|---------|----------|-----------------------------------------------------------------------|
| X'400C' | CPYPRFX  | CAN'T FIND SYS EXT PREFIX MSDMSE                                      |
| X'4010' | CPYPRFX  | CAN'T FIND THE MSC PREFIX MSGMSC                                      |
| 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'6010' | MRQ/IMS  | QBUF COUNT NOT EXPECTED NUMBER                                        |
| X'6014' | MSGPROC  | MSG WAS CANCELED BY IMS                                               |
| X'7004' | XLATPFX  | CAN'T FIND SYS EXT SEGMENT MSGSISEX                                   |
| X'7008' | XLATPFX  | CAN'T FIND PFX SEG MSGSITMR                                           |
| X'700C' | XLATPFX  | CAN'T FIND PFX SEG MSGMSC                                             |
| X'7010' | XLATPFX  | CAN'T FIND PFX SEG MSGMSCE                                            |
| X'7014' | XLATPFX  | ERROR CONVERTING MESSAGE TIME                                         |
| X'7018' | XLATPFX  | CAN'T FIND PFX MSGEPHDR                                               |
| X'8004' | QMR30    | BROWSE - SYSTEM NOT SHARED QUEUES                                     |
| X'8008' | QMR30    | INVALID FUNCTION PASSED TO BROWSE                                     |
| X'800C' | QMR30    | BROWSE RECEIVED ERROR CODE FROM SELECT                                |
| X'8010' | QMR30    | BROWSE COMMAND ERROR                                                  |
| X'8014' | QMR30    | BROWSE COMMAND ERROR                                                  |
| X'8018' | QMR30    | BROWSE COMMAND ERROR                                                  |
| X'801C' | QMR30    | BROWSE COMMAND ERROR                                                  |
| X'8020' | QMR30    | BROWSE COMMAND ERROR                                                  |
| X'8024' | QMR30    | BROWSE COMMAND ERROR                                                  |
| X'8028' | QMR30    | BROWSE COMMAND ERROR                                                  |
| X'802C' | QMR30    | BROWSE COMMAND ERROR                                                  |
| X'8030' | QMR30    | BROWSE COMMAND ERROR                                                  |
| X'8034' | QMR30    | BROWSE COMMAND ERROR                                                  |
| X'8038' | QMR30    | BROWSE COMMAND ERROR                                                  |
| X'803C' | QMR30    | BROWSE COMMAND ERROR                                                  |
| X'8040' | QMR30    | BROWSE INVALID DESTINATION                                            |
| X'8044' | QMR30    | BROWSE LOCAL QUEUES CONTROL BLOCK ERROR                               |
| X'8048' | QMR30    | BROWSE LOCAL QUEUES DESTINATION TYPE ERROR ON A MULTI-RECORD MESSAGE  |
| X'804C' | QMR30    | BROWSE LOCAL QUEUES CONTINUATION TYPE ERROR ON A MULTI-RECORD MESSAGE |
| X'8050' | QMR30    | QSN BLOCK ADDRESS IS ZERO                                             |
| X'8054' | QMR30    | BROWSE AREA PARM NOT SET                                              |
| X'9004' | QMR60    | QUERY - SYSTEM NOT SHARED QUEUES                                      |
| X'9008' | QMR60    | INVALID FUNCTION PASSED TO QUERY                                      |
| X'900C' | QMR60    | QUERY RECEIVED ERROR CODE FROM SELECT                                 |



Table 218. AIBREASN Codes Set by DFSQMRQ0 (continued)

| Code    | Routine | Error Message                                                                                                          |
|---------|---------|------------------------------------------------------------------------------------------------------------------------|
| X'9010' | QMR60   | QUERY - CMD QUEUE TYPE INVALID                                                                                         |
| X'9014' | QMR60   | QUERY - NO MESSAGE RETURNED ON INTERNAL CALL TO BROWSE                                                                 |
| X'9018' | QMR60   | QUERY - RETURN CODE ERROR ON INTERNAL CALL TO BROWSE                                                                   |
| X'901C' | QMR60   | QUERY - ERROR LOCATING APPC/OTMA PFX                                                                                   |
| X'9020' | QMR60   | QUERY - ERROR LOCATING TMR PREFIX                                                                                      |
| X'9024' | QMR60   | QUERY - SHOULD NOT OCCUR ERROR                                                                                         |
| X'9028' | QMR60   | ERROR FREEING BUFFER DURING CLEANUP                                                                                    |
| X'902C' | QMR60   | QUERY - QUERY CALLED BROWSE WITH A GET COMMAND                                                                         |
| X'9030' | QMR30   | BROWSE - QUERY CALLED BROWSE WITH AN DESTINATION OF ZERO                                                               |
| X'9034' | QMR30   | BROWSE COMMAND ERROR                                                                                                   |
| X'9038' | QMR60   | QUERY - QUERY CALLED BROWSE WITH AN INVALID QNAME                                                                      |
| X'903C' | QMR60   | QUERY - QUERY CALLED BROWSE WITH AN INVALID QUEUE SPACE NOTIFICATION BLOCK                                             |
| X'A004' | QMR50   | UNLOAD - SELECT QUEUE NAME ERROR                                                                                       |
| X'A008' | QMR50   | UNLOAD - Reserved                                                                                                      |
| X'A00C' | QMR50   | UNLOAD - GU CALL ERROR                                                                                                 |
| X'A010' | QMR50   | UNLOAD - GN CALL ERROR                                                                                                 |
| X'A014' | QMR50   | UNLOAD - REJECT CALL ERROR                                                                                             |
| X'A018' | QMR50   | UNLOAD - RELEASE CALL ERROR                                                                                            |
| X'A01C' | QMR50   | UNLOAD - INVALID CALL TYPE RECEIVED                                                                                    |
| X'A020' | QMR50   | UNLOAD - INVALID CALL SEQUEUCE                                                                                         |
| X'A024' | QMR50   | UNLOAD - SELECT MESSAGE ERROR                                                                                          |
| X'A028' | QMR50   | UNLOAD - SYSTEM NOT SHARE QUEUES                                                                                       |
| X'A02C' | QMR50   | UNLOAD - CMD QUEUE TYPE INVALID                                                                                        |
| X'A030' | QMR50   | UNLOAD - QUEUENAME INVALID                                                                                             |
| X'A034' | QMR50   | UNLOAD - DESTINATION IS INVALID                                                                                        |
| X'A038' | QMR50   | UNLOAD - CONFLICT BETWEEN QDFLG1 AND QDQCBDQ                                                                           |
| X'A03C' | QMR50   | UNLOAD - 1ST RECORD RETURNED NOT 1ST OF MESSAGEQ                                                                       |
| X'A040' | QMR50   | UNLOAD - MESSAGE CHAIN IS BROKEN                                                                                       |
| X'A044' | QMR50   | UNLOAD - ERROR GET/REL DFSBCB STORAGE                                                                                  |
| X'A048' | QMR50   | UNLOAD - QDQCBDQ DOES NOT POINT TO A QUEUE BLOCK                                                                       |
| X'A04C' | QMR50   | UNLOAD LOCAL QUEUES CONTINUATION TYPE ERROR PRIOR UNLOAD CALL WAS IN ERROR                                             |
| X'A050' | QMR50   | UNLOAD LOCAL QUEUES CONTINUATION TYPE REQUEST, THE SMB SUSPEND QUEUE WAS DRAINED DURING THE PROCESS OF BEING UNLOADED  |
| X'A054' | QMR50   | UNLOAD LOCAL QUEUES CONTINUATION TYPE REQUEST, THE SMB SUSPEND QUEUE WAS MODIFIED DURING THE PROCESS OF BEING UNLOADED |
| X'A058' | QMR50   | UNLOAD LOCAL QUEUES CONTINUATION TYPE REQUEST, THE CNT QUEUE WAS DRAINED DURING THE PROCESS OF BEING UNLOADED          |

Table 218. AIBREASN Codes Set by DFSQMRQ0 (continued)

| Code    | Routine  | Error Message                                                                                                            |
|---------|----------|--------------------------------------------------------------------------------------------------------------------------|
| X'A05C' | QMR50    | UNLOAD LOCAL QUEUES CONTINUATION TYPE REQUEST, THE CNT DEQUEUE POINTER WAS MODIFIED DURING THE PROCESS OF BEING UNLOADED |
| X'A060' | QMR50    | UNLOAD LOCAL QUEUES CONTINUATION TYPE REQUEST, THE SMB QUEUE WAS DRAINED DURING THE PROCESS OF BEING UNLOADED            |
| X'A064' | QMR50    | UNLOAD LOCAL QUEUES CONTINUATION TYPE REQUEST, THE SMB DEQUEUE POINTER WAS MODIFIED DURING THE PROCESS OF BEING UNLOADED |
| X'A068' | QMR50    | UNLOAD LOCAL QUEUES, REQUESTED DESTINATION IS BEING READ BY ANOTHER TASK                                                 |
| X'A06C' | QMR50    | ERROR TERMINATING IMS CONVERSATION                                                                                       |
| X'A070' | QMR50    | COMMAND RESPONSE MESSAGE CAN'T BE DELETED                                                                                |
| X'B004' | QMR40    | RECOVER COMMAND ERROR                                                                                                    |
| X'B008' | QMR40    | RECOVER COMMAND ERROR                                                                                                    |
| X'B00C' | QMR40    | RECOVER COMMAND ERROR                                                                                                    |
| X'B010' | QMR40    | RECOVER COMMAND ERROR                                                                                                    |
| X'B014' | QMR40    | RECOVER COMMAND ERROR                                                                                                    |
| X'B018' | QMR40    | RECOVER COMMAND ERROR                                                                                                    |
| X'B01C' | QMR40    | RECOVER COMMAND ERROR                                                                                                    |
| X'B020' | QMR40    | RECOVER COMMAND ERROR                                                                                                    |
| X'B024' | QMR40    | RECOVER COMMAND ERROR                                                                                                    |
| X'B028' | QMR40    | RECOVER COMMAND ERROR                                                                                                    |
| X'B02C' | QMR40    | RECOVER COMMAND ERROR                                                                                                    |
| X'B030' | QMR40    | RECOVER COMMAND ERROR                                                                                                    |
| X'B034' | QMR40    | RECOVER COMMAND ERROR                                                                                                    |
| X'B038' | QMR40    | RECOVER COMMAND ERROR                                                                                                    |
| X'B03C' | QMR40    | RECOVER COMMAND ERROR                                                                                                    |
| X'B040' | QMR40    | RECOVER COMMAND ERROR                                                                                                    |
| X'B044' | QMR40    | RECOVER COMMAND ERROR                                                                                                    |
| X'B048' | QMR40    | RECOVER COMMAND ERROR                                                                                                    |
| X'B04C' | QMR40    | RECOVER COMMAND ERROR                                                                                                    |
| X'C000' | QMR20    | SELECT SHOULD NOT OCCUR ERROR                                                                                            |
| X'C004' | QMRA0    | SELECT CRITERIA DFSSQQRQY ERROR                                                                                          |
| X'C008' | QMRA0    | SELECT CRITERIA DFSPPOOL ERROR                                                                                           |
| X'C00C' | QMRA0    | INVALID CMD CALL                                                                                                         |
| X'C010' | QMRA0    | DFSCBTS SCAN/FIND ERROR                                                                                                  |
| X'D004' | QMR70    | LOAD/INSERT - INVALID CALL TYPE REC                                                                                      |
| X'D008' | QMR70    | LOAD - ERROR CANCELING MESSAGE                                                                                           |
| X'D00C' | QMR70    | XFER - ERROR TRANSFERING MESSAGE                                                                                         |
| X'E000' | DFSQMR00 | QSN exit started too many BMPs                                                                                           |
| X'E004' | DFSQMRD0 | QC/QSN INVALID CMD CALL                                                                                                  |

Table 218. AIBREASN Codes Set by DFSQMRQ0 (continued)

| Code    | Routine  | Error Message                                                                                           |
|---------|----------|---------------------------------------------------------------------------------------------------------|
| X'E008' | DFSQMRD0 | QC/QSN SUPPORTED ONLY IN QCF ENVIRONMENT                                                                |
| X'E00C' | DFSQMRD0 | QC/QSN CMD CALL NO QQSN BLOCK                                                                           |
| X'E010' | DFSQMRD0 | QC/QSN COMMAND ACTION INVALID                                                                           |
| X'E014' | DFSQMRD0 | QC/QSN CMD CALL INVALID ITASK                                                                           |
| X'E018' | DFSQMRD0 | ERROR GET/REL AN AWE                                                                                    |
| X'E01C' | DFSQMRD0 | QC/QSN CMD CALL RECEIVED ERROR CODE FROM SELECT                                                         |
| X'E020' | QMRG0    | INVALID FUNCTION PASSED TO QC LOAD AND QUERY QUEUE SPACE BNOTIFICATION TABLE                            |
| X'E024' | DFSQMRG0 | QC LOAD CMD CALL RECEIVED ERROR GET/REL DFSPPOOL STORAGE SERVICES                                       |
| X'E028' | DFSQMRG0 | QC LOAD CMD PROCESSING - the value for QUOTNOTF is invalid                                              |
| X'E02C' | DFSQMRG0 | INVALID CMD CALL RECEIVED, ONLY /QC-LTBL IS CURRENTLY SUPPORTED                                         |
| X'E030' | DFSQMRG0 | INVALID QUEUE UPPER AND/OR LOWER THRESHOLD PERCENT                                                      |
| X'E034' | DFSQMRG0 | ERROR GET/REL AN AWE                                                                                    |
| X'E038' | DFSQMRG0 | /QC-LTBL AND /QC-QTBL NOT ACTIVE                                                                        |
| X'E03C' | DFSQMRD0 | QC/QSN COMMAND IS INVALID                                                                               |
| X'F000' |          | RESERVED                                                                                                |
| X'F004' | QMRC0    | INVALID FUNCTION PASSED TO ENVIRONMENT STATISTICS ROUTINE                                               |
| X'F008' | QMRC0    | IMS IS IN THE PROCESS OF SHUTDOWN OR QUIESCING                                                          |
| X'F00C' | QMRC0    | SHARED QUEUES ENVIRONMENT, NO SHARED QUEUES MASTER CONTROL BLOCK (SCDSQM)                               |
| X'F010' | QMRC0    | SHARED QUEUES ENVIRONMENT, NO STRUCTURE BLOCK (SQMSQSM)                                                 |
| X'F014' | QMRC0    | IMS INTERNAL ERROR                                                                                      |
| X'F018' | QMRC0    | CQS NOT AVAILABLE TO PROCESS THE CQS QUERY REQUEST                                                      |
| X'F01C' | QMRC0    | CQS RETURNED AN UNSUCCESSFUL RETURN CODE FOR THE CQS QUERY REQUEST                                      |
| X'F020' | QMRC0    | IN PROCESSING THE QCF ENVIRONMENT STATISTICS REQUEST STORAGE WAS NOT OBTAINED                           |
| X'F024' | QMRC0    | IN PROCESSING THE QCF ENVIRONMENT STATISTICS THE LIST PASSED TO CQS CONTAINED AN INVALID STRUCTURE NAME |
| X'F028' | QMRC0    | IN PROCESSING THE QCF ENVIRONMENT STATISTICS REQUEST THE DFSSQQRQ RETURNED A NON-ZERO RETURN CODE       |

## AIB Return Codes Set by DFSQMRQ0

X'00000F0' is a unique AIB return code assigned to the message queue manager message requester 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'00000F0' indicate IMS errors that are not specific to message requeuing. To analyze these return codes and their associated reason codes, see *IMS Version 9: Messages and Codes, Volume 1*.

Each AIBREASN code associated with AIB return code X'00000F0' 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'00000F0'.

---

**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 - INVLD FUNC PASSED TO QMRQ0 ENTRY**

**Explanation:** DFSQMRQ0 was called with an invalid function code in R1.

**Programmer Response:** This is an 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' ENTRY - INVALID CMD/GCMD/ISRT CALL**

**Explanation:** DFSQMRQ0 was called with an invalid CMD, GCMD, or ISRT call, or invalid sequence of these calls. The very first MRQ/QCF call to IMS must be a CMD or ISRT call. CMD calls must pass a valid command work area. ISRT calls must pass a valid MRQ prefix and IMS message.

**Programmer Response:** If the caller is the MRQ or QCF BMP, then this is either a MRQ, QCF, or IMS error. If the caller is a user BMP using the MRQPSB, this is a user error. The MRQPSB is for the exclusive use of MRQ/QCF. This problem can also occur if the release of MRQ or QCF is not supported on the IMS release. Verify that this is a valid MRQ/QCF release for this release of IMS.

---

**X'0020' ENTRY - INVALID MRQ FUNCTION**

**Explanation:** DFSQMR10 did not recognize the MRQ function code.

**Programmer Response:** This is an internal error. Trace back to caller of DFSQMR10. Function code is stored in QMRWFCN.

---

**X'0024' ENTRY - INVALID DFSQMR10 FUNCTION**

**Explanation:** Function not supported by DFSQMR10.

**Programmer Response:** This is an internal error. Trace back to caller of DFSQMR10. DFSQMR10 function is stored in QMRWFCN2.

---

**X'0028' ENTRY - INVALID SPANNED COMMAND**

**Explanation:** Invalid spanned command data was received from the MRQ BMP.

**Programmer Response:** This is an internal error. Either the command segment first or last flags, or both, or the workarea spanned flags are incorrect. The command data is in the I/O area pointed to by either Reg6 or QMRWIO in MRQWORK. The first/last flags are at flag2 (MRZZ2). The MRQ spanned flags are in MRQWORK (whose address is in REG5) flag QMRCFLG2.

---

**X'002C' ENTRY - INVALID BUILD MRQ PREFIX CALL**

**Explanation:** The Build MRQ prefix routine in DFSQMR10 was called to build a MRQ prefix, but the current function either did not have a prefix buffer or a message to build the prefix from.

**Programmer Response:** This is an internal error.

---

**X'0030' ENTRY - ERROR DURING INIT QC FUN CALL**

**Explanation:** DFSQMR10 encountered an error during initialization of the Queue Control (QC) function call.

**Programmer Response:** Locate the QC command buffer pointed to by QCMRQCMDP and verify it is a valid QC command. If QC-LTBL command, locate the QSN table being loaded and verify it has a valid length.

---

**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 FFFFFFFF 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 be 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 MSGIDSTM 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 FFFFFFFF 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. MSGFNQRU 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 MSGIDSTM starts with an FFFFFFFF 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'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 DFSPPOOL STORAGE**

**Explanation:** A DFSPPOOL 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 DFSPPOOL 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  
MSGSEPH**

**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 queued 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 queuing 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'10F4' ISRT - ERROR CREATING DYNAMIC LNB**

**Explanation:** DFSQMRQ0 called the create dynamic LNB routine but the create was unsuccessful.

**Programmer Response:** Locate the message and verify that the destination name at MSGODSTN is a valid MSNAME and is unique name in the IMS system.

---

**X'10F8' INIT - CAN'T FIND SYS PREFIX SEG MSGSIPEX**

**Explanation:** The system prefix segment could not be located. Segment cannot be located.

**Programmer Response:** Locate the message and verify the system prefix is present following the basic prefix. The system prefix code is 81. This is an internal error. Message is not valid without a system prefix.

---

**X'10FC' INIT - ERROR LOADING MODULE DFSTSPC0**

**Explanation:** The UTC to LOCAL time conversion routine could not be loaded.

**Programmer Response:** Verify module DFSTSPC0 is in the IMS RESLIB and can be loaded.

---

**X'1100' ISRT - /MSV CMD MESSAGE CANCELED**

**Explanation:** Message was a /MSVERIFY command message and was canceled.

**Programmer Response:** Messages containing /MSVERIFY data are canceled by MRQ because the data may no longer be valid. This is a normal condition.

---

**X'1104' INIT - QMRQWORK SHOULD NOT OCCUR ERROR**

**Explanation:** Either the QMRQWORK area could not be located, was determined to be invalid, or existed when it shouldn't have. In other words, prior use not freed or cleaned up.

**Programmer Response:** QMRQWORK is pointed to by QSAPWKAD. It should either be zero or an address in the HIOP pool. Usage varies by function (QMRWFCN, QMRWFCN2) being performed. This is an internal IMS error.

---

**X'1108' MSGPROC - CAN'T FIND TMR PREFIX MSGMSC**

**Explanation:** Message flag indicates TMR prefix segment is present but segment cannot be located.

**Programmer Response:** Locate the message and verify the prefix segment exists. The TMR prefix segment code is 8C. This is an internal IMS error.

---

**X'110C' MSGPROC - DFSSQQRV INVALID QUEUE TYPE**

**Explanation:** DFSQMR20 was called to query a queue type, but the type requested was invalid.

**Programmer Response:** MRQTYPE in the DFSMRCMD data passed from the MRQ BMP is not a valid shared queues queue type. This is an IMS or MRQ error.

---

**X'1110' INIT - INVALID INCLUDE/EXCLUDE ENTRY**

**Explanation:** DFSQMR10 received an invalid include or exclude entry type from the MRQ BMP.

**Programmer Response:** MRQSELECT pointer in MRQWORK points to the include or exclude table. R4 in QMRWESAV in MRQWORK contains the address of the entry (MRSELROW) in error. This is an IMS or MRQ error.

---

**X'1114' INIT - INVALID MRQWORK INIT CALL**

**Explanation:** DFSQMR10 QMRQINIT call was made to reinitialize the workarea (MRQWORK) and the call request was determined to be invalid.

**Programmer Response:** Currently, MRQWORK is reinitialized at each new insert or command call. R10, byte 2, in the REG0-15 savearea, contains the call type which should be MRQISRT (04) or MRQCMD (38). See macro DFSQMGR for list of MRQ function codes (QMRWFCN). R10, byte 3, is the QMRQINIT code (1C). Trace the call back to the caller of DFSQMR10/QMRQINIT. This is an internal IMS/MRQ error.

---

**X'1118' ISRT - QBUFF DIDN'T FIT IN DEP RGN COMM AREA**

**Explanation:** A message queue buffer was received from the MRQ BMP for a Load/Insert request, and the PSTVS0 flag was set to indicate it did not fit in the BMP to IMS region communications area.

**Programmer Response:** The dependent region communications area (DIRCA or also called PSBNDXSZ) size is supposed to be sufficient to handle the largest QBUF. If this condition is set, this is an internal IMS error.

---

**X'111C' ISRT - DFSRAC6 ERROR GETTING UTOKEN FOR APPC**

**Explanation:** A message is being inserted and is being converted to APPC. RACF was called to and issued a RACROUTE REQUEST=VERIFYX to obtain a UTOKEN for the APPC prefix of the message. The RACF call returned a non zero return code.

**Programmer Response:** The return code from the call to the IMS RACF interface routine (DFSRAC60) is in R15 in the REG0-15 save area. R1 is the PARMLIST address which is the QMRWLWA2 area within MRQWORK. PARMLIST+1C = USERID, PARMLIST+20 = GROUPNAME, PARMLIST+48 = APPC PLUNAME.

---

**X'1120' ISRT - QUEUE BUFFER FLAG ERROR**

**Explanation:** Error detected in either the MRQ prefix Z2 flag (MRPREZZ2) or queue buffer flag2 (MSGCFLG2) passed by the MRQ insert/load BMP.

**Programmer Response:** The MRQ Prefix that precedes the Qbuffer/Message being loaded or inserted, did not have the correct first or last flags (for example: Flags MRPZZFST or MRPZZLST of flag MRPREZZ2). See macro DFSMRQPF for MRQ prefix mapping. MRQ prefix address is in R4 of the REG0-15 area. Or, the message Qbuffer, MSGPRFX, did not have the correct first or last flags, (for example: Flags MSGFFRST or MSGFLAST of flag MSGFLAGS). See macro QLOGMSGP for the mapping of this prefix. The Qbuffer prefix address is in R6 of the REG0-15 area.

If a spanned buffer is being passed, then the prior buffer inserted might be the one with incorrect flags. The prior buffer flags and status is saved in fields QMRWBF1 and QMRWBF2 of the MRQWORK area. MRQWORK area is in R5 of the REG0-15 area and is mapped by the DFSMRQWK macro.

This is either an MRQ or IMS internal error, or the data being passed to IMS using the MRQ insert/load function is invalid.

---

**X'1124' ISRT - SEGMENT FLAG ERROR**

**Explanation:** Error detected in the segment Z1 flag (MSGXFLG1) in the queue buffer passed by the MRQ insert/load BMP.

**Programmer Response:** The segment of the message about to be inserted to the message queue was determined to have an invalid first or last flag (for example: Flags MSGX1FST or MSGX1LST of flag MSGXFLG1). The segment address is in R4 of the REG0-15 area and is mapped by the QLOGMSGP macro. If this is a spanned segment, the incorrect flag setting might be in the prior segment of the message being inserted. The prior segment flag and status is saved in fields QMRWSG1 and QMRWSG2 of the MRQWORK area. MRQWORK address is in R5 of the REG0-15 area and is mapped by the DFSMRQWK macro.

This is either an MRQ or IMS internal error, or the data being passed to IMS using the MRQ insert/load function is invalid.

---

**X'1128' ISRT - INVALID QUEUE BUFFER DETECTED**

**Explanation:** An invalid queue buffer was passed by the MRQ BMP on either a load or insert function.

**Programmer Response:** The QBUFFER received by INSERT/LOAD is validated by adding up the prefix length and the segment lengths and comparing them to the qbuffer length. If the two are not equal, the buffer is considered to be invalid. The buffer address is in REG6 in the REG0-15 save area. The buffer is mapped by the QLOGMSGP macro.

This error is either an MRQ or IMS internal error, or the data being passed to IMS by the MRQ INSERT/LOAD function is invalid.

---

**X'112C' INIT/CLEANUP - ISWITCH FAILURE**

**Explanation:** Either an ISWITCH to the CTL region or an ISWITCH RETURN to the MRQ/QCF dependent region failed.

**Programmer Response:** Save area QMRWESAV in MRQWORK contains the registers at time of error (R0 - R15). R15 is the ISWITCH error code. R1 is the module ID that was issuing the ISWITCH:

- QMR1 = DFSQMR10
- QMR7 = DFSQMR70
- QMR9 = DFSQMR90

Flag QMRWFLG9=QMRW9SWI, if off, indicates this is a ISWITCH to CTL failure. If on, this is an ISWITCH RETURN to DEP failure. This is an internal IMS error.

---

---

**X'1130'      INSERT - ISRT QBUFF THRESHOLD  
EXCEEDED**

**Explanation:** The ISRT Queue buffer threshold count was exceeded.

**Programmer Response:** This condition is detected by either a user queue space notification exit (DFSQSPC0) or the QCF queue space notification exit (DFSQMR10 which is activated by link editing IQCQMRH0 to IMS RESLIB as DFSQMRH0). If the QCF exit is being used, refer to the section “Preventing Message Queue Overflow (Nonshared Queues)” in the *IMS Queue Control Facility for z/OS User's Guide*. In either case, user or QCF exit, the exit set flag (see note below) caused the qbuffer threshold to be exceeded. The count of long and short Q-buffers being inserted by this message is at fields QMRLBCNT/QMRSBCNT in DFSMRQWK work area.

**Note:** The exit set flag, QMGROFL3=QMGRO3NO, located in the DFSQMGR parameter list, that was built in the PSTDCA area as a result of detecting the current message being inserted.

These messages will need to be reinserted at a time when the queue usage is not as high, or the queue data sets will need to be increased, or the value for the threshold exceeded will need to be increased.

---

**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 FMQSELECT. 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 requester 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 SYS PREFIX MSGSSEGM**

**Explanation:** The message prefix should contain a system prefix segment but one could not be located.



**Programmer Response:** Locate the message and verify the system prefix segment exists. The system prefix code is 81. REG1 in the REG0-15 area is the address of the prefix being copied. This is an internal IMS error.

---

**X'4008' CPYPRFX - CAN'T FIND TMR PREFIX MSGMSC**

**Explanation:** Message flag indicates TMR prefix segment is present but segment cannot be located.

**Programmer Response:** Locate the message and verify the prefix segment exists. The TMR prefix segment code is 8C. REG1 in the REG0-15 area is the address of the prefix being copied. This is an internal IMS error.

---

**X'400C' CPYPRFX - CAN'T FIND SYS EXT PREFIX MSDMSE**

**Explanation:** Extended prefix area should contain an extended system segment, but one could not be located.

**Programmer Response:** Locate the message and verify that the prefix segment exists. The system extension prefix segment code is X'8A'. REG1 in the REG0-15 area is the address of the prefix being copied. This is an internal IMS error.

---

**X'4010' CPYPRFX - CAN'T FIND THE MSC PREFIX MSGMSC**

**Explanation:** Message flag indicates MSC prefix segment is present but segment cannot be located.

**Programmer Response:** Locate the message and verify the prefix segment exists. The MSC prefix segment code is 82. REG1 in the REG0-15 area is the address of the prefix being copied. This is an internal IMS error.

---

**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 requester 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 requester 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 requester 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 FMQSELECT. This is either an IMS or MRQ internal error.

---

**X'6010' MRQ/IMS - QBUF COUNT NOT EXPECTED NUMBER**

**Explanation:** During transfer of a qbuffer between the MRQ BMP and IMS, the count of buffers transferred (MRPCOUNT) was in error.

**Programmer Response:** This error is detected by either IMS or the MRQ BMP when transferring messages (QBUFs) during the INSERT, LOAD, BROWSE, RECOVER, and UNLOAD functions. The QBUF transfer count is incremented in the MRQ prefix count field (MRPCOUNT in DFSMRQPF) and checked for one greater than the previous. If this is not the case, the error is issued and the QBUF transfer is rejected. This is either an IMS or MRQ internal error.

---

**X'6014' MSGPROC - MSG WAS CANCELED BY IMS**

**Explanation:** The original message was canceled by IMS and was logged for accounting or message queue recovery purposes. This error is similar to AIBREASN = X'1088', the difference is this is a multi-buffer message and part of the message was inserted to IMS by MRQ, when MRQ detected it was canceled. MRQ issues a PURG DL/I call, requesting IMS purge the message with a AIBREASN = X'6014'

**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 cancelled message. If MSC is sysgened and a 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, message is being inserted (requeued) and was gotten off the log. If processing a BROWSE, UNLOAD, or RECOVER, the message was gotten off the message queue and this condition should not occur.

---

**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.

---

**X'7008' XLATPFX - CAN'T FIND PFX SEG MSGSITMR**

**Explanation:** Message flag indicates TMR prefix segment is present but segment cannot be located.

**Programmer Response:** Locate the message and verify the PREFIX SEGMENT EXISTS. The TMR prefix segment code is 8C. REG1 in the REG0-15 area is the address of the prefix being copied. This is an internal IMS error.

---

**X'700C' XLATPFX - CAN'T FIND PFX SEG MSGMSC**

**Explanation:** Message flag indicates MSC prefix segment is present but segment cannot be located.

**Programmer Response:** Locate the message and verify the PREFIX SEGMENT EXISTS. The MSC prefix segment code is 82. This is an internal IMS error.

---

**X'7010' XLATPFX - CAN'T FIND PFX SEG MSGMSCE**

**Explanation:** Message flag indicates MSC prefix EXTENSION segment is present but segment cannot be located.

**Programmer Response:** Locate the message and verify the PREFIX SEGMENT EXISTS. The MSC prefix extension segment code is 8B. This is an internal IMS error.

---

**X'7014' XLATPFX - ERROR CONVERTING MESSAGE TIME**

**Explanation:** A non zero return code was returned from the time conversion routine while converting local time to UTC or UTC time to local.

**Programmer Response:** The probable cause is an incorrect time field in one of the message prefixes. Locate the time conversion work area in QMRQWORK at label QMRDSTWK. If error occurred converting local time to UTC, the return code is in QMRCVTM1. If error

| occurred converting UTC to local time, it's in  
 | QMRCVTM2. Return codes are from either DFSCVTM  
 | or DFSTSPC. Local time being converted is at  
 | QMRWLOCL. UTC time is at QMRDSUTC. Local time  
 | fields in the message are at MSGTMFAP, MSGMSCTS.  
 | UTC time fields in the message are at MSGUTC,  
 | MSGMSCEX, MSGMSCTS.

---

**X'7018' XLATPFX - CAN'T FIND PFX  
 MSGEPHDR**

| **Explanation:** Message flag indicates HEADER prefix  
 | extension segment is present, but segment cannot be  
 | located.  
 | **Programmer Response:** Locate the message and  
 | verify that flag MSGC2EPH is set and the header prefix  
 | (type 86) exists. All messages from IMS 5.1 and up  
 | should contain this flag and prefix segment. This is an  
 | IMS error.

---

**X'8004' QMR30 - BROWSE - SYSTEM NOT  
 SHARED QUEUES**

| **Explanation:** Command BROWSE was issued in a  
 | non-shared queues IMS system.  
 | **Programmer Response:** Make appropriate change.

---

**X'8008' QMR30 - INVALID FUNCTION PASSED  
 TO BROWSE**

| **Explanation:** Invalid call to DFSQMR30. Register 0  
 | did not contain X'28' (MRQCLEAN) or X'34' (MRQCMD)  
 | or X'38' (MRQGCMND).  
 | **Programmer Response:** REG0 in REG0-REG15 area  
 | contains the function code. REG14 is the address of the  
 | caller of DFSQMR30 (BALR reg). Trace the call back to  
 | the caller of DFSQMR30.

---

**X'800C' QMR30 - BROWSE RECEIVED ERROR  
 CODE FROM SELECT**

| **Explanation:** DFSQMR20 returned with RC = X'08' in  
 | QNAME selection call.  
 | **Programmer Response:** Trace back to DFSQMR20  
 | to determine the cause.

---

**X'8010' QMR30 - BROWSE COMMAND ERROR**

| **Explanation:** CQS BROWSE the READY Q returned  
 | partial data. The data object size was larger than 32K.  
 | **Programmer Response:** Locate the message and find  
 | out where this message came from. REG8 in  
 | REG0-REG15 area contained the message address.

---

**X'8014' QMR30 - BROWSE COMMAND ERROR**

| **Explanation:** No data objects returned while CQS  
 | BROWSE the READY Q.  
 | **Programmer Response:** Dump the associated queue  
 | to verify if any message exists. If so, it is an internal  
 | error.

---

**X'8018' QMR30 - BROWSE COMMAND ERROR**

| **Explanation:** CQS BROWSE the READY Q failed.  
 | **Programmer Response:** This is an internal error.  
 | Locate the CQS reason code (CQSRSNCD) in the  
 | parameter list (QMRWLWA) to determine the cause.

---

**X'801C' QMR30 - BROWSE COMMAND ERROR**

| **Explanation:** DFSQMR20 returned with RC = X'08' in  
 | message selection call.  
 | **Programmer Response:** Trace back to DFSQMR20  
 | to determine the cause.

---

**X'8020' QMR30 - BROWSE COMMAND ERROR**

| **Explanation:** The message segment form the READY  
 | Q was not the first segment.  
 | **Programmer Response:** Locate the message and  
 | verify the contents. REG8 in REG0-REG15 area  
 | contained the message address. This is an internal IMS  
 | error.

---

**X'8024' QMR30 - BROWSE COMMAND ERROR**

| **Explanation:** The TMR prefix segment could not be  
 | located.  
 | **Programmer Response:** Locate the message and  
 | verify the prefix segment exists. The TMR prefix  
 | segment code is X'8C'. REG8 in REG0-REG15 area  
 | contained the message address. This is an internal IMS  
 | error.

---

**X'8028' QMR30 - BROWSE COMMAND ERROR**

| **Explanation:** CQS BROWSE the STAGING Q  
 | returned partial data. The data object size was larger  
 | than 32K.  
 | **Programmer Response:** Locate the message and find  
 | out where the message came from. REG8 in  
 | REG0-REG15 area contained the message address.

---

**X'802C' QMR30 - BROWSE COMMAND ERROR**

| **Explanation:** No data objects returned while CQS  
 | BROWSE the STAGING Q.  
 | **Programmer Response:** Dump the associated queue  
 | to verify if any message exists. If so, it is an internal  
 | error.

---

**X'8030' QMR30 - BROWSE COMMAND ERROR**

**Explanation:** CQS BROWSE the STAGING Q failed.

**Programmer Response:** This is an internal error. Locate the CQS reason code (CQSRSNCD) in the parameter list (QMRWLWA) to determine the cause.

---

**X'8034' QMR30 - BROWSE COMMAND ERROR**

**Explanation:** The message segment from the STAGING Q was not the middle segment.

**Programmer Response:** Locate the message and verify the contents. REG8 in REG0-REG15 area contained the message address. This is an internal IMS error.

---

**X'8038' QMR30 - BROWSE COMMAND ERROR**

**Explanation:** The message segment from the STAGING Q was not the last segment.

**Programmer Response:** Locate the message and verify the contents. REG8 in REG0-REG15 area contained the message address. This is an internal IMS error.

---

**X'803C' QMR30 - BROWSE COMMAND ERROR**

**Explanation:** The request was terminated because the RESYNC was not done between IMS and CQS.

**Programmer Response:** Reissue the request after the RESYNC was done.

---

**X'8040' QMR30 - BROWSE INVALID DESTINATION**

**Explanation:** The BROWSE request found the destination field (BCURLNAM) to be zero.

**Programmer Response:** The destination is a required field for the browse of the local queues. This is an IMS error.

---

**X'8044' QMR30 - BROWSE LOCAL QUEUES CONTROL BLOCK ERROR**

**Explanation:** The BROWSE request found the QDEST block to be in error. One of the following conditions was found:

- QDFLG1 indicated that there were messages on this control block and QDQCBDDQ was zero.
- QDFLG1 indicated that there were messages on this control block and QDQCBDDQ did not point to a queue block if the destination was a cnt type block.

**Programmer Response:** This is an IMS error.

---

**X'8048' QMR30 - BROWSE LOCAL QUEUES DESTINATION TYPE ERROR ON A MULTI-RECORD MESSAGE**

**Explanation:** The BROWSE request found, during the processing of the second through the nth record of a multi-record message, that the destination address in field (BCURLNAM) and the destination address in the current PCB were not the same.

**Programmer Response:** This is an IMS error.

---

**X'804C' QMR30 - BROWSE LOCAL QUEUES CONTINUATION TYPE ERROR ON A MULTI-RECORD MESSAGE**

**Explanation:** The BROWSE request found during the processing of the second through the nth record of a multi-record message that the field pointed to by (BMRQQPCB) did not contain a valid token in field (QTPRRN).

**Programmer Response:** This is an IMS error.

---

**X'8050' QMR30 - QSN BLOCK ADDRESS IS ZERO**

**Explanation:** The BROWSE was called to process the Queue Space Notification (QSN) Queue and was passed an invalid or zero QSN block address in field MRCURQQSN.

**Programmer Response:** If MRCURQQSN is zero, this is probably an internal Browse or select error. Trace back to where the field was set. If invalid, this is probably a bad QSN block on the QSN chain or an overlaid QSN block. Verify this chain and the blocks on it.

This is an IMS internal error.

---

**X'8054' QMR30 - BROWSE AREA PARM NOT SET**

**Explanation:** The BROWSE request could not find a valid area to process. Valid areas are LOCAL, GLOBAL, OVERFLOW, and QUEUE SPACE NOTIFICATION (QSN). Global and Overflow are valid for Shared Queues only.

**Programmer Response:** Verify that a valid AREA = LOCAL, GLOBAL, OVERFLOW, or QSN was specified on the function control card passed to QCF. If valid, then verify that IMS is processing a valid area, as indicated by flag BMRQFLG2, if a BROWSE function was requested; or in flag QMRQFLG2, if a QUERY function was requested and QUERY called BROWSE internally. This is either a QCF or IMS error.

---

**X'9004' QMR60 - QUERY - SYSTEM NOT SHARED QUEUES**

**Explanation:** QUERY Command was issued in a non-shared queues IMS system.

**Programmer Response:** The MRQ Query function is only supported in a shared queues environment.

---

**X'9008' QMR60 - INVALID FUNCTION PASSED TO QUERY**

**Explanation:** Invalid call to DFSQMR60. Register 0 did not contain either X'28' (MRQCLEAN), X'34' (MRQCMD), or X'38' (MRQGCMD).

**Programmer Response:** REG0 in REG0-REG15 area contains the function code. REG14 is the address of the caller of DFSQMR60 (BALR reg). Trace the call back to the caller of DFSQMR60.

---

**X'900C' QMR60 - QUERY RECEIVED ERROR CODE FROM SELECT**

**Explanation:** Call to DFSQMR20 to select a message queue name resulted in a error return code of eight or greater.

**Programmer Response:** REG15 in the REG0-REG15 save area contains the return code. CMDQNAME in MRQWORK is the queue name being processed or the last queue name successfully processed. Trace the error back to DFSQMR20.

---

**X'9010' QMR60 - QUERY - CMD QUEUE TYPE INVALID**

**Explanation:** An invalid or no queue type was passed on the QUERY command call from the MRQ BMP.

**Programmer Response:** MQTYPQUE in MRQCMDWK contains either zero or invalid queue type or types. The cold queue is an invalid QUERY queue type. REG8 in the REG0-REG15 save area contains the MRQCMDWK address which contains the QUERY command from the MRQ BMP.

---

**X'9014' QMR60 - QUERY - NO MESSAGE RETURNED ON INTERNAL CALL TO BROWSE**

**Explanation:** While querying either the APPC, OTMA, or Cold queues, the query processor called browse internally to get the message and extract information for the CQSQRQT entry for the queue name. Browse returned no messages for the queue name.

**Programmer Response:** REG15 in the REG0-15 save area contains the BROWSE return code of 4. REG6 is the address of the DFSSQRY buffer with queue names that have messages. REG7 is the current queue name entry that encountered the error. REG2 has the queue type from MQCURQNM in the MRSELWK area;

the queue type is one of the following:

- 01 = APPC
- 02 = COLD
- 08=OTMA

This is most likely either a IMS or CQS error.

---

**X'9018' QMR60 - QUERY - RETURN CODE ERROR ON INTERNAL CALL TO BROWSE**

**Explanation:** While querying either the APPC, OTMA, or Cold queues, the query processor called browse internally to get the message and extract information for the CQSQRQT entry for the queue name. Browse returned an error code for the query queue name.

**Programmer Response:** REG15 in the REG0-15 save area contains the BROWSE return code. REG6 is the address of the DFSSQRY buffer with queue names that have messages. REG7 is the current queue name entry that encountered the error. REG2 has the queue type from MQCURQNM in the MRSELWK area which is one of the following:

- 01 = APPC
- 02 = COLD
- 08=OTMA

This is probably either a IMS or CQS error.

---

**X'901C' QMR60 - QUERY - ERROR LOCATING APPC/OTMA PFX**

**Explanation:** While querying either the APPC or OTMA queues, a message was returned by an internal BROWSE call; A DFSGMPL request was issued to locate either the APPC or OTMA prefix of the message, to extract information from the prefix for the CQSQRQT entry for the queue name. The DFSGMPL call encountered an error while trying to locate the prefix.

**Programmer Response:** REG1 in the REG0-15 save area is the address of the message, which is in the BROWSE buffer MRQBROMC. REG6 is the address of DFSSQRY buffer with queue names that have messages. REG7 is the current queue name entry that encountered the error. The APPC/OTMA prefix is in the extended prefix area of the message and is a type X'87'. The message prefixes are mapped by macro QLOGMSGP. The message needs to be analyzed to determine the error. This is an IMS error.

---

**X'9020' QMR60 - QUERY - ERROR LOCATING TMR PREFIX**

**Explanation:** While querying either the APPC or OTMA queues, a message was returned by an internal BROWSE call; A DFSGMPL request was issued to locate the Transaction Manager Routing (TMR) prefix in the message to extract information from the prefix for

| the CQSQRQT entry for the queue name. The  
| DFSMGPL call encountered an error while trying to  
| locate the prefix.

| **Programmer Response:** REG1 in the REG0-15 save  
| area is the address of the message which is in the  
| BROWSE buffer MRQBROMC. REG6 is the address of  
| the DFSSQQRQY buffer with queue names that have  
| messages. REG7 is the current queue name entry that  
| encountered the error. The TMR prefix is in the  
| extended prefix area of the message and is a type  
| X'8C'. The message prefixes are mapped by macro  
| QLOGMSGP. The message needs to be analyzed to  
| determine the error. This is an IMS error.

---

**X'9024' QMR60 - QUERY - SHOULD NOT OCCUR ERROR**

| **Explanation:** Query called the select processor  
| (DFSQMR20) to select a queue to query. Select  
| detected there are queues to process but did not return  
| a query buffer or the COLDDQ to process.

| **Programmer Response:** This condition should not  
| occur. The select processor will need to be analyzed to  
| determine the error.

---

**X'9028' QMR60 - QUERY - ERROR FREEING BUFFER DURING CLEANUP**

| **Explanation:** During clean up of a QUERY request at  
| termination of the MRQ BMP, the query buffer obtained  
| by the DFSSQQRQY call was freed with a DFSPPOOL  
| request. A nonzero return code was returned on the call.

| **Programmer Response:** The query buffer address is  
| in REG3 of the REG0-15 save area. The return code is  
| in REG15. This is an IMS error.

---

**X'902C' QMR60 - QUERY - QUERY CALLED BROWSE WITH A GET COMMAND**

| **Explanation:** QUERY (DFSQMR60) called BROWSE  
| (DFSQMR30) with a GET command. Browse is not  
| setup to handle this call.

| **Programmer Response:** This condition should not  
| occur. The query routine or browse routine, or both, will  
| need to be analyzed to determine the error. This is an  
| IMS error.

---

**X'9030' QMR30 - BROWSE - QUERY CALLED BROWSE WITH AN DESTINATION OF ZERO**

| **Explanation:** QUERY (DFSQMR60) called BROWSE  
| (DFSQMR30) with a destination of zero. Destination is  
| required for LOCAL queue.

| **Programmer Response:** This condition should not  
| occur. The query routine or browse routine, or both, will  
| need to be analyzed to determine the error. This is an  
| IMS error.

---

**X'9034' QMR30 - BROWSE COMMAND ERROR**

| **Explanation:** The APPC/OTMA prefix segment could  
| not be located.

| **Programmer Response:** Locate the message and  
| verify the prefix segment exists. The APPC/OTMA  
| segment code is X'87'. REG8 in REG0-REG15 area  
| contained the message address. This is an internal IMS  
| error.

---

**X'9038' QMR60 - QUERY - QUERY CALLED BROWSE WITH AN INVALID QNAME**

| **Explanation:** Query called browse with a request to  
| retrieve either an APPC/OTMA with an invalid queue  
| name.

| **Programmer Response:** Locate the select work area  
| (QMRQSETP) and validate that the queue name is  
| invalid. This is an internal IMS error.

---

**X'903C' QMR60 - QUERY - QUERY CALLED BROWSE WITH AN INVALID QUEUE SPACE NOTIFICATION BLOCK**

| **Explanation:** QUERY (DFSQMR60) called BROWSE  
| (DFSQMR30) with a queue space notification block  
| address of zero. Queue space notification block address  
| (MRCURQQSN) is required for query of QSN.

| **Programmer Response:** This condition should not  
| occur. The query routine and browse routine will need to  
| be analyzed to determine the error. This is an IMS error.

---

**X'A004' QMR50 - UNLOAD - SELECT QUEUE NAME ERROR**

| **Explanation:** Call to DFSQMR20 to select a message  
| queuename resulted in a error return code of eight or  
| greater.

| **Programmer Response:** REG15 in the REG0-REG15  
| save area contains the return code. CMDQNAME in  
| MRQWORK is the queue name being processed or the  
| last queue name successfully processed. Trace the  
| error back to DFSQMR20.

---

**X'A008' QMR50 - UNLOAD - Reserved**

| **Explanation:** Reserved for future use.

---

**X'A00C' QMR50 - UNLOAD - GU CALL ERROR**

| **Explanation:** GET UNIQUE (GU) call to QMGR  
| returned an error return code.

| **Programmer Response:** REG15 in the REG0-REG15  
| save area contains the return code. CMDQNAME in  
| MRQWORK is the queue name being processed.  
| QMRWLWA in MRQWORK contains the QMGR  
| parameter list. Trace the error to QMGR GU processing.

---

**X'A010' QMR50 - UNLOAD - GN CALL ERROR**

**Explanation:** GET NEXT (GN) call to QMGR returned an error code.

**Programmer Response:** REG15 in the REG0-REG15 save area contains the return code. CMDQNAME in MRQWORK is the queue name being processed. QMRWLWA in MRQWORK contains the QMGR parameter list. REG4 is the address of the QTPPCB passed to QMGR. The first two words of the QTPPCB contain the DRRN and buffer address of the message being processed. Trace the error to QMGR GN processing.

---

**X'A014' QMR50 - UNLOAD - REJECT CALL ERROR**

**Explanation:** REJECT (REJ) call to QMGR returned an error return code.

**Programmer Response:** REG15 in the REG0-REG15 save area contains the return code. CMDQNAME in MRQWORK is the queue name being processed. QMRWLWA in MRQWORK contains the QMGR parameter list. REG4 is the address of the QTPPCB passed to QMGR. The first two words of the QTPPCB contain the DRRN and buffer address of the message being rejected. The message may or may not have been successfully rejected (deleted). Trace the error to QMGR reject processing.

---

**X'A018' QMR50 - UNLOAD - RELEASE CALL ERROR**

**Explanation:** RELEASE (REL) call to QMGR returned an error code.

**Programmer Response:** REG15 in the REG0-REG15 save area contains the return code. CMDQNAME in MRQWORK is the queue name being processed. QMRWLWA in MRQWORK contains the QMGR parameter list. PSTQIMSG will contain the DRRN of the message or message chain being released. Some of the messages may have remained locked on the shared queue (for example, not released). Trace the error to QMGR REL processing.

---

**X'A01C' QMR50 - UNLOAD - INVALID CALL TYPE RECEIVED**

**Explanation:** Invalid call to DFSQMR50. REG0 did not contain either X'28' (CLEANUP), X'2C' (GU), X'30' (GN), or X'34' (CMD) call.

**Programmer Response:** REG1 in the REG0-REG15 save area contains the call type. Trace the problem to the caller of DFSQMR50.

---

**X'A020' QMR50 - UNLOAD - INVALID CALL SEQUEUCE**

**Explanation:** Invalid sequence of calls to DFSQMR50. Error is currently set if a GN call is issued for the next message buffer and no message is in progress of being processed (GU).

**Programmer Response:** REG1 in the REG0-REG15 contains the call type. REG6 or MRQUNLMC in MRQWORK points to the Unload buffer. The first two bytes are zero if no message is in progress. The prior message returned on Unload may still be in the buffer. Cause may be the last message contains incorrect first and last flags or logic error between MRQ and IMS.

---

**X'A024' QMR50 - UNLOAD - SELECT MESSAGE ERROR**

**Explanation:** Call to DFSQMR20 to select a message resulted in a error return code of eight or greater.

**Programmer Response:** REG15 in the REG0-REG15 save area contains the return code. CMDQNAME in MRQWORK is the queue name being processed. MRQUNLMC in MRQWORK points to the Unload message being selected. Trace the problem to the select routine DFSQMR20.

---

**X'A028' QMR50 - UNLOAD - SYSTEM NOT SHARE QUEUES**

**Explanation:** Unload could not process the request because the system is not shared queues.

**Programmer Response:** Function is only valid for Shared Queues.

---

**X'A02C' QMR50 - UNLOAD - CMD QUEUE TYPE INVALID**

**Explanation:** An invalid or no queue type was passed on the UNLOAD command call from the MRQ BMP.

**Programmer Response:** MQTYPQUE in MRQCMDWK contains either zero or invalid queue types. The cold queue is an invalid Unload queue type. REG8 in the REG0-REG15 save area contains the MRQCMDWK address which contains the UNLOAD command from the MRQ BMP.

---

**X'A030' QMR50 - UNLOAD - QUEUENAME INVALID**

**Explanation:** An invalid queue name was detected while making a call to the queue manager.

**Programmer Response:** MRQNAME in MRQCMDWK contains either zero or invalid queue name.

---

**X'A034' QMR50 - UNLOAD - DESTINATION IS INVALID**

---

**Explanation:** An invalid destination address was detected while making an Unload request for the local queues.

**Programmer Response:** UCURLNAM in MRQCMDWK contains either zero or invalid destination address.

---

**X'A038' QMR50 - UNLOAD - CONFLICT BETWEEN QDFLG1 AND QDQCBDQ**

---

**Explanation:** A conflict between QDFLG1 and QDQCBDQ while making an Unload request for the local queues. QDFLG1 indicated that messages were on the destination, but QDQCBDQ did not point to a DRRN of a Queue Block or a message.

**Programmer Response:** UMRQQBLK in MRQCMDWK contains the work area that detected this condition. This is an internal IMS error.

---

**X'A03C' QMR50 - UNLOAD - 1ST RECORD RETURNED NOT 1ST OF MESSAGEQ**

---

**Explanation:** A message is being retrieved from the local queues and 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. If not this is an internal IMS error.

---

**X'A040' QMR50 - UNLOAD - MESSAGE CHAIN IS BROKEN**

---

**Explanation:** A message is being retrieved from the local queues and the chain of messages is broken.

**Programmer Response:** This is an internal IMS error.

---

**X'A044' QMR50 - UNLOAD - ERROR GET/REL DFSBCB STORAGE**

---

**Explanation:** DFSBCB call received a non zero return code attempting to get or release storage from the storage pool for a work area.

**Programmer Response:** R15 = return code from DFSBCB call. This is either an internal error or insufficient storage available in the control region private area.

---

**X'A048' QMR50 - UNLOAD - QDQCBDQ DOES NOT POINT TO A QUEUE BLOCK**

---

**Explanation:** A message is being retrieved from the local queues and the field (QDQCBDQ) does not point to a Queue Block.

**Programmer Response:** This is an internal IMS error.

---

**X'A04C' QMR50 - UNLOAD LOCAL QUEUES CONTINUATION TYPE ERROR PRIOR UNLOAD CALL WAS IN ERROR**

---

**Explanation:** A message is being unloaded from the local queues and the prior unload request was terminated with an error.

**Programmer Response:** This is an internal IMS error.

---

**X'A050' QMR50 - UNLOAD LOCAL QUEUES CONTINUATION TYPE REQUEST, THE SMB SUSPEND QUEUE WAS DRAINED DURING THE PROCESS OF BEING UNLOADED**

---

**Explanation:** A message is being unloaded from the local queues SMB suspend queue. The queue was drained during the unload request. The queue should be empty and no action is required.

**Programmer Response:** This is an information type warning.

---

**X'A054' QMR50 - UNLOAD LOCAL QUEUES CONTINUATION TYPE REQUEST, THE SMB SUSPEND QUEUE WAS MODIFIED DURING THE PROCESS OF BEING UNLOADED**

---

**Explanation:** A message is being unloaded from the local queues SMB suspend queue. The queue was modified during the unload request. The unload request must be resubmitted if the SMB suspend queue is to unloaded.

**Programmer Response:** This is an information type warning.

---

**X'A058' QMR50 - UNLOAD LOCAL QUEUES CONTINUATION TYPE REQUEST, THE CNT QUEUE WAS DRAINED DURING THE PROCESS OF BEING UNLOADED**

---

**Explanation:** A message is being unloaded from the local queues CNT QUEUE. The queue was drained during the unload request. The queue should be empty and no action is required.

**Programmer Response:** This is an information type warning.

---

**X'A05C' QMR50 - UNLOAD LOCAL QUEUES CONTINUATION TYPE REQUEST, THE CNT DEQUEUE POINTER WAS MODIFIED DURING THE PROCESS OF BEING UNLOADED**

---

**Explanation:** A message is being unloaded from the local queues CNT QUEUE. The queue was modified



| during the unload request. The unload request must be  
| resubmitted if the CNT QUEUE is to be unloaded.

| **Programmer Response:** This is an information type  
| warning.

---

| **X'A060' QMR50 - UNLOAD LOCAL QUEUES  
| CONTINUATION TYPE REQUEST, THE  
| SMB QUEUE WAS DRAINED DURING  
| THE PROCESS OF BEING UNLOADED**

| **Explanation:** A message is being unloaded from the  
| local queues SMB QUEUE. The queue was drained  
| during the unload request. The queue should be empty  
| and no action is required.

| **Programmer Response:** This is an information type  
| warning.

---

| **X'A064' QMR50 - UNLOAD LOCAL QUEUES  
| CONTINUATION TYPE REQUEST, THE  
| SMB DEQUEUE POINTER WAS  
| MODIFIED DURING THE PROCESS OF  
| BEING UNLOADED**

| **Explanation:** A message is being unloaded from the  
| local queues SMB QUEUE. The queue was modified  
| during the unload request. The unload request must be  
| resubmitted if the SMB QUEUE is to be unloaded.

| **Programmer Response:** This is an information type  
| warning.

---

| **X'A068' QMR50 - UNLOAD LOCAL QUEUES,  
| REQUESTED DESTINATION IS BEING  
| READ BY ANOTHER TASK**

| **Explanation:** A message is being unloaded from the  
| local queues CNT QUEUE. The queue is currently  
| being read by another task. The unload request must be  
| resubmitted if the CNT QUEUE is to be unloaded.

| **Programmer Response:** This is an information type  
| warning.

---

| **X'A06C' QMR50 - ERROR TERMINATING IMS  
| CONVERSATION**

| **Explanation:** During unload (delete) of a message  
| associated with a IMS conversation transaction, an error  
| was encountered while trying to terminate the  
| conversation. The message is deleted but the  
| conversation may not have been terminated.

| **Programmer Response:** This is probably an IMS  
| internal error. Get the 6701-MRQE log record with the  
| AIBREASN=0000A06C error. Fields QCIDIAG1 and  
| QCIDIAG2 will indicate the type of error that was  
| detected in the conversation termination routine  
| (DFSCON20) and QMRWLWA3 will contain the  
| DFSCON20 registers R0 - R15.

---

| **X'A070' QMR50 - COMMAND RESPONSE  
| MESSAGE CAN'T BE DELETED**

| **Explanation:** The messages being unloaded (deleted)  
| are an AOI command response message for an active  
| application program that issued the AOI command (Flag  
| MSGFPADL=MSGSCAMD is set). These messages  
| cannot be deleted.

| **Programmer Response:** This is a normal condition.  
| The message was not unloaded (deleted) by QCF. The  
| message is deleted by IMS when the AOI application  
| program completes processing the response, or reaches  
| a SYNC point, or is terminated. IMS logged a type  
| 6701-MRQE record for this condition and skips to the  
| next destination. The message can be found in the  
| 6701-MRQE record, in the buffer labeled UMRQMSG.

---

| **X'B004' QMR40 - RECOVER COMMAND ERROR**

| **Explanation:** Command /RECOVER was issued in a  
| non-shared queues IMS system.

| **Programmer Response:** Make appropriate change.

---

| **X'B008' QMR40 - RECOVER COMMAND ERROR**

| **Explanation:** Invalid call to DFSQMR40. Register 0  
| did not contain either X'28' (MRQCLEAN) or X'34'  
| (MRQCMD) or X'38' (MRQGCMD).

| **Programmer Response:** Correct and reissue the  
| command.

---

| **X'B00C' QMR40 - RECOVER COMMAND ERROR**

| **Explanation:** CQS BROWSE the COLD Q returned  
| partial data. The data object size was larger than 32K.

| **Programmer Response:** Locate the message and find  
| out from where this message came. REG8 in  
| REG0-REG15 area contained the message address.

---

| **X'B010' QMR40 - RECOVER COMMAND ERROR**

| **Explanation:** CQS BROWSE the COLD Q failed.

| **Programmer Response:** This is an internal error.  
| Locate the CQS reason code (CQSRSNCD) in the  
| parameter list (QMRWLWA) to determine the cause. If  
| reason code = NO DATA OBJECTS RETURNED, dump  
| the COLD Q to verify if any message exists. If so, it is  
| an internal error.

---

| **X'B014' QMR40 - RECOVER COMMAND ERROR**

| **Explanation:** DFSQMR20 returned with RC = X'08' for  
| message selection.

| **Programmer Response:** Trace back to DFSQMR20  
| to determine the cause.

---

**X'B018' QMR40 - RECOVER COMMAND ERROR**

**Explanation:** Subroutine MR4DELET detected that the message segment from the COLD Q was not the first segment.

**Programmer Response:** Locate the message and verify the contents. REG8 in REG0-REG15 area contained the message address. This is an internal IMS error.

---

**X'B01C' QMR40 - RECOVER COMMAND ERROR**

**Explanation:** Subroutine MR4DELET detected that the TMR prefix segment could not be located.

**Programmer Response:** Locate the message and verify the prefix segment exists. The TMR prefix segment code is X'8C'. REG8 in REG0-REG15 area contained the message address. This is an internal IMS error.

---

**X'B020' QMR40 - RECOVER COMMAND ERROR**

**Explanation:** CQSRECVR FUNC=DELETE a message from the COLD Q failed.

**Programmer Response:** This is an internal error. Locate the CQS reason code (CQSRSNCD) in the parameter list (QMRWLWA) to determine the cause.

---

**X'B024' QMR40 - RECOVER COMMAND ERROR**

**Explanation:** CQS DELETE a message from the STAGING Q failed.

**Programmer Response:** This is an internal error. Locate the CQS reason code (CQSRSNCD) in the parameter list (QMRWLWA) to determine the cause.

---

**X'B028' QMR40 - RECOVER COMMAND ERROR**

**Explanation:** Subroutine MR4UNLCK detected that the message segment from the COLD Q was not the first segment.

**Programmer Response:** Locate the message and verify the contents. REG8 in REG0-REG15 area contained the message address. This is an internal IMS error.

---

**X'B02C' QMR40 - RECOVER COMMAND ERROR**

**Explanation:** CQSRECVR FUNC=UNLOCK a message from the COLD Q failed.

**Programmer Response:** This is an internal error. Locate the CQS reason code (CQSRSNCD) in the parameter list (QMRWLWA) to determine the cause.

---

**X'B030' QMR40 - RECOVER COMMAND ERROR**

**Explanation:** Subroutine MR4READ detected that the message segment from the COLD Q was not the first segment.

**Programmer Response:** Locate the message and verify the contents. REG8 in REG0-REG15 area contained the message address. This is an internal IMS error.

---

**X'B034' QMR40 - RECOVER COMMAND ERROR**

**Explanation:** Subroutine MR4READ detected that the TMR prefix segment cannot be located.

**Programmer Response:** Locate the message and verify the prefix segment exists. The TMR prefix segment code is X'8C'. REG8 in REG0-REG15 area contained the message address. This is an internal IMS error.

---

**X'B038' QMR40 - RECOVER COMMAND ERROR**

**Explanation:** CQS BROWSE the STAGING Q returned partial data. The data object size was larger than 32K.

**Programmer Response:** Locate the message and find out from where this message came. REG8 in REG0-REG15 area contained the message address.

---

**X'B03C' QMR40 - RECOVER COMMAND ERROR**

**Explanation:** No data objects for QNAME while CQS BROWSE the STAGING Q for a multi buffer message.

**Programmer Response:** The message may be deleted by other requestor while being browsed. All previously returned segments of this message should be discarded.

---

**X'B040' QMR40 - RECOVER COMMAND ERROR**

**Explanation:** CQS BROWSE the STAGING Q failed.

**Programmer Response:** This is an internal error. Locate the CQS reason code (CQSRSNCD) in the parameter list (QMRWLWA) to determine the cause.

---

**X'B044' QMR40 - RECOVER COMMAND ERROR**

**Explanation:** Subroutine MR4READ detected that the message segment from the STAGING Q was not a middle segment.

**Programmer Response:** Locate the message and verify the contents. REG8 in REG0-REG15 area contained the message address. This is an internal IMS error.

---

**X'B048' QMR40 - RECOVER COMMAND ERROR**

**Explanation:** Subroutine MR4READ detected that the message segment from the STAGING Q was not the last segment.

**Programmer Response:** Locate the message and verify the contents. REG8 in REG0-REG15 area contained the message address. This is an internal IMS error.

**X'B04C' QMR40 - RECOVER COMMAND ERROR**

**Explanation:** The request was terminated because the RESYNC was not done between IMS and CQS.

**Programmer Response:** Reissue the request after the RESYNC was done.

**X'C000' QMR20 - SELECT SHOULD NOT OCCUR ERROR**

**Explanation:** The selection criteria routine (DFSQMR20) was called and detected include or exclude processing was to be performed but did not find any rows (INCL/EXCL) to process.

**Programmer Response:** Error is detected at label MRSEL500 in DFSQMR20. Select routine reached this

Table 219. DFSSQRY Return Codes

| Shared Queues Message Return Code | Decimal | Meaning                            |
|-----------------------------------|---------|------------------------------------|
| SQRRRC_OK                         | 0       | CALL SUCCESSFUL                    |
| SQRRRC_SOME                       | 4       | SUCCESSFUL ONLY FOR SOME RESOURCES |
| SQRRRC_NONE                       | 8       | SUCCESSFUL FOR NO RESOURCES        |
| SQRRRC_CQS_NOT_AVAIL              | 12      | CQS IS NOT AVAILABLE               |
| SQRRRC_IMS_STG_ERR                | 16      | IMS STORAGE ERROR                  |
| SQRRRC_CQS_ERR                    | 20      | CALL UNSUCCESSFUL - CQS ERROR      |
| SQRRRC_IMS_ERR                    | 24      | CALL UNSUCCESSFUL - IMS ERROR      |
| SQRRRC_QTP_NOMSG                  | 28      | NO MSGS FOR QTYPE                  |

**X'C008' QMRA0 - SELECT CRITERIA DFSPPOOL ERROR**

**Explanation:** The selection criteria routine (DFSQMRA0) was called by either BROWSE, QUERY, RECOVER, or UNLOAD, and issued a DFSPPOOL call to free storage. DFSQMRA0 received a nonzero return code.

**Programmer Response:** REG15 in the REG0-15 save area contains the DFSPPOOL return code. REG3 is the address of the storage being freed, and REG4 is the address of the parmlist passed to DFSPPOOL. This storage being freed is the DFSSQRY buffer obtained on a DFSSQRY call. This is an IMS internal error.

routine and the INCL/EXCL register (R6) was zero. R6 should be the address of one of the INCL/EXCL rows mapped by DFSMRQCT. R7 is the select work area (DFSQRQSW) that anchors the rows. MRINCTTR in DFSMRQSW is the number of include entries and MREXCCTR is the number of exclude entries. Flag SMRQ0INC=1 in MRQWORK (R5) means the include chain is being processed, if 0, the exclude chain is being processed. This is a IMS/QCF This is an internal error.

**X'C004' QMRA0 - SELECT CRITERIA DFSSQRY ERROR**

**Explanation:** The selection criteria routine (DFSQMRA0) was called by either BROWSE, QUERY, RECOVER, or UNLOAD, and issued a DFSSQRY call to query the shared queues to determine which queue names have messages. DFSSQRY returned a nonzero return code.

**Programmer Response:** REG1 in the REG0-15 save area, and the CMDCQSRC field in the MRQWORK (macro DFSMRQWK) contain the DFSSQRY return code. REG4 is the address of the parameter list passed to DFSSQRY. DFSSQRY return codes are also listed in the DFSSQRY macro. Table 219 lists the DFSSQRY return codes:

**X'C00C' QMRA0 - INVALID CMD CALL**

**Explanation:** DFSQMR20 or DFSQMRA0 was called with an invalid command call.

**Programmer Response:** Check QMRWFLG0 in DFSMRQWK for the command in progress. It should be BROWSE, QUERY, RECOVER, or UNLOAD.

**X'C010' QMRA0 - DFSCBTS SCAN/FIND ERROR**

**Explanation:** DFSCBTS call resulted in a return code greater than 4.

**Programmer Response:** If the return code is 8, then

| the error is due to CBTE not found. Otherwise, if the  
| return code is 12, then the error is because no scan  
| routine exists (set it DFSCBT40).

---

| **X'D004' QMR70 - LOAD/INSERT - INVALID  
| CALL TYPE REC**

| **Explanation:** Invalid call to DFSQMR70. REG0 did not  
| contain either X'04' (INSERT), X'08' (PURG), X'0C'  
| (CANCEL), X'1C' (REROUTE) X'24' (REROUTE  
| PURG), X'28' (CLEANUP), OR X'34' (COMAND).

| **Programmer Response:** REG1 in the REG0-REG15  
| save area contains the call type. Trace the problem to  
| the caller of DFSQMR70. Function being processed  
| should be either a load or insert and call should be one  
| of the above.

---

| **X'D008' QMR70 - LOAD - ERROR CANCELING  
| MESSAGE**

| **Explanation:** At cleanup time when the MRQ BMP  
| ended, the load/insert routine detected a message or  
| partial message that had not been enqueued and  
| attempted to cancel it. A nonzero return code was  
| returned on the QMGR cancel call. Because the BMP  
| was ending, this AIBREASN code will not be returned to  
| the MRQ BMP.

| **Programmer Response:** REG15 in the REG0-REG15  
| save area contains the QMGR return code from the  
| cancel. REG2 contains the QTPPCB used for the  
| cancel call. The nonzero cancel return code is a  
| MRQ/QMGR This is an internal error. The MRQ BMP  
| terminating with a insert message in progress may be a  
| MRQ BMP This is an internal error.

---

| **X'D00C' QMR70 - XFER - ERROR  
| TRANSFERING MESSAGE**

| **Explanation:** While processing a RESET or  
| CLEANUP command from the QCF BMP, a Transfer call  
| was issued to transfer inserted messages from the  
| temporary to the permanent destination. A nonzero  
| return code was returned on the XFER call.

| **Programmer Response:** REG15 in the REG0-REG15  
| save area contains the QMGR return code from the  
| XFER. REG2 contains the QTPPCB used for the XFER  
| call. The nonzero XFER return code is a QCF or QMGR  
| This is an internal error.

---

| **X'E000' DFSQMR00 - QSN exit started too  
| many BMPs**

| **Explanation:** The QCF QSN exit (DFSQMRI0) started  
| more than one concurrent BMP.

| **Programmer Response:** The QSN exit routine  
| DFSQMRI0 detected the message queue threshold was  
| reached and started a BMP to process the message  
| queue, however, a previous started BMP had not yet  
| completed. The BMP that is processing the lowest

| threshold (for example: threshold A to B, or B to C, or C  
| to D) is returned this AIBREASN code to cause it to  
| terminate. To eliminate message queue thrashing, only  
| one BMP processing the highest threshold is allowed to  
| run. This is not considered an error unless the lower  
| threshold BMP (for example: the one that receives this  
| code) is stalled for some reason and isn't completing.  
| Note if the BMP that scheduled for a higher threshold  
| completes OK. R5 is the address of MRQWORK and  
| MRQWORK flag QMRWFL02 = QM2BMPAB or  
| QM2BMPBC, or QM2BMPCD indicates which threshold  
| exceeded condition this BMP was processing.

---

| **X'E004' DFSQMRD0 - QC/QSN INVALID CMD  
| CALL**

| **Explanation:** DFSQMRD0 was called with an invalid  
| command call.

| **Programmer Response:** Check QMRWFL00 in  
| DFSMRQWK for the command in progress. It should be  
| QC-ABE, QC-REL, QC-SND or QC-SUS. This is an IMS  
| or QCF internal error.

---

| **X'E008' DFSQMRD0 - QC/QSN SUPPORTED  
| ONLY IN QCF ENVIRONMENT**

| **Explanation:** DFSQMRD0 was called in an non QCF  
| environment.

| **Programmer Response:** This is an IMS or QCF  
| internal error.

---

| **X'E00C' DFSQMRD0 - QC/QSN CMD CALL NO  
| QSN BLOCK**

| **Explanation:** DFSQMRD0 was called with an  
| QC/QSN CMD CALL but no QSN block.

| **Programmer Response:** This is an IMS or QCF  
| internal error.

---

| **X'E010' DFSQMRD0 - QC/QSN COMMAND  
| ACTION INVALID**

| **Explanation:** QC/QSN command action is invalid.

| **Programmer Response:** Check QCMRQFL0 in  
| DFSMRQWK for the command in progress. It should be  
| QC-ABE, QC-REL, QC-SND or QC-SUS. This is an IMS  
| or QCF This is an internal error.

---

| **X'E014' DFSQMRD0 - QC/QSN CMD CALL  
| INVALID ITASK**

| **Explanation:** QC/QSN command call ITASK type is  
| invalid.

| **Programmer Response:** Check MRPTASK1 and  
| MRPTASK2 in DFSMRQPF for the valid ITASK types.  
| This is an IMS or QCF internal error.

---

**X'E018' DFSQMRD0 - ERROR GET/REL AN AWE**

**Explanation:** DFSBCB GET or REL for an AWE block received a nonzero return code.

**Programmer Response:** R15 = return code from DFBCB GET call. This is either an internal error or insufficient storage available in the control region PRIVATE AREA.

---

**X'E01C' DFSQMRD0 - QC/QSN CMD CALL RECEIVED ERROR CODE FROM SELECT**

**Explanation:** DFSQMRA0 returned with RC = X'08' in QSQN selection call.

**Programmer Response:** Trace back to DFSQMRA0 to determine the cause.

---

**X'E020' QMRG0 - INVALID FUNCTION PASSED TO QC LOAD AND QUERY QUEUE SPACE NOTIFICATION TABLE**

**Explanation:** Invalid call to DFSQMRG0. Register 0 did not contain either X'28' (MRQCLEAN), X'34' (MRQCMD), or X'38' (MRQGCMD).

**Programmer Response:** REG0 in REG0-REG15 area contains the function code. REG14 is the address of the caller of DFSQMRG0 (BALR REG). Trace the call back to the caller of DFSQMRG0.

---

**X'E024' DFSQMRG0 - QC LOAD CMD CALL RECEIVED ERROR GET/REL DFSPPOOL STORAGE SERVICES**

**Explanation:** DFSPPOOL call received a nonzero return code attempting to get or release storage from the HIOP STORAGE POOL for a DFSMRQTB work area.

**Programmer Response:** R15 = return code from DFSPPOOL call. This is either an internal error or insufficient storage available in the control region private area.

---

**X'E028' DFSQMRG0 - QC LOAD CMD PROCESSING - the value for QUOTNOTF is invalid**

**Explanation:** The percentage for QUOFNOTF passed to IMS on a QC LOAD CMD call was invalid.

**Programmer Response:** This is either an internal QCF or IMS error.

---

**X'E02C' DFSQMRG0 - INVALID CMD CALL RECEIVED, ONLY /QC-LTBL IS CURRENTLY SUPPORTED**

**Explanation:** Invalid call to DFSQMRG0. The flag(QCMRQFL0) in DFSMRQWK did not indicate that the CMD was /QC-LTBL.

**Programmer Response:** Check QCMRQFL0 in DFSMRQWK for the command in progress. It should be QC-LTBL. This is an IMS or QCF internal error.

---

**X'E030' DFSQMRG0 - INVALID QUEUE UPPER AND/OR LOWER THRESHOLD PERCENT**

**Explanation:** Invalid queue upper, lower, or both, threshold percents were detected. DFSMRQTB work table was constructed using default values of upper(75%) lower(60%).

**Programmer Response:** Check QUOFQTU and QUOTQTL in DFSMRQO for the command in progress.

---

**X'E034' DFSQMRG0 - ERROR GET/REL AN AWE**

**Explanation:** DFSBCB GET or REL for an AWE block received a nonzero return code.

**Programmer Response:** R15 = return code from DFBCB GET call. This is either an internal error or insufficient storage available in the control region private area.

---

**X'E038' DFSQMRG0 - /QC-LTBL AND /QC-QTBL NOT ACTIVE**

**Explanation:** The required function is not active for the current active IMS system.

**Programmer Response:** If the function is required the module IQCQMRH0 must be linked into the IMS reslib (replacing IMS module DFSQMRH0) or linked into a user reslib as DFSQMRH0. The function is not currently supported for shared queues environment.

---

**X'E03C' DFSQMRD0 - QC/QSN COMMAND IS INVALID**

**Explanation:** The required function is not active or not supported for the current active IMS system.

**Programmer Response:** If the function is required, the module IQCQMRH0 must be linked into THE IMS RESLIB (replacing IMS module DFSQMRH0) or linked into a user RESLIB as DFSQMRH0. The function is not currently supported for shared queues environment.

---

---

**X'F000'**

**Explanation:** Reserved for future use.

**Programmer Response:**

---

**X'F004'**      **QMRC0 - INVALID FUNCTION PASSED TO ENVIRONMENT STATISTICS ROUTINE**

**Explanation:** Invalid call to DFSQMRC0. Register 0 did not contain either X'28' (MRQCLEAN), X'34' (MRQCMD), or X'38' (MRQGCMD).

**Programmer Response:** REG0 in REG0-REG15 area contains the function code. REG14 is the address of the caller of DFSQMRC0 (BALR REG). Trace the call back to the caller of DFSQMRC0.

---

**X'F008'**      **QMRC0 - IMS IS IN THE PROCESS OF SHUTDOWN OR QUIESCING**

**Explanation:** IMS is in the process of shutdown or quiescing and the CQS QUERY command is not allowed at this time.

**Programmer Response:** This is an IMS information AIBREASN code.

---

**X'F00C'**      **QMRC0 - SHARED QUEUES ENVIRONMENT, NO SHARED QUEUES MASTER CONTROL BLOCK (SCDSQM)**

**Explanation:** IMS is running in a shared queues environment, the pointer to the shared queues MASTER CONTROL BLOCK is zero.

**Programmer Response:** This is an IMS internal error.

---

**X'F010'**      **QMRC0 - SHARED QUEUES ENVIRONMENT, NO STRUCTURE BLOCK (SQMSQSM)**

**Explanation:** IMS is running in a shared queues environment. The pointer to the shared queues structure block is zero.

**Programmer Response:** This is an IMS internal error.

---

**X'F014'**      **QMRC0 - IMS This is an internal error**

**Explanation:** DFSSQI30 has returned an unsupported return code.

**Programmer Response:** This is an IMS internal error.

---

**X'F018'**      **QMRC0 - CQS NOT AVAILABLE TO PROCESS THE CQS QUERY REQUEST**

**Explanation:** CQS is not available to process the CQS QUERY request.

**Programmer Response:** This is an IMS information AIBREASN code.

---

**X'F01C'**      **QMRC0 - CQS RETURNED AN UNSUCCESSFUL RETURN CODE FOR THE CQS QUERY REQUEST**

**Explanation:** CQS returned an unsuccessful return code on the CQS QUERY request. Refer to CQSRRQRY for return codes.

**Programmer Response:** This is an IMS internal error.

---

**X'F020'**      **QMRC0 - IN PROCESSING THE QCF ENVIRONMENT STATISTICS REQUEST STORAGE WAS NOT OBTAINED**

**Explanation:** DFSPPOOL call received a nonzero return code attempting to get or release storage from the HIOP storage pool for a work area or buffer.

**Programmer Response:** R15 = return code from DFSPPOOL call. This is either an internal error or insufficient storage available in the control region private area.

---

**X'F024'**      **QMRC0 - IN PROCESSING THE QCF ENVIRONMENT STATISTICS THE LIST PASSED TO CQS CONTAINED AN INVALID STRUCTURE NAME**

**Explanation:** The DFSSQQRY list contained an invalid CQS structure name.

**Programmer Response:** This is an internal IMS problem.

---

**X'F028'**      **QMRC0 - IN PROCESSING THE QCF ENVIRONMENT STATISTICS REQUEST THE DFSSQQRY RETURNED A NON-ZERO RETURN CODE**

**Explanation:** The DFSSQQRY returned a nonzero return code.

**Programmer Response:** This is an internal IMS problem.

## Appendix D. 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.

In this section:

- “Load List Areas”
- “Control Block Table (CBT) Pools” on page 583

### Load List Areas

Table 220 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 220. Load List Areas

| Load List Name | IMS Block/Work Area                                                             | Pool Type           |
|----------------|---------------------------------------------------------------------------------|---------------------|
| DFSABSxx       | Abend Diagnostic Area, xx=PST number                                            | Global              |
| DFSBFSP        | 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 <sup>1</sup> |
| DFSCBT10       | Storage pool headers for the pools defined in DFSCBT00                          | Global <sup>1</sup> |
| 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 <sup>2</sup> |
| 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 <sup>4</sup>  |
| DFSMFDDP       | MFS Pool Dynamic Directory Prime Area                                           | Local <sup>4</sup>  |
| DFSMFDD0       | MFS Pool Dynamic Directory Entry Area                                           | Local <sup>4</sup>  |
| DFSMFPDS       | MFS Pool PDS Directory Indexes                                                  | Local <sup>4</sup>  |
| DFSMFSTG       | MFS Pool Staging Buffers                                                        | Local <sup>4</sup>  |

Table 220. Load List Areas (continued)

| Load List Name | IMS Block/Work Area                                    | Pool Type           |
|----------------|--------------------------------------------------------|---------------------|
| DFSMTCLB       | CLB (ECB) for DFSCMTIO                                 | Global              |
| DFSMTIOT       | Monitor TIOT Table                                     | Global              |
| DFSMTMH        | MSC Main Storage-to-Main Storage Queue Header          | Local <sup>3</sup>  |
| DFSMTMW        | MSC Main Storage-to-Main Storage Window                | Local <sup>3</sup>  |
| DFSOFPL        | OSAM Buffer Pool                                       | Global <sup>2</sup> |
| DFSOFWA        | 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 <sup>2</sup> |
| 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: SD SG                            | 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 <sup>3</sup>  |
| 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               |



Table 220. Load List Areas (continued)

| Load List Name | IMS Block/Work Area                                   | Pool Type           |
|----------------|-------------------------------------------------------|---------------------|
| DFSTSAV        | Temporary Save Sets                                   | Local               |
| DFSVRFXL       | Fixlist for EXCPVR                                    | Local               |
| DFSXCWxx       | Exclusive Control Enqueue/Dequeue Work Area, xx=01-99 | Global <sup>2</sup> |
| DFSZIBxx       | ZIB/FAQE Pool, xx=01-99                               | Global              |

**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 Table 221 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 221. 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)           |

Table 221. CBT Pool Names and Descriptions (continued)

| CBT Pool | Description                                                                                                                                                                         |
|----------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| DPST     | Dependent region PST: The following blocks are associated with the dependent region structure:<br>D1WA, DG2W, EPST, FSRB, GQMW, IDT, IOSB, IRLM, KLSO, 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                                                                                                                                                          |
| 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                                                                                                                                             |
| KLSO     | 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                                                                                                                                                                 |

Table 221. CBT Pool Names and Descriptions (continued)

| <b>CBT Pool</b> | <b>Description</b>                                                           |
|-----------------|------------------------------------------------------------------------------|
| 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                                              |
| 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                                                 |
| VTGB            | VTAM terminal control blocks                                                 |
| VWA             | Volatile work area                                                           |
| XMCI            | Cross memory ITASK block                                                     |
| XPST            | Dependent region PST extension                                               |
| X124            | DL/I pool below the 16MB line for MVS/ESA                                    |



## Appendix E. Acronyms and Abbreviations Used in This Section

For a complete listing of acronyms and abbreviations in the IMS library, see the *IMS Version 9: Master Index and Glossary*.

|              |                                     |
|--------------|-------------------------------------|
| <b>ACB</b>   | access method control block         |
| <b>AIB</b>   | application interface block         |
| <b>AMP</b>   | access method prefix block          |
| <b>APAR</b>  | authorized program analysis report  |
| <b>BMP</b>   | batch message processing            |
| <b>BSAM</b>  | Basic Sequential Access Method      |
| <b>CBT</b>   | control block table                 |
| <b>CCB</b>   | conversational control block        |
| <b>CCTL</b>  | coordinator controller              |
| <b>CDE</b>   | contents directory element          |
| <b>CIB</b>   | communication interface block       |
| <b>CICS</b>  | Customer Information Control System |
| <b>CLB</b>   | communication line block            |
| <b>CNT</b>   | communication name table            |
| <b>CQS</b>   | Common Queue Server                 |
| <b>CRB</b>   | communication restart block         |
| <b>CTB</b>   | communication terminal block        |
| <b>CTL</b>   | control                             |
| <b>CTRL</b>  | IMS control region                  |
| <b>CTT</b>   | communication translate table       |
| <b>DBCTL</b> | Database Control                    |
| <b>DBRC</b>  | Database Recovery Control           |
| <b>DB</b>    | Database function                   |
| <b>DCB</b>   | data control block                  |
| <b>DC</b>    | data communication function         |
| <b>DDIR</b>  | data management block directory     |
| <b>DEDB</b>  | Data entry database                 |
| <b>DMAC</b>  | data management area control block  |
| <b>DMB</b>   | data management block               |
| <b>DMCB</b>  | data management control block       |
| <b>DRA</b>   | Database Resource Adapter           |
| <b>DSG</b>   | data set group                      |
| <b>DSP</b>   | IMS dispatcher                      |

|               |                                        |
|---------------|----------------------------------------|
| <b>DSPWRK</b> | IMS dispatcher work area               |
| <b>ECB</b>    | event control block                    |
| <b>ECNT</b>   | extended communication name table      |
| <b>EEVT</b>   | external entry vector table            |
| <b>EEVTP</b>  | external entry vector table prefix     |
| <b>EPST</b>   | extended partition specification table |
| <b>ES</b>     | extended security support              |
| <b>ESCD</b>   | extended system contents directory     |
| <b>ESETP</b>  | external subsystem entry table prefix  |
| <b>ESS</b>    | external subsystem                     |
| <b>EWS</b>    | Early Warning System                   |
| <b>EZS</b>    | external connection status element     |
| <b>FP</b>     | Fast Path                              |
| <b>FTSC</b>   | Field Technical Support Center         |
| <b>GESE</b>   | global external subsystem entry        |
| <b>HALDB</b>  | High Availability Large Database       |
| <b>ID</b>     | identification                         |
| <b>ILS</b>    | isolated log send                      |
| <b>IMS</b>    | Information Management System          |
| <b>I/O</b>    | input/output                           |
| <b>IOB</b>    | input/output block                     |
| <b>IRLM</b>   | Internal Resource Lock Manager         |
| <b>ISC</b>    | Intersystem Communication              |
| <b>ISI</b>    | resource access security               |
| <b>ISL</b>    | IRLM identified subsystem list         |
| <b>ITASK</b>  | IMS task                               |
| <b>IWALE</b>  | internal work area list elements       |
| <b>IXRF</b>   | IMS-related XRF complex                |
| <b>LCB</b>    | link control block                     |
| <b>LCRE</b>   | local current recovery entry           |
| <b>LESE</b>   | local external subsystem entry         |
| <b>LLB</b>    | logical link block                     |
| <b>LNB</b>    | logical link name block                |
| <b>LTERM</b>  | logical terminal                       |
| <b>MFS</b>    | Message Format Service                 |
| <b>MNOTE</b>  | macro note                             |
| <b>MPP</b>    | message processing program             |

|              |                                     |
|--------------|-------------------------------------|
| <b>MRMB</b>  | randomizing module block            |
| <b>MRQ</b>   | Message Requeuer                    |
| <b>MSC</b>   | Multiple Systems Coupling           |
| <b>MSDB</b>  | main storage database               |
| <b>MVS</b>   | Multiple Virtual System             |
| <b>NM</b>    | notify message                      |
| <b>OLR</b>   | Online Reorganization               |
| <b>OM</b>    | Operations Manager                  |
| <b>OSAM</b>  | Overflow Sequential Access Method   |
| <b>PCB</b>   | program communication block         |
| <b>PCIB</b>  | Partition CIB                       |
| <b>PDIR</b>  | PSB directory                       |
| <b>PSB</b>   | program specification block         |
| <b>PST</b>   | partition specification table       |
| <b>PSW</b>   | program status word                 |
| <b>PTERM</b> | physical terminal                   |
| <b>PTF</b>   | program temporary fix               |
| <b>QCB</b>   | queue control block                 |
| <b>QSAM</b>  | Queued Sequential Access Method     |
| <b>RCTE</b>  | routing code table entry            |
| <b>RHB</b>   | IRLM resource header block          |
| <b>RLB</b>   | IRLM request lock block             |
| <b>RLMCB</b> | IRLM master control block           |
| <b>RPL</b>   | request parameter list              |
| <b>RRE</b>   | residual recovery element           |
| <b>RSR</b>   | Remote Site Recovery                |
| <b>SAP</b>   | save area prefix                    |
| <b>SB</b>    | sequential buffering                |
| <b>SCD</b>   | system contents directory           |
| <b>SCP</b>   | system control program              |
| <b>SE</b>    | system engineer                     |
| <b>SMB</b>   | scheduler message block             |
| <b>SMP</b>   | System Modification Program         |
| <b>SPA</b>   | scratch pad area                    |
| <b>SQ</b>    | shared queues                       |
| <b>SSCD</b>  | secondary system contents directory |
| <b>SSF</b>   | Software Support Facility           |

|              |                                   |
|--------------|-----------------------------------|
| <b>SSIE</b>  | subsystem status index block      |
| <b>SSQ</b>   | schedule sequence queue           |
| <b>SUR</b>   | Database Surveyor utility feature |
| <b>SYS</b>   | systems                           |
| <b>TCB</b>   | task control block                |
| <b>TCT</b>   | transaction class table           |
| <b>TKO</b>   | takeover                          |
| <b>TPPCB</b> | telecommunication program PCB     |
| <b>TRK</b>   | tracking                          |
| <b>UTIL</b>  | utility                           |
| <b>VTCB</b>  | VTAM terminal control block       |
| <b>XRF</b>   | Extended Recovery Facility        |



## Appendix F. Fast Path Trace Entries

Table 222 shows the Fast Path trace entries. For more information, see Chapter 11, “FP—Fast Path Service Aids,” on page 415.

Table 222. Fast Path Trace Entries

| Trace ID | Module   | Trace Point                       | Comments                     |
|----------|----------|-----------------------------------|------------------------------|
| ALOC     | DBFALOC0 | Entry                             | FP trace data set allocation |
| ALOX     | DBFALOC0 | Exit                              | FP trace data set allocation |
| RTYE     | DBFALOC0 | EMHB Present                      | FP trace data set allocation |
| RTYS     | DBFALOC0 | No EMHB Present                   | FP trace data set allocation |
| ALOP     | DBFALOC0 | Trace data set already allocated  | FP trace data set allocation |
| BBIN     | DBFBBIN0 | Entry, Exit (Shift)               | MSDB Binary Search           |
| OFSE     | DBFBBIN0 | Binary Search                     | MSDB Binary Search           |
| BIN1     | DBFBBIN0 | Binary Search Entry, Exit (Shift) | MSDB Binary Search           |
| BCHG     | DBFBCHG0 | Entry                             | MSDB FLD Call Change         |
| RCHG     | DBFBCHG0 | Exit                              | MSDB FLD Call Change         |
| BOFL     | DBFBCHG0 | Overflow                          | MSDB FLD Call Change         |
| BCL0     | DBFBCL10 | Entry                             | MSDB Call Handler            |
| RCL0     | DBFBCL10 | Exit                              | MSDB Call Handler            |
| IRC2     | DBFBCL10 | Copy Call Data                    | MSDB Call Handler            |
| BDLT     | DBFBDLT0 | Entry                             | MSDB Delete Call             |
| CDLT     | DBFBDLT0 | Delete OK                         | MSDB Delete Call             |
| RDLT     | DBFBDLT0 | Exit                              | MSDB Delete Call             |
| BENQ     | DBFBENQ0 | Entry                             | MSDB Resource Locking        |
| NQ16     | DBFBENQ0 | Function 16 = Enqueue             | MSDB Resource Locking        |
| ENQ1     | DBFBENQ0 | Resource Locked, call Lock Manger | MSDB Resource Locking        |
| ENQ2     | DBFBENQ0 | Resource Locked                   | MSDB Resource Locking        |
| RENQ     | DBFBENQ0 | Exit                              | MSDB Resource Locking        |
| BDEQ     | DBFBENQ0 | Dequeue                           | MSDB Resource Locking        |
| BFLD     | DBFBFLD0 | Entry                             | MSDB FLD Call Processor      |
| RFLD     | DBFBFLD0 | Exit                              | MSDB FLD Call Processor      |
| BGET     | DBFBGET0 | Entry                             | MSDB Get Processor           |
| RGET     | DBFBGET0 | Exit                              | MSDB Get Processor           |
| BINC     | DBFBINC0 | Entry, Exit (Shift)               | MSDB Decimal Field Verify    |
| BNXT     | DBFBNXT0 | Entry                             | MSDB Get Next                |
| RNXT     | DBFBNXT0 | Exit                              | MSDB Get Next                |
| BRPL     | DBFBRPL0 | Entry                             | MSDB Replace                 |
| RRPL     | DBFBRPL0 | Exit                              | MSDB Replace                 |
| BSEQ     | DBFBSEQ0 | Entry, Exit (Shift)               | MSDB Sequential Search       |

Table 222. Fast Path Trace Entries (continued)

| Trace ID | Module   | Trace Point                          | Comments                                     |
|----------|----------|--------------------------------------|----------------------------------------------|
| SEQ1     | DBFBSEQ0 | ECNT Search Entry,Exit (Shift)       | MSDB Sequential Search                       |
| SEQ1     | DBFBSEQ0 | ECNT Scan                            | MSDB Sequential Search                       |
| SEQ2     | DBFBSEQ0 | Segment Search Entry, Exit (Shift)   | MSDB Sequential Search                       |
| SEQ3     | DBFBSEQ0 | Search Forward Entry, Exit (Shift)   | MSDB Sequential Search                       |
| BSRT     | DBFBSRT0 | Entry                                | MSDB Insert Processor                        |
| CSRT     | DBFBSRT0 | Count Free Segments                  | MSDB Insert Processor                        |
| DSRT     | DBFBSRT0 | Insert Complete                      | MSDB Insert Processor                        |
| RSRT     | DBFBSRT0 | Exit                                 | MSDB Insert Processor                        |
| BUPB     | DBFBUPB0 | Entry                                | MSDB Update Buffer Space Handler             |
| RUPB     | DBFBUPB0 | Exit                                 | MSDB Update Buffer Space Handler             |
| BVAL     | DBFBVAL0 | Entry,Exit (Shift)                   | MSDB Decimal Segment Validate                |
| BVfy     | DBFBVfy0 | Entry                                | MSDB Field Verify Processor                  |
| RVfy     | DBFBVfy0 | Exit                                 | MSDB Field Verify Processor                  |
| BXTR     | DBFBXTR0 | Entry,Exit (Shift)                   | MSDB Hex Field Translator                    |
| CBHL     | DBFCBHL0 | Entry                                | DEDB Hard Luck Buffer Handler (Buffer Steal) |
| YBHL     | DBFCBHL0 | Exit to caller or to wait for buffer | DEDB Hard Luck Buffer Handler (Buffer Steal) |
| ZBHL     | DBFCBHL0 | OBA required                         | DEDB Hard Luck Buffer Handler (Buffer Steal) |
| BDU0     | DBFDBDU0 | Entry                                | MSDB Log Update Processor                    |
| CHGA     | DBFDBDU0 | Change - Before                      | MSDB Log Update Processor                    |
| CHGB     | DBFDBDU0 | Change - After                       | MSDB Log Update Processor                    |
| DECA     | DBFDBDU0 | Decimal Operation - Before           | MSDB Log Update Processor                    |
| DECB     | DBFDBDU0 | Decimal Operation - After            | MSDB Log Update Processor                    |
| DLTA     | DBFDBDU0 | Delete - Before                      | MSDB Log Update Processor                    |
| DLBT     | DBFDBDU0 | Delete - After                       | MSDB Log Update Processor                    |
| SRTA     | DBFDBDU0 | Insert - Before                      | MSDB Log Update Processor                    |
| SRTB     | DBFDBDU0 | Insert - After                       | MSDB Log Update Processor                    |
| RDU0     | DBFDBDU0 | Exit                                 | MSDB Log Update Processor                    |
| DCAP     | DBFDCAP0 | Entry                                | DEDB Change Data Capture                     |
| CAPD     | DBFDCAP0 | Build CAPD Block                     | DEDB Change Data Capture                     |
| DATA     | DBFDCAP0 | Build CAPD_DATA Blocks               | DEDB Change Data Capture                     |
| READ     | DBFDCAP0 | Read DEDB CI                         | DEDB Change Data Capture                     |
| DCAx     | DBFDCAP0 | Should not occur, invalid call type  | DEDB Change Data Capture                     |
| SLG2     | DBFDLG20 | Good Sync                            | FP Resync Commit/Abort Log Processor         |
| SLOG     | DBFDLG20 | Bad Sync                             | FP Resync Commit/Abort Log Processor         |

Table 222. Fast Path Trace Entries (continued)

| Trace ID | Module   | Trace Point                          | Comments                                |
|----------|----------|--------------------------------------|-----------------------------------------|
| TLG2     | DBFDLG20 | Exit                                 | FP Resync Commit/Abort Log Processor    |
| DRSC     | DBFDRSC0 | Entry                                | FP Resync Controller                    |
| DSRP     | DBFDSRP0 | Entry                                | DEDB SDEP Resync Processor              |
| DSRN     | DBFDSRP0 | Exit                                 | DEDB SDEP Resync Processor              |
| HCHG     | DBFHCHG0 | Entry                                | EMH Alt PCB CHNG Call Processor         |
| NCHG     | DBFHCHG0 | Exit                                 | EMH Alt PCB CHNG Call Processor         |
| HCL0     | DBFHCL00 | Entry                                | EMH and FP Utility Call Analyzer        |
| NCL0     | DBFHCL00 | Exit                                 | EMH and FP Utility Call Analyzer        |
| HGN0     | DBFHGN00 | Entry                                | EMH Get Next Call Processor             |
| NGN0     | DBFHGN00 | Exit                                 | EMH Get Next Call Processor             |
| HGU1     | DBFHGU10 | Entry                                | EMH Get Unique + Sync Control Processor |
| NGU1     | DBFHGU10 | Exit                                 | EMH Get Unique + Sync Control Processor |
| EOTR     | DBFHGU10 | End of Thread                        | EMH Get Unique + Sync Control Processor |
| RTRY     | DBFHGU10 | Retried Transaction                  | EMH Get Unique + Sync Control Processor |
| BOTR     | DBFHGU10 | Start of Thread                      | EMH Get Unique + Sync Control Processor |
| HRLB     | DBFHRLB0 | Entry                                | EMH ROLB Processor                      |
| NRLB     | DBFHRLB0 | Exit                                 | EMH ROLB Processor                      |
| HSRT     | DBFHSRT0 | Entry                                | EMH TP PCB ISRT Processor               |
| NSRT     | DBFHSRT0 | Exit                                 | EMH TP PCB ISRT Processor               |
| FPR3     | DBFIRC10 | DL/I Call Start                      | FP Inter-Region Communication           |
| IRC1     | DBFIRC10 | DL/I Call                            | FP Inter-Region Communication           |
| IR09     | DBFIRC10 | Post Call, DEDB FLD,<br>or MSDB      | FP Inter-Region Communication           |
| OPMV     | DBFIRC10 | Post Call, Move Data to<br>Dependent | FP Inter-Region Communication           |
| IRCZ     | DBFIRC10 | Post Call, Pseudo<br>Abend Set       | FP Inter-Region Communication           |
| MBED     | DBFMBED0 | Entry                                | DEDB Get CI                             |
| MBE2     | DBFMBED0 | HSSP Async Read<br>Ahead Wait        | DEDB Get CI                             |
| MBEH     | DBFMBED0 | HSSP, found CI in<br>Private Buffer  | DEDB Get CI                             |
| GPRS     | DBFMBED0 | Exit without XCRB                    | DEDB Get CI                             |
| NBED     | DBFMBED0 | Exit                                 | DEDB Get CI                             |
| BFL9     | DBFMBFL9 | Entry                                | Build FLDC for ISRT                     |
| BFLX     | DBFMBFL9 | Exit                                 | Build FLDC for ISRT                     |
| MBMM     | DBFMBMM9 | Entry                                | Build SSAs, set Minimum   Maximum       |
| MB02     | DBFMBMM9 | GT Operator no<br>Minimum            | Build SSAs, set Minimum   Maximum       |
| MB03     | DBFMBMM9 | GT Operator Minimum                  | Build SSAs, set Minimum   Maximum       |
| MB04     | DBFMBMM9 | GE Operator no<br>Minimum            | Build SSAs, set Minimum   Maximum       |

Table 222. Fast Path Trace Entries (continued)

| Trace ID | Module   | Trace Point                     | Comments                          |
|----------|----------|---------------------------------|-----------------------------------|
| MB05     | DBFMBMM9 | GE Operator Minimum             | Build SSAs, set Minimum   Maximum |
| MB06     | DBFMBMM9 | LT Operator no Maximum          | Build SSAs, set Minimum   Maximum |
| MB07     | DBFMBMM9 | LT Operator Maximum             | Build SSAs, set Minimum   Maximum |
| MB08     | DBFMBMM9 | LE Operator no Maximum          | Build SSAs, set Minimum   Maximum |
| MB09     | DBFMBMM9 | Invalid Boolean Operator        | Build SSAs, set Minimum   Maximum |
| MB10     | DBFMBMM9 | EQ Operator set Maximum         | Build SSAs, set Minimum   Maximum |
| MB11     | DBFMBMM9 | EQ Operator Maximum already set | Build SSAs, set Minimum   Maximum |
| MB12     | DBFMBMM9 | Set Minimum                     | Build SSAs, set Minimum   Maximum |
| MB13     | DBFMBMM9 | Minimum already set             | Build SSAs, set Minimum   Maximum |
| MB14     | DBFMBMM9 | NE Operator                     | Build SSAs, set Minimum   Maximum |
| MB15     | DBFMBMM9 | No Key Fields                   | Build SSAs, set Minimum   Maximum |
| MB16     | DBFMBMM9 | Error in Maximum or Minimum     | Build SSAs, set Minimum   Maximum |
| MB17     | DBFMBMM9 | Set Maximum into SSA            | Build SSAs, set Minimum   Maximum |
| MB18     | DBFMBMM9 | Set Maximum into SSA            | Build SSAs, set Minimum   Maximum |
| MB19     | DBFMBMM9 | Set Maximum into SSA            | Build SSAs, set Minimum   Maximum |
| MB20     | DBFMBMM9 | Maximum zero                    | Build SSAs, set Minimum   Maximum |
| MB21     | DBFMBMM9 | Set Minimum into SSA            | Build SSAs, set Minimum   Maximum |
| MB22     | DBFMBMM9 | Set Minimum into SSA            | Build SSAs, set Minimum   Maximum |
| MB23     | DBFMBMM9 | Set Minimum into SSA            | Build SSAs, set Minimum   Maximum |
| MB25     | DBFMBMM9 | Set Minimum into SSA            | Build SSAs, set Minimum   Maximum |
| MB26     | DBFMBMM9 | Minimum zero                    | Build SSAs, set Minimum   Maximum |
| MBMM     | DBFMBMM9 | Exit                            | Build SSAs, set Minimum   Maximum |
| CVAL     | DBFMCCV9 | Entry, Exit (Shift)             | Check Command Code Validity       |
| SSP1     | DBFMCCV9 | Subset Pointer                  | Check Command Code Validity       |
| SSR1     | DBFMCCV9 | Command Code R                  | Check Command Code Validity       |
| SSP2     | DBFMCCV9 | Other Subset Command            | Check Command Code Validity       |
| SSP3     | DBFMCCV9 | Check for Conflicts             | Check Command Code Validity       |
| SSPX     | DBFMCCV9 | Check Subset Pointer Conflict   | Check Command Code Validity       |
| SSP4     | DBFMCCV9 | Set Pointers                    | Check Command Code Validity       |
| SSP5     | DBFMCCV9 | Not Command Code C              | Check Command Code Validity       |
| SSP6     | DBFMCCV9 | Command Code C                  | Check Command Code Validity       |
| SSP7     | DBFMCCV9 | Command Code F or R             | Check Command Code Validity       |
| SSP8     | DBFMCCV9 | F, R, or L not at ISRT Level    | Check Command Code Validity       |
| LOPP     | DBFMCCV9 | Loop checking position          | Check Command Code Validity       |

Table 222. Fast Path Trace Entries (continued)

| Trace ID | Module   | Trace Point                         | Comments                             |
|----------|----------|-------------------------------------|--------------------------------------|
| ELOP     | DBFMCCV9 | End of loop checking position       | Check Command Code Validity          |
| SSP9     | DBFMCCV9 | F, R, or L at ISRT level            | Check Command Code Validity          |
| SSPA     | DBFMCCV9 | Command Code U                      | Check Command Code Validity          |
| LOPU     | DBFMCCV9 | Loop checking position back to root | Check Command Code Validity          |
| SSPB     | DBFMCCV9 | Command Code V                      | Check Command Code Validity          |
| AMST     | DBFMCCV9 | Status Code AM set                  | Check Command Code Validity          |
| AJST     | DBFMCCV9 | Status Code AJ set                  | Check Command Code Validity          |
| MCHG     | DBFMCHG0 | Entry                               | DEDB FLD Call Processor              |
| XCHG     | DBFMCHG0 | Exit                                | DEDB FLD Call Processor              |
| MOFL     | DBFMCHG0 | Overflow                            | DEDB FLD Call Processor              |
| MCL0     | DBFMCLX0 | Entry                               | DEDB Call Analyzer                   |
| SSAX     | DBFMCLX0 | Count SSAs                          | DEDB Call Analyzer                   |
| SEG4     | DBFMCLX0 | Good RC, Trace Segment              | DEDB Call Analyzer                   |
| PARP     | DBFMCLX0 | Trace Parent                        | DEDB Call Analyzer                   |
| NCL0     | DBFMCLX0 | Exit                                | DEDB Call Analyzer                   |
| CRP9     | DBFMCRP9 | Entry                               | Check for Subset Pointer             |
| PPRE     | DBFMCRP9 | Trace Parent Prefix                 | Check for Subset Pointer             |
| CRPX     | DBFMCRP9 | Exit                                | Check for Subset Pointer             |
| MCSS     | DBFMCSS9 | Entry                               | Compare Current Segment Field to SSA |
| CALL     | DBFMCSS9 | Entry, Trace Call Argument          | Compare Current Segment Field to SSA |
| MC01     | DBFMCSS9 | Key SSA + Key Value                 | Compare Current Segment Field to SSA |
| MC02     | DBFMCSS9 | Compare Key                         | Compare Current Segment Field to SSA |
| MC03     | DBFMCSS9 | Compare Key                         | Compare Current Segment Field to SSA |
| MC04     | DBFMCSS9 | Compare Key                         | Compare Current Segment Field to SSA |
| MC05     | DBFMCSS9 | Not Satisfied RC=8                  | Compare Current Segment Field to SSA |
| MC06     | DBFMCSS9 | Compare Key                         | Compare Current Segment Field to SSA |
| MC07     | DBFMCSS9 | Compare Key                         | Compare Current Segment Field to SSA |
| MC08     | DBFMCSS9 | Not Satisfied RC=12                 | Not Satisfied RC=12                  |
| MC09     | DBFMCSS9 | Compare Key                         | Compare Current Segment Field to SSA |
| MC10     | DBFMCSS9 | Compare Key                         | Compare Current Segment Field to SSA |
| MC11     | DBFMCSS9 | Compare                             | Compare Current Segment Field to SSA |
| MC12     | DBFMCSS9 | Compare                             | Compare Current Segment Field to SSA |
| MC13     | DBFMCSS9 | Compare                             | Compare Current Segment Field to SSA |
| MC14     | DBFMCSS9 | Compare                             | Compare Current Segment Field to SSA |
| MC1A     | DBFMCSS9 | No Match                            | Compare Current Segment Field to SSA |
| MC1B     | DBFMCSS9 | Compare                             | Compare Current Segment Field to SSA |
| CSSF     | DBFMCSS9 | Compare, no Boolean                 | Compare Current Segment Field to SSA |

Table 222. Fast Path Trace Entries (continued)

| Trace ID | Module   | Trace Point                       | Comments                                        |
|----------|----------|-----------------------------------|-------------------------------------------------|
| MCS2     | DBFMCSS9 | Relational Operator Satisfied     | Compare Current Segment Field to SSA            |
| MCS1     | DBFMCSS9 | Relational Operator Not Satisfied | Compare Current Segment Field to SSA            |
| CSS9     | DBFMCSS9 | Exit                              | Compare Current Segment Field to SSA            |
| CSL9     | DBFMCSL9 | Entry,Exit (Shift)                | Compare Current Segment Field to SSA            |
| CALL     | DBFMCSL9 | Entry, Trace Call Argument        | Compare Current Segment Field to SSA            |
| LOPC     | DBFMCSL9 | Compare Loop                      | Compare Current Segment Field to SSA            |
| NEXT     | DBFMCSL9 | Read Next Buffer                  | Compare Current Segment Field to SSA            |
| MCTL     | DBFMCTL0 | Entry, Exit (Shift)               | Check this Level                                |
| COML     | DBFMCTL0 | Command Code L                    | Check this Level                                |
| FRST     | DBFMCTL0 | Command Code F, R, or unqualified | Check this Level                                |
| GETN     | DBFMCTL0 | Get Next                          | Check this Level                                |
| ISRT     | DBFMCTL0 | ISRT Here                         | Check this Level                                |
| MCT3     | DBFMCTL0 | Trace Process Return Code         | Check this Level                                |
| MDEQ     | DBFMDEQ0 | Entry                             | DEDB DEQ Command Processor                      |
| XDEQ     | DBFMDEQ0 | Exit                              | DEDB DEQ Command Processor                      |
| MDLT     | DBFMDLT0 | Entry                             | DEDB Direct Delete                              |
| EPCB     | DBFMDLT0 | Check for other PCBs              | DEDB Direct Delete                              |
| PRBA     | DBFMDLT0 | Update other PCB PRBA             | DEDB Direct Delete                              |
| CRBA     | DBFMDLT0 | Clear other PCB CRBA              | DEDB Direct Delete                              |
| KILL     | DBFMDLT0 | Reset Parentage other PCB         | DEDB Direct Delete                              |
| NRBA     | DBFMDLT0 | Update other PCB NRBA             | DEDB Direct Delete                              |
| XRBA     | DBFMDLT0 | Update other PCB XRBA             | DEDB Direct Delete                              |
| GRBA     | DBFMDLT0 | Update other PCB GRBA             | DEDB Direct Delete                              |
| DPTX     | DBFMDPT9 | Exit                              | Delete PCL and Subset Pointers in Parent Prefix |
| DPTX     | DBFMDPT9 | Exit                              | Delete PCL and Subset Pointers in Parent Prefix |
| MDRA     | DBFMDRA9 | Entry,Exit (Shift)                | Determine possibility of randomizing            |
| MD01     | DBFMDRA9 | Read First Root                   | Determine possibility of randomizing            |
| MD02     | DBFMDRA9 | Use Current Position              | Determine possibility of randomizing            |
| MD03     | DBFMDRA9 | Can't use CP, randomize           | Determine possibility of randomizing            |
| MD04     | DBFMDRA9 | Current key LT SSA min, randomize | Determine possibility of randomizing            |
| MD05     | DBFMDRA9 | Must Move this Level              | Determine possibility of randomizing            |

Table 222. Fast Path Trace Entries (continued)

| Trace ID | Module   | Trace Point                      | Comments                             |
|----------|----------|----------------------------------|--------------------------------------|
| MD06     | DBFMDRA9 | Current Key GT SSA min           | Determine possibility of randomizing |
| MD07     | DBFMDRA9 | Current Key GT SSA min           | Determine possibility of randomizing |
| MD08     | DBFMDRA9 | Current Key GT SSA max           | Determine possibility of randomizing |
| MD09     | DBFMDRA9 | Must Move this Level             | Determine possibility of randomizing |
| MD10     | DBFMDRA9 | Level Acceptable                 | Determine possibility of randomizing |
| MD11     | DBFMDRA9 | Current Key LT SSA max           | Determine possibility of randomizing |
| MD12     | DBFMDRA9 | Current Key GT SSA max           | Determine possibility of randomizing |
| MD13     | DBFMDRA9 | Current Key LT SSA max           | Determine possibility of randomizing |
| MD14     | DBFMDRA9 | Must Move this Level             | Determine possibility of randomizing |
| MD15     | DBFMDRA9 | Level Acceptable                 | Determine possibility of randomizing |
| MA13     | DBFMDRA9 | SSA Min = SSA Max, randomizer    | Determine possibility of randomizing |
| MD19     | DBFMDRA9 | Set SSA Min, Max                 | Determine possibility of randomizing |
| MD20     | DBFMDRA9 | No Low Boundary                  | Determine possibility of randomizing |
| MD21     | DBFMDRA9 | Current Key LT SSA Max           | Determine possibility of randomizing |
| MD22     | DBFMDRA9 | Must Move this Level             | Determine possibility of randomizing |
| MD23     | DBFMDRA9 | Do Nothing                       | Determine possibility of randomizing |
| MD24     | DBFMDRA9 | Current Key GT SSA Max           | Determine possibility of randomizing |
| MD26     | DBFMDRA9 | Do Sequential Read               | Determine possibility of randomizing |
| MD27     | DBFMDRA9 | Do Sequential Read               | Determine possibility of randomizing |
| MD28     | DBFMDRA9 | Do Nothing                       | Determine possibility of randomizing |
| MD29     | DBFMDRA9 | Current Key LT SSA Max           | Determine possibility of randomizing |
| MD31     | DBFMDRA9 | Current Key GT FDLC Low Key      | Determine possibility of randomizing |
| MD32     | DBFMDRA9 | Current Key GT FDLC Low Key      | Determine possibility of randomizing |
| MD33     | DBFMDRA9 | Do Nothing                       | Determine possibility of randomizing |
| MD34     | DBFMDRA9 | Do Sequential Read               | Determine possibility of randomizing |
| MD35     | DBFMDRA9 | Do Nothing                       | Determine possibility of randomizing |
| MD36     | DBFMDRA9 | Do Sequential Read               | Determine possibility of randomizing |
| MD37     | DBFMDRA9 | Do Nothing                       | Determine possibility of randomizing |
| MD38     | DBFMDRA9 | Go to Next Set Process           | Determine possibility of randomizing |
| MD41     | DBFMDRA9 | Search for Lowest Min GT Current | Determine possibility of randomizing |

Table 222. Fast Path Trace Entries (continued)

| Trace ID | Module   | Trace Point                        | Comments                             |
|----------|----------|------------------------------------|--------------------------------------|
| MD42     | DBFMDRA9 | Search for Next Higher Set Minimum | Determine possibility of randomizing |
| MD43     | DBFMDRA9 | Address next Set                   | Determine possibility of randomizing |
| MD44     | DBFMDRA9 | Set Min Found, Randomize           | Determine possibility of randomizing |
| MD45     | DBFMDRA9 | Goto NOUSE                         | Determine possibility of randomizing |
| MD46     | DBFMDRA9 | Read First Root                    | Determine possibility of randomizing |
| MD47     | DBFMDRA9 | Do Nothing - current position good | Determine possibility of randomizing |
| MD48     | DBFMDRA9 | Clear current position             | Determine possibility of randomizing |
| MD49     | DBFMDRA9 | Call Randomizer                    | Determine possibility of randomizing |
| MD50     | DBFMDRA9 | Continue Sequential Read           | Determine possibility of randomizing |
| MDRB     | DBFMDRBO | Entry                              | Delete, get root backwards           |
| MDRT     | DBFMDRX0 | Entry, Exit DDEP,SDEP (Shift)      | DEDB Insert                          |
| DSG9     | DBFMDSG9 | Entry                              | DEDB Delete Direct Dependent         |
| RECU     | DBFMDSG9 | Recursive Call Stack Information   | DEDB Delete Direct Dependent         |
| DSG1     | DBFMDSG9 | Trace Segment to be processed      | DEDB Delete Direct Dependent         |
| LOP1     | DBFMDSG9 | Twin Chain Loop                    | DEDB Delete Direct Dependent         |
| SIBL     | DBFMDSG9 | First Child                        | DEDB Delete Direct Dependent         |
| LO1X     | DBFMDSG9 | Loop over Parent Prefix Complete   | DEDB Delete Direct Dependent         |
| FRE1     | DBFMDSG9 | Call DBFMFSE0 to free space        | DEDB Delete Direct Dependent         |
| LOP2     | DBFMDSG9 | Twin Chain Loop                    | DEDB Delete Direct Dependent         |
| FRE2     | DBFMDSG9 | Call DBFMFSE0 to free space        | DEDB Delete Direct Dependent         |
| DSGX     | DBFMDSG9 | Exit                               | DEDB Delete Direct Dependent         |
| DS14     | DBFMDSG9 | Return after Recursive Call        | DEDB Delete Direct Dependent         |
| MFL0     | DBFMFL00 | Entry                              | DEDB FLD Call Processor              |
| XMFL     | DBFMFL00 | Exit                               | DEDB FLD Call Processor              |
| MFLD     | DBFMFL10 | Entry                              | DEDB FSA Processor                   |
| XFLD     | DBFMFL10 | Exit                               | DEDB FSA Processor                   |
| MFSE     | DBFMFSE0 | Entry                              | DEDB Space Manager                   |
| MFSS     | DBFMFSE0 | Scrap Handling                     | DEDB Space Manager                   |
| NFSE     | DBFMFSE0 | Exit                               | DEDB Space Manager                   |
| OFSE     | DBFMFSE0 | Read AP or Root CI to find space   | DEDB Space Manager                   |
| PFSE     | DBFMFSE0 | Read 1st DOVF CI                   | DEDB Space Manager                   |



Table 222. Fast Path Trace Entries (continued)

| Trace ID | Module   | Trace Point                         | Comments                                                               |
|----------|----------|-------------------------------------|------------------------------------------------------------------------|
| GPDS     | DBFMFSE0 | Got Conditional Lock IOVF SM CI     | DEDB Space Manager                                                     |
| GPDN     | DBFMFSE0 | Bad Conditional Lock IOVF SM CI     | DEDB Space Manager                                                     |
| MGAP     | DBFMGAP0 | Entry                               | DEDB Get Anchor Point                                                  |
| ERAN     | DBFMGAP0 | Entry to Randomizer                 | DEDB Get Anchor Point                                                  |
| XRAN     | DBFMGAP0 | Exit from Randomizer                | DEDB Get Anchor Point                                                  |
| MGA1     | DBFMGAP0 | PROCOPT=P UOW BDY crossed           | DEDB Get Anchor Point                                                  |
| MGA2     | DBFMGAP0 | PROCOPT=P UOW Set GC status         | DEDB Get Anchor Point                                                  |
| MGA3     | DBFMGAP0 | PROCOPT=H Save Position             | DEDB Get Anchor Point                                                  |
| NGAP     | DBFMGAP0 | Exit                                | DEDB Get Anchor Point                                                  |
| MGFD     | DBFMGFD0 | Entry                               | DEDB Initialize Level Table                                            |
| NGFD     | DBFMGFD0 | Exit                                | DEDB Initialize Level Table                                            |
| MGL9     | DBFMGLA9 | Entry, Exit (Shift)                 | DEDB Get Last Occurrence of Segment Under Parent                       |
| CLA9     | DBFMGLA9 | Enter, Exit(shift) EP DBFMCLA9      | DEDB Check if Another Occurrence of Segment under Parent satisfies SSA |
| MGNR     | DBFMGNR0 | Entry                               | DEDB Get Next Root                                                     |
| NGNR     | DBFMGNR0 | Exit                                | DEDB Get Next Root                                                     |
| EXAP     | DBFMGNR0 | Get Next RAP with a root start loop | DEDB Get Next Root                                                     |
| EXA1     | DBFMGNR0 | Get Next RAP with a root block#     | DEDB Get Next Root                                                     |
| EOC1     | DBFMGNR0 | Out of Area Range                   | DEDB Get Next Root                                                     |
| EXA2     | DBFMGNR0 | Read CI                             | DEDB Get Next Root                                                     |
| XXAP     | DBFMGNR0 | Return the RAP                      | DEDB Get Next Root                                                     |
| ECAL     | DBFMGNR0 | Calc UOW#, DMAC from BLK#           | DEDB Get Next Root                                                     |
| ECA1     | DBFMGNR0 | PROCOPT=P set GC status             | DEDB Get Next Root                                                     |
| ECA2     | DBFMGNR0 | PROCOPT=P set GC status             | DEDB Get Next Root                                                     |
| XCAL     | DBFMGNR0 | Rerun UOW#, DMAC                    | DEDB Get Next Root                                                     |
| MGN0     | DBFMGNX0 | Entry, Exit(Shift)                  | DEDB Get Next Root                                                     |
| GPDE     | DBFMGPD0 | Entry                               | DEDB Retrieve Sequential Dependent Segment                             |
| GPD1     | DBFMGPD0 | Scan Segment Chain                  | DEDB Retrieve Sequential Dependent Segment                             |
| GPD2     | DBFMGPD0 | SDEP Pointer -> uncommitted seg     | DEDB Retrieve Sequential Dependent Segment                             |
| GPD3     | DBFMGPD0 | SDEP Pointer -> normal seg          | DEDB Retrieve Sequential Dependent Segment                             |
| GPD4     | DBFMGPD0 | Must Read CI                        | DEDB Retrieve Sequential Dependent Segment                             |

Table 222. Fast Path Trace Entries (continued)

| Trace ID | Module   | Trace Point                            | Comments                                   |
|----------|----------|----------------------------------------|--------------------------------------------|
| GPDS     | DBFMGPD0 | IRLM Notify to Partner                 | DEDB Retrieve Sequential Dependent Segment |
| GDPN     | DBFMGPD0 | SDEP CI found to be locked             | DEDB Retrieve Sequential Dependent Segment |
| GDP7     | DBFMGPD0 | Re-read CI                             | DEDB Retrieve Sequential Dependent Segment |
| GDCP     | DBFMGPD0 | Compare Segment to SSA                 | DEDB Retrieve Sequential Dependent Segment |
| GPD5     | DBFMGPD0 | SSA does not match this segment        | DEDB Retrieve Sequential Dependent Segment |
| GPD6     | DBFMGPD0 | Copy segment                           | DEDB Retrieve Sequential Dependent Segment |
| GDPX     | DBFMGPD0 | Exit                                   | DEDB Retrieve Sequential Dependent Segment |
| MGPF     | DBFMGPF0 | Entry                                  | Get Page of Common Storage                 |
| NGPF     | DBFMGPF0 | Exit                                   | Get Page of Common Storage                 |
| MGRF     | DBFMGRF0 | Entry                                  | Get Root Forward Search                    |
| MGR1     | DBFMGRF0 | Run Chain in RAP CI                    | Get Root Forward Search                    |
| MGR2     | DBFMGRF0 | Scan RAP CI                            | Get Root Forward Search                    |
| MGR3     | DBFMGRF0 | Run Chain next CI                      | Get Root Forward Search                    |
| MGR4     | DBFMGRF0 | Scan next CI                           | Get Root Forward Search                    |
| MGR5     | DBFMGRF0 | Root does not exist (status GE)        | Get Root Forward Search                    |
| MGR6     | DBFMGRF0 | Root does not exist, other roots found | Get Root Forward Search                    |
| MGR8     | DBFMGRF0 | Root found during a scan               | Get Root Forward Search                    |
| MGR9     | DBFMGRF0 | Root found by Run Chain                | Get Root Forward Search                    |
| NGRF     | DBFMGRF0 | Exit                                   | Get Root Forward Search                    |
| MGRG     | DBFMGRF0 | Anchor Point Scan routine              | Get Root Forward Search                    |
| MGRL     | DBFMGRF0 | CI Scan routine                        | Get Root Forward Search                    |
| MGRM     | DBFMGRF0 | Nextitem routine                       | Get Root Forward Search                    |
| MGRC     | DBFMGRF0 | Call DBFMPGO0 check PROCOPT            | Get Root Forward Search                    |
| MGRD     | DBFMGRF0 | Return from DBFMPGO0                   | Get Root Forward Search                    |
| MGU0     | DBFMGUX0 | Entry, Exit(Shift)                     | Get Unique, Unqualified                    |
| MGXC     | DBFMGXC0 | Entry, Entry SEGLOCK                   | Get Control of Resource                    |
| NGXC     | DBFMGXC0 | Exit, Entry SEGLOCK, Exit CI EXCL      | Get Control of Resource                    |
| ENQR     | DBFMGXC0 | Lock Resource for this caller          | Get Control of Resource                    |
| ENQO     | DBFMGXC0 | Lock Resource on behalf of other       | Get Control of Resource                    |
| SHXC     | DBFMGXC0 | Just CI SHR Lock, Entry/Exit           | Get Control of Resource                    |

Table 222. Fast Path Trace Entries (continued)

| Trace ID | Module   | Trace Point                       | Comments                                       |
|----------|----------|-----------------------------------|------------------------------------------------|
| EXXC     | DBFMGXC0 | Just CI Lock Exclusive, Entry     | Get Control of Resource                        |
| VLOC     | DBFMGXC0 | DBFVLOCK Entry                    | Get Control of Resource                        |
| NLOC     | DBFMGXC0 | DBFVLOCK Exit                     | Get Control of Resource                        |
| MINC     | DBFMINC0 | Entry                             | DEDB Included Decimal FLD Call                 |
| XINC     | DBFMINC0 | Exit                              | DEDB Included Decimal FLD Call                 |
| IRC9     | DBFMIRC9 | Entry, Exit(Shift)                | DEDB Retrieve Previous Parent, Set MLTE Fields |
| MIRT     | DBFMIRT0 | Entry, Exit(Shift)                | DEDB Insert                                    |
| MIR1     | DBFMIRT0 | Trace CI RBA                      | DEDB Insert                                    |
| MIR2     | DBFMIRT0 | Previous Root Twin not in Same CI | DEDB Insert                                    |
| MIR3     | DBFMIRT0 | Trace after reading CI            | DEDB Insert                                    |
| MIR4     | DBFMIRT0 | Previous Segment not in Same CI   | DEDB Insert                                    |
| MIR5     | DBFMIRT0 | Previous Segment not Parent       | DEDB Insert                                    |
| MIRB     | DBFMIRT0 | Set PCF Pointer in Parent         | DEDB Insert                                    |
| MIR6     | DBFMIRT0 | Previous Segment in Same CI       | DEDB Insert                                    |
| MIR7     | DBFMIRT0 | Update RAP -> new segment         | DEDB Insert                                    |
| MIR8     | DBFMIRT0 | Previous is not a RAP             | DEDB Insert                                    |
| MIR9     | DBFMIRT0 | Previous Segment is Twin          | DEDB Insert                                    |
| MIRA     | DBFMIRT0 | Set Log Data                      | DEDB Insert                                    |
| ISLL     | DBFMISL9 | Entry, Exit(Shift)                | DEDB Process Insert Last Level                 |
| MLCS     | DBFMLCL0 | Entry                             | DEDB Logical Close Area                        |
| MLCE     | DBFMLCL0 | Exit                              | DEDB Logical Close Area                        |
| MLEV     | DBFMLEV0 | Entry                             | DEDB Adjust MLTE Sequence Numbers              |
| NLEV     | DBFMLEV0 | Exit                              | DEDB Adjust MLTE Sequence Numbers              |
| MLOG     | DBFMLOG0 | Entry                             | DEDB SDEP CI Logging                           |
| NLOG     | DBFMLOG0 | Exit                              | DEDB SDEP CI Logging                           |
| MLOS     | DBFMLOP0 | Entry                             | DEDB Logical Open Area                         |
| MLOE     | DBFMLOP0 | Exit                              | DEDB Logical Open Area                         |
| MMIT     | DBFMMIT0 | Entry                             | DEDB Media Manager Connect/Disconnect          |
| MMIE     | DBFMMIT0 | Exit                              | DEDB Media Manager Connect/Disconnect          |
| MOCI     | DBFMOCIO | Entry                             | DEDB DMAC Update                               |
| NOCI     | DBFMOCIO | Exit                              | DEDB DMAC Update                               |
| PCC9     | DBFMPCC9 | Entry, Exit(Shift)                | DEDB Process 'C' Command Code                  |
| MCLS     | DBFMPCL0 | Entry                             | DEDB Physical Area Close                       |
| MCLE     | DBFMPCL0 | Exit                              | DEDB Physical Area Close                       |

Table 222. Fast Path Trace Entries (continued)

| Trace ID | Module   | Trace Point                        | Comments                                                |
|----------|----------|------------------------------------|---------------------------------------------------------|
| PED9     | DBFMPE9  | Entry,Exit (Shift)                 | DEDB Process Position Fields in Parallel EPCBs - DELETE |
| PEI9     | DBFMPEI9 | Entry, Exit (Shift)                | DEDB Process Position Fields in Parallel EPCBs - INSERT |
| LOP1     | DBFMPEI9 | Loop through EPCBs                 | DEDB Process Position Fields in Parallel EPCBs - INSERT |
| LOP2     | DBFMPEI9 | Loop through MLTEs                 | DEDB Process Position Fields in Parallel EPCBs - INSERT |
| FRGU     | DBFMPEI9 | Update before EPCB GU Position     | DEDB Process Position Fields in Parallel EPCBs - INSERT |
| FGRN     | DBFMPEI9 | Update before EPCB GN Position     | DEDB Process Position Fields in Parallel EPCBs - INSERT |
| AFGU     | DBFMPEI9 | Update after EPCB GU Position      | DEDB Process Position Fields in Parallel EPCBs - INSERT |
| DPTX     | DBFMPEI9 | Entry                              | DEDB Relocate PCL or SSPT in Parent                     |
| DPTX     | DBFMPEI9 | Exit                               | DEDB Relocate PCL or SSPT in Parent                     |
| MPGO     | DBFMPEI9 | Entry, Exit(Shift)                 | DEDB Process PROCOPT GOX, GON                           |
| PIO9     | DBFMPEI9 | Entry, Exit(Shift)                 | DEDB Process I/O Area for REPLACE                       |
| MOPS     | DBFMPEI9 | Entry                              | DEDB Physical Area Open                                 |
| MOPE     | DBFMPEI9 | Exit                               | DEDB Physical Area Open                                 |
| MPOS     | DBFMPEI9 | Entry                              | DEDB POS Call                                           |
| NPOS     | DBFMPEI9 | Exit                               | DEDB POS Call                                           |
| MGUP     | DBFMPEI9 | Entry to Find Root Segment         | DEDB POS Call                                           |
| MGNP     | DBFMPEI9 | Entry to find next SDEP Segment    | DEDB POS Call                                           |
| NGN0     | DBFMPEI9 | Exit from find next SDEP Segment   | DEDB POS Call                                           |
| MPO2     | DBFMPEI9 | Notify Partners to Harden SDEP Cis | DEDB POS Call                                           |
| VMAI     | DBFMPEI9 | Exit from Notify Partners          | DEDB POS Call                                           |
| MPSG     | DBFMPEI9 | Entry, Exit(Shift)                 | DEDB Process Subset Pointer Commands S W Z M            |
| SSPL     | DBFMPEI9 | Loop down through MLTEs            | DEDB Process Subset Pointer Commands S W Z M            |
| LOPC     | DBFMPEI9 | Loop over SSPTRs                   | DEDB Process Subset Pointer Commands S W Z M            |
| SCOM     | DBFMPEI9 | Command Code S                     | DEDB Process Subset Pointer Commands S W Z M            |
| WCOM     | DBFMPEI9 | Command Code W                     | DEDB Process Subset Pointer Commands S W Z M            |
| ZCOM     | DBFMPEI9 | Command Code Z                     | DEDB Process Subset Pointer Commands S W Z M            |
| MPUG     | DBFMPEI9 | Entry, Exit(Shift)                 | DEDB Process Unqualified GN                             |
| FCHL     | DBFMPEI9 | First Child                        | DEDB Process Unqualified GN                             |

Table 222. Fast Path Trace Entries (continued)

| Trace ID | Module   | Trace Point                          | Comments                                    |
|----------|----------|--------------------------------------|---------------------------------------------|
| MUP1     | DBFMPEG0 | Move Up A Level                      | DEDB Process Unqualified GN                 |
| VIO1     | DBFMPEG0 | Violation 1                          | DEDB Process Unqualified GN                 |
| VIO2     | DBFMPEG0 | Violation 2                          | DEDB Process Unqualified GN                 |
| GSBL     | DBFMPEG0 | Get Sibling                          | DEDB Process Unqualified GN                 |
| MPU2     | DBFMPEG0 | Sibling Located                      | DEDB Process Unqualified GN                 |
| MRCU     | DBFMRCU0 | Entry, Exit(Shift)                   | DEDB Read Current Dependent Segment         |
| MRPU     | DBFMRPU0 | Entry, Exit(Shift)                   | DEDB Reset Position after Unqualified GN    |
| LMLT     | DBFMRPU0 | Loop through MLTEs                   | DEDB Reset Position after Unqualified GN    |
| MRPL     | DBFMRPX0 | Entry, Exit(Shift)                   | DEDB REPLACE                                |
| AMST     | DBFMRPX0 | Return AM status code                | DEDB REPLACE                                |
| SPR9     | DBFMRPX0 | Update PRBA in MLTE of children      | DEDB REPLACE                                |
| SPR1     | DBFMRPX0 | Loop through siblings                | DEDB REPLACE                                |
| SPR2     | DBFMRPX0 | End loop through siblings            | DEDB REPLACE                                |
| PED9     | DBFMRPX0 | End child PRBA updates               | DEDB REPLACE                                |
| MRQC     | DBFMRQC0 | Entry, Exit(Shift)                   | DEDB Retrieve by Qualified Call             |
| MRQU     | DBFMRQC0 | Check current position               | DEDB Retrieve by Qualified Call             |
| MRUU     | DBFMRQC0 | MLTE not qualified                   | DEDB Retrieve by Qualified Call             |
| MRNQ     | DBFMRQC0 | Get Next Loop                        | DEDB Retrieve by Qualified Call             |
| MRNU     | DBFMRQC0 | Not qualified SSA                    | DEDB Retrieve by Qualified Call             |
| MRUF     | DBFMRQC0 | At Least Root Satisfies              | DEDB Retrieve by Qualified Call             |
| REQ1     | DBFMRQC0 | Diverge U or GN *F                   | DEDB Retrieve by Qualified Call             |
| REQL     | DBFMRQC0 | MLTE downward loop qualification     | DEDB Retrieve by Qualified Call             |
| RC04     | DBFMRQC0 | Found, moved off current position    | DEDB Retrieve by Qualified Call             |
| RC08     | DBFMRQC0 | Not found at that level              | DEDB Retrieve by Qualified Call             |
| LROT     | DBFMRQC0 | Loop up to root                      | DEDB Retrieve by Qualified Call             |
| TOGH     | DBFMRQC0 | GU, all levels satisfied current pos | DEDB Retrieve by Qualified Call             |
| DIVE     | DBFMRQC0 | MLTE Loop to clear DIVERGE flag      | DEDB Retrieve by Qualified Call             |
| NOPA     | DBFMRQC0 | Not a PATH call                      | DEDB Retrieve by Qualified Call             |
| PMVE     | DBFMRQC0 | Data to be moved                     | DEDB Retrieve by Qualified Call             |
| PHIL     | DBFMRQC0 | Path call Highest Level to move      | DEDB Retrieve by Qualified Call             |
| PCOM     | DBFMRQC0 | Path call complete                   | DEDB Retrieve by Qualified Call             |
| PLOP     | DBFMRQC0 | Loop for P command up in MLTEs       | DEDB Retrieve by Qualified Call             |
| MRUC     | DBFMRUC0 | Entry, Exit(Shift)                   | DEDB Reset U Command at Current/Lower Level |
| SFIT     | DBFMFSI9 | Entry, Exit(Shift)                   | DEDB Search Field Name                      |

Table 222. Fast Path Trace Entries (continued)

| Trace ID | Module    | Trace Point                    | Comments                                  |
|----------|-----------|--------------------------------|-------------------------------------------|
| SFLP     | DBFMFSI9  | Loop over Fields in Segment    | DEDB Search Field Name                    |
| SFTP     | DBFMFSI9  | Verify Relational Operator     | DEDB Search Field Name                    |
| CALL     | DBFMFSI9  | Trace SSA and Fields           | DEDB Search Field Name                    |
| SFO9     | DBFMSFO9  | Entry, Exit(Shift)             | DEDB Set First Position of Segment Type   |
| PPRE     | DBFMSFO9  | Trace Parent Prefix            | DEDB Set First Position of Segment Type   |
| SFOT     | DBFMSFO9  | No Floating Pointer in Call    | DEDB Set First Position of Segment Type   |
| SIMP     | DBFMSIM9  | Entry, Exit(Shift)             | DEDB Set Implied for Upper Levels of Call |
| MSPC     | DBFMSPC0  | Entry                          | DEDB IOVF Free Space Calculator           |
| NSPC     | DBFMSPC0  | Exit                           | DEDB IOVF Free Space Calculator           |
| MSRB     | DBFMSTRB0 | Entry                          | DEDB Schedule DBFMIOS0 SRB routine        |
| NSRB     | DBFMSTRB0 | Exit                           | DEDB Schedule DBFMIOS0 SRB routine        |
| MSRT     | DBFMSRT0  | Entry                          | DEDB Insert SDEP Segment to LSRT          |
| NSRT     | DBFMSRT0  | Exit                           | DEDB Insert SDEP Segment to LSRT          |
| MSR1     | DBFMSRT0  | MSRTCIAL entry                 | DEDB Preallocate SDEP Cis                 |
| MSR2     | DBFMSRT0  | After Recheck #1 still need PA | DEDB Preallocate SDEP Cis                 |
| MSR3     | DBFMSRT0  | After Recheck #2 still need PA | DEDB Preallocate SDEP Cis                 |
| MSR4     | DBFMSRT0  | DMAC Read Error                | DEDB Preallocate SDEP Cis                 |
| MSR5     | DBFMSRT0  | DMAC Read Successful           | DEDB Preallocate SDEP Cis                 |
| MSR6     | DBFMSRT0  | SDEP Part full after DMAC Read | DEDB Preallocate SDEP Cis                 |
| MSR7     | DBFMSRT0  | Allocate an RBAT               | DEDB Preallocate SDEP Cis                 |
| MSR8     | DBFMSRT0  | DMACLBTs was zero, set here    | DEDB Preallocate SDEP Cis                 |
| MSR9     | DBFMSRT0  | Trace ACCUM_LENG               | DEDB Preallocate SDEP Cis                 |
| MSRA     | DBFMSRT0  | Trace RBAT, #Cis to allocate   | DEDB Preallocate SDEP Cis                 |
| MSRB     | DBFMSRT0  | Trace RBAT, min # Cis          | DEDB Preallocate SDEP Cis                 |
| MSRD     | DBFMSRT0  | SDEP Part now full             | DEDB Preallocate SDEP Cis                 |
| MSRE     | DBFMSRT0  | SDEP Part now full             | DEDB Preallocate SDEP Cis                 |
| MSRF     | DBFMSRT0  | Lock failure on SDEP PACI      | DEDB Preallocate SDEP Cis                 |
| MSRG     | DBFMSRT0  | Add SDEP CI XCRBs to RBAT      | DEDB Preallocate SDEP Cis                 |
| MSRH     | DBFMSRT0  | Trace ACCUM_LENG               | DEDB Preallocate SDEP Cis                 |
| MLOG     | DBFMSRT0  | Log 5953 Record - entry        | DEDB Preallocate SDEP Cis                 |
| NLOG     | DBFMSRT0  | Log 5953 Record - exit         | DEDB Preallocate SDEP Cis                 |
| SSA9     | DBFMSSA9  | Entry, Exit(Shift)             | DEDB Search SSA for Data                  |
| MMOV     | DBFMSSA9  | Must Move                      | DEDB Search SSA for Data                  |

Table 222. Fast Path Trace Entries (continued)

| Trace ID | Module   | Trace Point                     | Comments                                               |
|----------|----------|---------------------------------|--------------------------------------------------------|
| BACK     | DBFMSSA9 | Must Move Back                  | DEDB Search SSA for Data                               |
| STAY     | DBFMSSA9 | Stay at this Level              | DEDB Search SSA for Data                               |
| FOND     | DBFMSSA9 | Found                           | DEDB Search SSA for Data                               |
| COML     | DBFMSSA9 | Command Code L                  | DEDB Search SSA for Data                               |
| TWLF     | DBFMSSA9 | Not found - higher key found    | DEDB Search SSA for Data                               |
| EIGH     | DBFMSSA9 | Not found, no higher key found  | DEDB Search SSA for Data                               |
| SAGI     | DBFMSSC9 | Entry, Exit(Shift)              | DEDB SSA Handler for GET and INSERT                    |
| ACST     | DBFMCSS9 | Count SSAs                      | DEDB SSA Handler for GET and INSERT                    |
| NAMF     | DBFMCSS9 | Segment Name Found              | DEDB SSA Handler for GET and INSERT                    |
| DESC     | DBFMCSS9 | Level Descending                | DEDB SSA Handler for GET and INSERT                    |
| NDES     | DBFMCSS9 | Level Not Descending            | DEDB SSA Handler for GET and INSERT                    |
| NOTN     | DBFMCSS9 | Segment Name Not Found          | DEDB SSA Handler for GET and INSERT                    |
| NOSG     | DBFMCSS9 | No Segment Name Found           | DEDB SSA Handler for GET and INSERT                    |
| ENAC     | DBFMCSS9 | Hierarchy Error in Segment Name | DEDB SSA Handler for GET and INSERT                    |
| SSD9     | DBFMSSD9 | Entry, Exit(Shift)              | DEDB SSA Handler for DELETE                            |
| SAGE     | DBFMSSG9 | Entry,Exit (Shift)              | DEDB SSA Handler for GET                               |
| SAIN     | DBFMSSI9 | Entry,Exit (Shift)              | DEDB SSA Handler for INSERT                            |
| SSP9     | DBFMSSP9 | Entry,Exit (Shift)              | DEDB SSA Handler for POS                               |
| SSR9     | DBFMSSR9 | Entry,Exit (Shift)              | DEDB SSA Handler for REPLACE                           |
| MSTP     | DBFMSTP0 | Entry                           | DEDB I/O substitute routine when ADS is closing/closed |
| NSTP     | DBFMSTP0 | Exit                            | DEDB I/O substitute routine when ADS is closing/closed |
| SVC9     | DBFMVVC9 | Entry, Exit(Shift)              | DEDB set V Command as U command in MLTE                |
| MUHE     | DBFMUHE0 | Entry                           | DEDB Update Log Entry in DMHR                          |
| NUHE     | DBFMUHE0 | Exit                            | DEDB Update Log Entry in DMHR                          |
| MUH1     | DBFMUHE1 | Entry, Exit(Shift)              | DEDB Front End/Back End Elimination                    |
| MUPB     | DBFMUPB0 | Entry                           | DEDB View=MSDB Update Buffer handler                   |
| XUPB     | DBFMUPB0 | Exit                            | DEDB View=MSDB Update Buffer handler                   |
| MVfy     | DBFMVfy0 | Entry                           | DEDB FLD Call Verify Processor                         |
| XVfy     | DBFMVfy0 | Exit                            | DEDB FLD Call Verify Processor                         |
| VSNA     | DBFMVSN9 | Entry, Exit(Shift)              | DEDB Call Handler Verify Segment Name                  |
| FOND     | DBFMVSN9 | Segment Name Found              | DEDB Call Handler Verify Segment Name                  |
| NOTM     | DBFNOTM0 | Entry                           | Intersystem NOTIFY processor                           |
| EOTM     | DBFNOTM0 | Exit                            | Intersystem NOTIFY processor                           |
| PDNA     | DBFPDNA0 | Entry                           | DEDB SETR Positioning                                  |
| PGA5     | DBFPDNA0 | Exit                            | DEDB SETR Positioning                                  |

Table 222. Fast Path Trace Entries (continued)

| Trace ID | Module   | Trace Point                      | Comments                                   |
|----------|----------|----------------------------------|--------------------------------------------|
| PENQ     | DBFPENQ0 | Entry                            | DEDB UOW Resource Enqueue                  |
| NENQ     | DBFPENQ0 | Exit                             | DEDB UOW Resource Enqueue                  |
| PFDS     | DBFPFDS0 | Entry                            | DEDB Unallocate, Unchain, and release ADSC |
| PGAB     | DBFPGAB0 | Entry                            | DEDB Get Private Buffer or Buffers         |
| NGAB     | DBFPGAB0 | Exit                             | DEDB Get Private Buffer or Buffers         |
| PGAP     | DBFPGAP0 | Entry                            | DEDB HSSP Positioning                      |
| PGAE     | DBFPGAP0 | Exit                             | DEDB HSSP Positioning                      |
| PGA1     | DBFPGAP0 | New AREA                         | DEDB HSSP Positioning                      |
| PGA2     | DBFPGAP0 | Previous AREA still active       | DEDB HSSP Positioning                      |
| PGA3     | DBFPGAP0 | Same AREA                        | DEDB HSSP Positioning                      |
| PGDS     | DBFPGDS0 | Entry                            | DEDB Allocate and Chain ADSC               |
| PHST     | DBFPHST0 | Entry                            | DEDB HSSP/Utility Process Termination      |
| PICS     | DBFPICS0 | Entry                            | DEDB HSSP Image Copy Process Setup         |
| NICS     | DBFPICS0 | Exit                             | DEDB HSSP Image Copy Process Setup         |
| PIOS     | DBFPIOS0 | Entry                            | DEDB HSSP Image Copy I/O Routine           |
| IOSH     | DBFPIOS0 | Imac_IC_cursor GT req uow 1st CI | DEDB HSSP Image Copy I/O Routine           |
| IOSL     | DBFPIOS0 | Imac_IC_cursor LT req uow 1st CI | DEDB H DEDB HSSP Image Copy I/O Routine    |
| IOSE     | DBFPIOS0 | Imac_IC_cursor EQ req uow 1st CI | DEDB HSSP Image Copy I/O Routine           |
| NIOS     | DBFPIOS0 | Exit                             | DEDB HSSP Image Copy I/O Routine           |
| PRAB     | DBFPRAB0 | Entry                            | DEDB Release Current UOW Resources         |
| NRAB     | DBFPRAB0 | Exit                             | DEDB Release Current UOW Resources         |
| PSET     | DBFPSET0 | Entry, Exit(Shift)               | DEDB HSSP Process Setup                    |
| PULI     | DBFPULI0 | Entry                            | DEDB UOW Lock Mode Initiation              |
| NULI     | DBFPULI0 | Exit                             | DEDB UOW Lock Mode Initiation              |
| PUL1     | DBFPULI0 | Wait for CI locks to be released | DEDB UOW Lock Mode Initiation              |
| PUXC     | DBFPUXC0 | Entry                            | DEDB UOW Resource Handler                  |
| NUXC     | DBFPUXC0 | Exit                             | DEDB UOW Resource Handler                  |
| LUXC     | DBFPUXC0 | Lock Subroutine Entry            | DEDB UOW Resource Handler                  |
| PUXR     | DBFPUXR0 | Entry                            | DEDB Release UXRBS                         |
| NUXR     | DBFPUXR0 | Exit                             | DEDB Release UXRBS                         |
| BSBP     | DBFSBP10 | Entry                            | MSDB Syncpoint Phase I                     |
| RSBP     | DBFSBP10 | Exit                             | MSDB Syncpoint Phase I                     |
| BACC     | DBFSBP10 | Use MSDB data for operation      | MSDB Syncpoint Phase I                     |
| RACC     | DBFSBP10 | Use Record data for operation    | MSDB Syncpoint Phase I                     |



Table 222. Fast Path Trace Entries (continued)

| Trace ID | Module   | Trace Point                       | Comments                                     |
|----------|----------|-----------------------------------|----------------------------------------------|
| CACC     | DBFSBP10 | Trace Segment to be processed     | MSDB Syncpoint Phase I                       |
| DACC     | DBFSBP10 | After move to EPST work area      | MSDB Syncpoint Phase I                       |
| EACC     | DBFSBP10 | Move MSDB Data                    | MSDB Syncpoint Phase I                       |
| FACC     | DBFSBP10 | Move MSDB Data                    | MSDB Syncpoint Phase I                       |
| BMSG     | DBFSBP10 | Setup Arithmetic Overflow message | MSDB Syncpoint Phase I                       |
| SDEQ     | DBFSDEQ0 | Entry                             | FP Resource Dequeue                          |
| TDEQ     | DBFSDEQ0 | Exit                              | FP Resource Dequeue                          |
| UOWX     | DBFSDEQ0 | Change UOW Lock Ownership         | FP Resource Dequeue                          |
| SFLD     | DBFSFLD0 | Entry                             | DEDB Syncpoint Phase I FLD Call              |
| RSFL     | DBFSFLD0 | Exit                              | DEDB Syncpoint Phase I FLD Call              |
| SGAB     | DBFSGAB0 | Entry                             | DEDB Get Buffer from Shared Pool             |
| ZGAB     | DBFSGAB0 | Exit                              | DEDB Get Buffer from Shared Pool             |
| SHQD     | DBFSHDQ0 | Entry                             | DEDB HSSP Resource Dequeue Phase II          |
| SHDX     | DBFSHDQ0 | Exit                              | DEDB HSSP Resource Dequeue Phase II          |
| HPRE     | DBFSHDQ0 | Trace DMAC                        | DEDB HSSP Resource Dequeue Phase II          |
| HPR1     | DBFSHDQ0 | Application got GC status         | DEDB HSSP Resource Dequeue Phase II          |
| HPR3     | DBFSHDQ0 | Application did not get GC status | DEDB HSSP Resource Dequeue Phase II          |
| HPRA     | DBFSHDQ0 | Sync Abort Flow                   | DEDB HSSP Resource Dequeue Phase II          |
| SIC1     | DBFSIC10 | Entry DBFSIC10                    | DEDB HSSP Image Copy Phase I and Phase II    |
| NIC1     | DBFSIC10 | Exit DBFSIC10                     | DEDB HSSP Image Copy Phase I and Phase II    |
| SIC2     | DBFSIC10 | Entry DBFSIC20                    | DEDB HSSP Image Copy Phase I and Phase II    |
| NIC2     | DBFSIC10 | Exit DBFSIC10                     | DEDB HSSP Image Copy Phase I and Phase II    |
| SICC     | DBFSIC10 | Good Sync, enqueue DMHRSET        | DEDB HSSP Image Copy Phase I and Phase II    |
| SICE     | DBFSIC10 | Last UOW, enqueue TERM AWE        | DEDB HSSP Image Copy Phase I and Phase II    |
| SLGE     | DBFSLGE0 | Entry, Exit(Shift)                | DEDB Sync Log Exit                           |
| SLGE2S   | DBFSLGE2 | Entry                             | DEDB Sync Log Exit for Segment Level Locking |
| SLGE20E  | DBFSLGE2 | Exit                              | DEDB Sync Log Exit for Segment Level Locking |
| SLG2     | DBFSLG20 | Entry, Exit(Shift)                | DEDB Sync/Abort Log Processor                |
| SLOG     | DBFSLOG0 | Entry,Bad Sync(Shift)             | FP Log Processor                             |
| TLOG     | DBFSLOG0 | Exit                              | FP Log Processor                             |
| MP10     | DBFSMP10 | Entry                             | DEDB SDEP Syncpoint Phase I                  |
| SYPB     | DBFSMP10 | New current SDEP Buffer           | DEDB SDEP Syncpoint Phase I                  |
| MP11     | DBFSMP10 | Trace #PA Cis, #required Cis      | DEDB SDEP Syncpoint Phase I                  |

Table 222. Fast Path Trace Entries (continued)

| Trace ID | Module   | Trace Point               | Comments                                          |
|----------|----------|---------------------------|---------------------------------------------------|
| MP12     | DBFSMP10 | Trace needed CI Space     | DEDB SDEP Syncpoint Phase I                       |
| MP13     | DBFSMP10 | Trace new CI space        | DEDB SDEP Syncpoint Phase I                       |
| NP10     | DBFSMP10 | Exit                      | DEDB SDEP Syncpoint Phase I                       |
| SPIX     | DBFSPIX0 | Entry                     | DEDB Process unused XCRB/DMHR at Sync             |
| XPIX     | DBFSPIX0 | Exit                      | DEDB Process unused XCRB/DMHR at Sync             |
| BSY0     | DBFSYN00 | Entry                     | Pure FP Sync Point Controller                     |
| SYN1     | DBFSYN10 | Entry                     | FP Syncpoint Phase I Controller                   |
| TYN1     | DBFSYN10 | Exit                      | FP Syncpoint Phase I Controller                   |
| SYN2     | DBFSYN20 | Entry, Entry 2nd Call     | FP Syncpoint Phase II Controller                  |
| TYN2     | DBFSYN20 | Exit                      | FP Syncpoint Phase II Controller                  |
| SYP2     | DBFSYP20 | Entry                     | FP Syncpoint Phase II                             |
| TYP2     | DBFSYP20 | Exit                      | FP Syncpoint Phase II                             |
| SYPB     | DBFSYP20 | New current SDEP Buffer   | FP Syncpoint Phase II                             |
| AFCE     | DBFTAFC9 | Entry                     | Analyze FPTCNTRL control cards                    |
| AFCX     | DBFTAFC9 | Exit                      | Analyze FPTCNTRL control cards                    |
| ATCE     | DBFTATC9 | Entry                     | Analyze Trace Calls                               |
| TON0     | DBFTATC9 | Trace ON                  | Analyze Trace Calls                               |
| TOFF     | DBFTATC9 | Trace OFF                 | Analyze Trace Calls                               |
| TSEL     | DBFTATC9 | TOM Table built           | Analyze Trace Calls                               |
| TDAT     | DBFTATC9 | Deactivate Trace          | Analyze Trace Calls                               |
| TACT     | DBFTATC9 | Activate Trace            | Analyze Trace Calls                               |
| ATCX     | DBFTATC9 | Exit without error        | Analyze Trace Calls                               |
| ATCY     | DBFTATC9 | Error Exit                | Analyze Trace Calls                               |
| BLTE     | DBFTBLT9 | Entry                     | Build TOM Table Structure                         |
| BLTX     | DBFTBLT9 | Exit                      | Build TOM Table Structure                         |
| BMIE     | DBFTBMI9 | Entry                     | Build Trace Message In I/O Area                   |
| BMIX     | DBFTBMI9 | Exit                      | Build Trace Message In I/O Area                   |
| COTE     | DBFTCOT9 | Entry                     | Construct Trace Option Table                      |
| COT1     | DBFTCOT9 | Trace before IMODULE LOAD | Construct Trace Option Table                      |
| COT2     | DBFTCOT9 | After call to DBFTBLT9    | Construct Trace Option Table                      |
| ILAR     | DBFTCOT9 | After GETMAIN             | Construct Trace Option Table                      |
| COT3     | DBFTCOT9 | After GETMAIN             | Construct Trace Option Table                      |
| COT4     | DBFTCOT9 | Delete TOM                | Construct Trace Option Table                      |
| COTX     | DBFTCOT9 | Exit without error        | Construct Trace Option Table                      |
| COTY     | DBFTCOT9 | Error Exit                | Construct Trace Option Table                      |
| ABN2     | DBFTDEB9 | Trace without SDWA        | Provide Debugging Information for Abending Module |
| ABN1     | DBFTDEB9 | Trace with SDWA           | Provide Debugging Information for Abending Module |

Table 222. Fast Path Trace Entries (continued)

| Trace ID | Module   | Trace Point                        | Comments                                       |
|----------|----------|------------------------------------|------------------------------------------------|
| FTOE     | DBFTDEB9 | Entry                              | Free Previous TOM                              |
| FTOX     | DBFTDEB9 | Exit                               | Free Previous TOM                              |
| TRAF     | DBFTIR1S | Trace OFF                          | DBFIRC10 Connection to FP Trace                |
| TRAN     | DBFTIR1S | Trace ON                           | DBFIRC10 Connection to FP Trace                |
| SIEE     | DBFTSIE9 | Entry                              | Setup Initial Environment for FP Trace         |
| SIEX     | DBFTSIE9 | Exit                               | Setup Initial Environment for FP Trace         |
| STS9     | DBFTSTS9 | Entry                              | Set Trace Suppress Flag                        |
| STSX     | DBFTSTS9 | Exit                               | Set Trace Suppress Flag                        |
| VIAE     | DBFTVIA9 | Entry                              | Verify I/O Area of Trace Call                  |
| TCAL     | DBFTVIA9 | Trace I/O Area                     | Verify I/O Area of Trace Call                  |
| TC01     | DBFTVIA9 | Call is TON or TOFF                | Verify I/O Area of Trace Call                  |
| TC02     | DBFTVIA9 | Call is TSEL                       | Verify I/O Area of Trace Call                  |
| TC03     | DBFTVIA9 | Trace 1st ID in I/O Area           | Verify I/O Area of Trace Call                  |
| VIAX     | DBFTVIA9 | Exit without error                 | Verify I/O Area of Trace Call                  |
| VIAY     | DBFTVIA9 | Error Exit                         | Verify I/O Area of Trace Call                  |
| 24BE     | DBFT24B0 | Entry                              | Trace Get/Put Routines in 24-bit mode          |
| 24BX     | DBFT24B0 | Exit                               | Trace Get/Put Routines in 24-bit mode          |
| UHAC     | DBFUHAC7 | Entry                              | DEDB HSRE Access segment in non reorganized CI |
| UHAX     | DBFUHAC7 | Exit                               | DEDB HSRE Access segment in non reorganized CI |
| UHAR     | DBFUHAR0 | Entry                              | DEDB HS Utility Async Read-Ahead               |
| HAR2     | DBFUHAR0 | Wait for inflight async read ahead | DEDB HS Utility Async Read-Ahead               |
| HAR1     | DBFUHAR0 | Setup SRB for async read ahead     | DEDB HS Utility Async Read-Ahead               |
| UHAX     | DBFUHAR0 | Exit                               | DEDB HS Utility Async Read-Ahead               |
| PCH0     | DBFUHAR0 | Entry to routine UHARPCH0          | DEDB HS Utility Async Read-Ahead               |
| PCH1     | DBFUHAR0 | Found non-HSSP XCRB in UOW         | DEDB HS Utility Async Read-Ahead               |
| PCH2     | DBFUHAR0 | XCRB has DMHR                      | DEDB HS Utility Async Read-Ahead               |
| PCH3     | DBFUHAR0 | Data copied to HSSP buffer         | DEDB HS Utility Async Read-Ahead               |
| PCH4     | DBFUHAR0 | Release DMHR and XCRB              | DEDB HS Utility Async Read-Ahead               |
| PCHX     | DBFUHAR0 | Exit from routine UHARPCH0         | DEDB HS Utility Async Read-Ahead               |
| UHDA     | DBFUHDA0 | Entry                              | DEDB HSRE Process Alloc/Dealloc of IOVF Cis    |
| UHDX     | DBFUHDA0 | Exit                               | DEDB HSRE Process Alloc/Dealloc of IOVF Cis    |
| UHDS     | DBFUHDS0 | Entry                              | DMAC Sync                                      |
| UHDX     | DBFUHDS0 | Exit                               | DMAC Sync                                      |
| UHGS     | DBFUHGS7 | Entry                              | DEDB HSRE Get Space to Copy Segment            |

Table 222. Fast Path Trace Entries (continued)

| Trace ID | Module   | Trace Point                        | Comments                                    |
|----------|----------|------------------------------------|---------------------------------------------|
| UHGX     | DBFUHGS7 | Exit                               | DEDB HSRE Get Space to Copy Segment         |
| UHIO     | DBFUHIO0 | Entry                              | DEDB HSRE Read a specified CI               |
| UHIX     | DBFUHIX0 | Exit                               | DEDB HSRE Read a specified CI               |
| UHPR     | DBFUHPR7 | Entry                              | DEDB HSRE Process Root Chain of RAP CI      |
| UHPX     | DBFUHPR7 | Exit                               | DEDB HSRE Process Root Chain of RAP CI      |
| UHRD     | DBFUHRD7 | Entry                              | DEDB HSRE Read Database Record              |
| UHRX     | DBFUHRD7 | Exit                               | DEDB HSRE Read Database Record              |
| UHRE     | DBFUHRE0 | Entry                              | DEDB HSRE UOW Reorg Mainline                |
| UHRX     | DBFUHRE0 | Exit                               | DEDB HSRE UOW Reorg Mainline                |
| UHSR     | DBFUHSR0 | Entry                              | DEDB HSRE Mainline                          |
| UHSX     | DBFUHSR0 | Exit                               | DEDB HSRE Mainline                          |
| UHSS     | DBFUHSS0 | Entry                              | DEDB High Speed Utility Services            |
| UHS1     | DBFUHSS0 | Wait for inflight async read ahead | DEDB High Speed Utility Services            |
| NHSS     | DBFUHSS0 | Exit                               | DEDB High Speed Utility Services            |
| UHSI     | DBFUHSS0 | HSSP Image Copy Termination        | DEDB High Speed Utility Services            |
| UMAF     | DBFUMAF0 | Entry                              | DEDB Utility Page Fix Services              |
| VMAF     | DBFUMAF0 | Exit                               | DEDB Utility Page Fix Services              |
| UMAI     | DBFUMAI0 | Entry                              | DEDB Utility I/O Services                   |
| ENQR     | DBFUMAI0 | Test for SDEP CI Lock              | DEDB Utility I/O Services                   |
| VMAI     | DBFUMAI0 | Exit                               | DEDB Utility I/O Services                   |
| UMAL     | DBFUMAL0 | Entry                              | DEDB Utility Logging                        |
| VMAL     | DBFUMAL0 | Exit                               | DEDB Utility Logging                        |
| UMAN     | DBFUMAN0 | Entry                              | DEDB Utility Services                       |
| VMAN     | DBFUMAN0 | Exit                               | DEDB Utility Services                       |
| UMAV     | DBFUMAV0 | Entry                              | DEDB Utility Set ADS Available              |
| VMAC     | DBFUMAV0 | Exit                               | DEDB Utility Set ADS Available              |
| UMDS     | DBFUMDS0 | Entry                              | DEDB Utility DMAC Sync (CONNECT/DISCONNECT) |
| VMDS     | DBFUMDS0 | Exit                               | DEDB Utility DMAC Sync (CONNECT/DISCONNECT) |
| UMFT     | DBFUMFT0 | Entry                              | DEDB ADS Format                             |
| VMFT     | DBFUMFT0 | Exit                               | DEDB ADS Format                             |
| UMMT     | DBFUMMT0 | Entry                              | DEDB Utility MTO Message Services           |
| UNMT     | DBFUMMT0 | Exit                               | DEDB Utility MTO Message Services           |
| UMNO     | DBFUMNO0 | Entry                              | DEDB Utility Notify Partners to Open ADS    |
| VMNO     | DBFUMNO0 | Exit                               | DEDB Utility Notify Partners to Open ADS    |
| VSCL     | DBFVSCL0 | Entry                              | DEDB VSO Area Close                         |
| VSCE     | DBFVSCL0 | Exit                               | DEDB VSO Area Close                         |
| VSOP     | DBFVSOP0 | Entry                              | DEDB VSO Area Open                          |

| *Table 222. Fast Path Trace Entries (continued)*

| Trace ID | Module   | Trace Point | Comments                    |
|----------|----------|-------------|-----------------------------|
| VSOE     | DBFVSOP0 | Exit        | DEDB VSO Area Open          |
| XPIX     | DBFXPIX0 | Entry       | Free a chain of XCRBs/UXRBs |
| NPIX     | DBFXPIX0 | Exit        | Free a chain of XCRBs/UXRBs |



## 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.

#### COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrates programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. You may copy, modify, and distribute these sample programs in any form without payment to IBM for the purposes of developing, using, marketing, or distributing application programs conforming to IBM's application programming interfaces.

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:

© (your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corp. \_enter the year or years\_. All rights reserved.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

---

## Programming Interface Information

The information in these topics is intended to help system programmers diagnose IMS problems. This information also documents Diagnosis, Modification or Tuning Information provided by IMS.



Diagnosis, Modification or Tuning information is provided to help you diagnose, modify, or tune IMS. 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, other countries, or both:

|                        |           |
|------------------------|-----------|
| BookManager            | MVS/ESA   |
| CICS                   | NetView   |
| DataPropagator         | OS/390    |
| DB2                    | Tivoli    |
| DB2 Universal Database | VTAM      |
| IBM                    | WebSphere |
| IMS                    | z/OS      |
| MVS                    |           |

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and other countries.

Other company, product, and service names may be trademarks or service marks of others.



## Bibliography

This bibliography includes all the publications cited in this book, including the publications in the IMS library.

- *Authorized Assembler Programming Guide*, GC28-1645
- *DB2 UDB for OS/390 and z/OS Diagnosis Guide and Reference*, LY37-3740
- *IMS Message Requeuer Program Description/Operations Manual*, SH21-1089
- *MVS Data Areas*, LY28-1857 though LY28-1861
- *MVS Diagnosis: Tools and Service Aids*, GA22-7589
- *MVS/ESA Diagnosis: Procedures*, LY28-1844
- *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
- *IMS Queue Control Facility for z/OS User's Guide*, SC26-9685
- *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
- *z/OS MVS Diagnosis: Tools and Service Aids*, GA22-7589
- *z/OS MVS Initialization and Tuning Guide*, SA22-7591
- *z/OS MVS Interactive Problem Control System (IPCS) Customization*, SA22-7594
- *z/OS MVS System Commands*, SA22-7627

### IMS Version 9 Library

| Title                                                           | Acronym | Order number |
|-----------------------------------------------------------------|---------|--------------|
| <i>IMS Version 9: Administration Guide: Database Manager</i>    | ADB     | SC18-7806    |
| <i>IMS Version 9: Administration Guide: System</i>              | AS      | SC18-7807    |
| <i>IMS Version 9: Administration Guide: Transaction Manager</i> | ATM     | SC18-7808    |
| <i>IMS Version 9: Application Programming: Database Manager</i> | APDB    | SC18-7809    |

| Title                                                                             | Acronym | Order number |
|-----------------------------------------------------------------------------------|---------|--------------|
| <i>IMS Version 9: Application Programming: Design Guide</i>                       | APDG    | SC18-7810    |
| <i>IMS Version 9: Application Programming: EXEC DLI Commands for CICS and IMS</i> | APCICS  | SC18-7811    |
| <i>IMS Version 9: Application Programming: Transaction Manager</i>                | APTM    | SC18-7812    |
| <i>IMS Version 9: Base Primitive Environment Guide and Reference</i>              | BPE     | SC18-7813    |
| <i>IMS Version 9: Command Reference</i>                                           | CR      | SC18-7814    |
| <i>IMS Version 9: Common Queue Server Guide and Reference</i>                     | CQS     | SC18-7815    |
| <i>IMS Version 9: Common Service Layer Guide and Reference</i>                    | CSL     | SC18-7816    |
| <i>IMS Version 9: Customization Guide</i>                                         | CG      | SC18-7817    |
| <i>IMS Version 9: Database Recovery Control (DBRC) Guide and Reference</i>        | DBRC    | SC18-7818    |
| <i>IMS Version 9: Diagnosis Guide and Reference</i>                               | DGR     | LY37-3203    |
| <i>IMS Version 9: Failure Analysis Structure Tables (FAST) for Dump Analysis</i>  | FAST    | LY37-3204    |
| <i>IMS Version 9: IMS Connect Guide and Reference</i>                             | CT      | SC18-9287    |
| <i>IMS Version 9: IMS Java Guide and Reference</i>                                | JGR     | SC18-7821    |
| <i>IMS Version 9: Installation Volume 1: Installation Verification</i>            | IIV     | GC18-7822    |
| <i>IMS Version 9: Installation Volume 2: System Definition and Tailoring</i>      | ISDT    | GC18-7823    |
| <i>IMS Version 9: Master Index and Glossary</i>                                   | MIG     | SC18-7826    |
| <i>IMS Version 9: Messages and Codes, Volume 1</i>                                | MC1     | GC18-7827    |
| <i>IMS Version 9: Messages and Codes, Volume 2</i>                                | MC2     | GC18-7828    |
| <i>IMS Version 9: Open Transaction Manager Access Guide and Reference</i>         | OTMA    | SC18-7829    |
| <i>IMS Version 9: Operations Guide</i>                                            | OG      | SC18-7830    |
| <i>IMS Version 9: Release Planning Guide</i>                                      | RPG     | GC17-7831    |

| <b>Title</b>                                                                | <b>Acronym</b> | <b>Order number</b> |
|-----------------------------------------------------------------------------|----------------|---------------------|
| <i>IMS Version 9: Summary of Operator Commands</i>                          | SOC            | SC18-7832           |
| <i>IMS Version 9: Utilities Reference: Database and Transaction Manager</i> | URDBTM         | SC18-7833           |
| <i>IMS Version 9: Utilities Reference: System</i>                           | URS            | SC18-7834           |

---

## Supplementary Publications

| <b>Title</b>                                                                                                                              | <b>Order number</b> |
|-------------------------------------------------------------------------------------------------------------------------------------------|---------------------|
| <i>IMS Connector for Java 2.2.2 and 9.1.0.1 Online Documentation for WebSphere Studio Application Developer Integration Edition 5.1.1</i> | SC09-7869           |
| <i>IMS Version 9 Fact Sheet</i>                                                                                                           | GC18-7697           |
| <i>IMS Version 9: Licensed Program Specifications</i>                                                                                     | GC18-7825           |

---

## Publication Collections

| <b>Title</b>                                                               | <b>Format</b>   | <b>Order number</b> |
|----------------------------------------------------------------------------|-----------------|---------------------|
| IMS Version 9 Softcopy Library                                             | CD              | LK3T-7213           |
| IMS Favorites                                                              | CD              | LK3T-7144           |
| Licensed Bill of Forms (LBOF): IMS Version 9 Hardcopy and Softcopy Library | Hardcopy and CD | LBOF-7789           |
| Unlicensed Bill of Forms (SBOF): IMS Version 9 Unlicensed Hardcopy Library | Hardcopy        | SBOF-7790           |
| OS/390 Collection                                                          | CD              | SK2T-6700           |
| z/OS Software Products Collection                                          | CD              | SK3T-4270           |
| z/OS and Software Products DVD Collection                                  | DVD             | SK3T-4271           |

---

## Accessibility Titles Cited in This Library

| <b>Title</b>                                   | <b>Order number</b> |
|------------------------------------------------|---------------------|
| <i>z/OS V1R1.0 TSO Primer</i>                  | SA22-7787           |
| <i>z/OS V1R5.0 TSO/E User's Guide</i>          | SA22-7794           |
| <i>z/OS V1R5.0 ISPF User's Guide, Volume 1</i> | SC34-4822           |

## Index

### Special characters

- /DIAGNOSE command SNAP function
  - console dump alternative 190
- /TRACE command
  - starting DC trace 327
  - stopping DC trace 328
- (ORTT) Online Recovery Manager
  - example 310
  - format 306
  - starting 306

### Numerics

- 3270 error recovery
  - BTAM DDM 364
  - sense-status message 363
- 6701-MRQB records (command for obtaining) 347
- 6701-MRQE diagnostic records
  - control blocks and mapping macros 346
  - description 345
  - sample JCL for printing 346
- 67D0 log record 138, 409, 410

### A

- abbreviation list 587
- abend dump, cause 259
- abend processing for Spool API support 409
- ABENDU1026
  - Fast Path problem analysis
    - description 415
    - procedure 416
- ABENDUxxxx keyword procedure 33
- ABENDxxx keyword procedure 32
- abnormal save area set 45
- acronym list 587
- active save set
  - finding during DC analysis 363
- ADSC definition/mapping macro 67
- AIBREASN codes
  - description 547
- AIBREASN Codes
  - list 554
  - Set by DFSQMRQ0 547
- AIBREASN codes (Queue Control Facility/Message Requeuer)
  - description 347
- aids for debugging and diagnosing Spool API
  - See debugging and diagnostic aids for Spool API
- ALDS definition/mapping macro 67
- AMPB definition/mapping macro 67
- analyzing problems
  - using log records 127
- APARs
  - preparing 61
  - procedure 61
  - searching for 60

- APPC problem, diagnosing 23
- automatic dump data set allocation 4

### B

- BALG definition/mapping macro 67
  - basic telecommunications access method (BTAM)
    - DDM
      - log record format 364
    - error recovery
      - IEA000I message 364
    - MFS error
      - diagnosing 364
      - diagram of normal BTAM path 364
    - terminal
      - starting DC trace 328
      - stopping DC trace 328
  - batch environment
    - call image capture trace 258
  - BFSP definition/mapping macro 67
  - BFUS definition/mapping macro 67
  - BGNRETRY trace entry 456
  - BHDR definition/mapping macro 67
  - BLOCKHDR definition/mapping macro 67
  - BSPH definition/mapping macro 67
  - BTAM (basic telecommunications access method)
    - DDM
      - log record format 364
    - error recovery
      - IEA000I message 364
    - MFS error
      - diagnosing 364
      - diagram of normal BTAM path 364
    - terminal
      - starting DC trace 328
      - stopping DC trace 328
  - BUFC definition/mapping macro 67
  - BUFENTRY definition/mapping macro 67
  - buffer handler
    - function codes 296
    - module trace IDs 298
    - pool (VSAM) 84
    - return codes 298
  - BUFMSTRA (message processing) trace
    - description 435
  - BUFSMVID trace description 446
- ### C
- CADSECT definition/mapping macro 67
  - call image capture trace
    - batch environment 258
    - online environment 258
    - retrieving data from log data set 258
  - CALLER= parameter
    - FMTIMS statement example 158

- calls used with Spool API support
  - CHNG 405
  - SETO 405
- CBT (control block table) pool 583
- CBT definition/mapping macro 67
- CCB definition/mapping macro 67
- channel-to-channel access method trace stack
  - See MSC (Multiple Systems Coupling)
- CHE FREEZE 53
- CHNG call
  - Spool API 405
- CHNGDUMP MAXSPACE
  - recommended setting 4
- CI (control interval)
  - DEDB problem
    - CI 0 419
    - CI 1 419
    - common data 422
    - first DOVF CI 420
    - first IOVF CI 420
    - other DOVF CIs 420
    - other IOVF CIs 421
    - other SDEP CI 421
    - RAP CI 420
    - scraps 421
    - type identification 419
- CIB definition/mapping macro 67
- CIBSTRAC trace
  - content
    - entry 371
    - example 371
  - locating 371
- CIBTRACE trace
  - content
    - entry 372
    - example 372
  - locating 372
- CIRCA definition/mapping macro 67
- CLB definition/mapping macro 67
- CLLE definition/mapping macro 67
- CNT definition/mapping macro 67
- codes
  - error code examples 406
  - status of CHNG and SETO calls 406
- collecting data
  - APPC-related problem 23
  - control or DL/I region loop 19
  - control region hang 18
  - control region wait 18
  - CQS-related problem 521
  - database-related problem 25
  - DB2 ESS interface problem 20
  - DBCTL-related problem 21
  - DBRC-related problem 21
  - DC-related problem 22
  - ESAF Interface-related problem 25
  - general problems 17
  - IMS dependent region loop 20
  - IMS dependent region wait 20
  - Recovery Resource Service-related problem 26
- Common Service Layer trace (CSLT)
  - about 197
  - format 198
- Common Storage Tracker 4
- common trace table interface 190
- communication analyzer (DFSICIO0)
  - DDM
    - entry point 325
    - trace ID 326
  - description 325
  - save area 326
  - trace output 327
  - trace record example 327
  - trace record format 327
- communication task trace
  - description 433
- COMPARE option 318
- COMPARE statement 255
- component identification keyword procedure 31
- contents
  - DBRC 447
- control (CTL) address space
  - online formatted dump 184
- control address space
  - See CTL address space
- control block
  - acronym 67
  - database manager 92
  - definitions 67, 68, 69, 70, 71, 72, 73
  - description 67
  - external SNAP call 256
  - interrelationship diagram 74
  - linkage for static DB/DC environment 66
  - locating in an IMS dump
    - Fast Path 422
  - locating using load list 581
  - logged at time of error 346
  - macros that generate 67
  - mapping macros 67, 346
  - relationships created for MAIN pool 87
  - relationships for DFSCBT00 pools 91
  - relationships for DFSPOOL pools 90
  - relationships for preallocated storage blocks 88
  - sequential buffering diagram 83
- control block table (CBT) pool 583
- control blocks, relocation during special abend processing 409
- control interval
  - DEDB problem
    - CI 0 419
    - CI 1 419
    - common data 422
    - first DOVF CI 420
    - first IOVF CI 420
    - other DOVF CIs 420
    - other IOVF CIs 421
    - other SDEP CI 421
    - RAP CI 420
    - scraps 421
    - type identification 419
- control region loop, diagnosing 19

control region wait or hang, diagnosing 18  
control region, FMTIMS statement example 159  
CPM definition/mapping macro 67  
CPT definition/mapping macro 67  
CQS (Common Queue Server)  
    additional manual dump intervention 522  
    CQS structure recovery data set 524  
    CQS-z/OS log stream 524  
    diagnosis 521  
    log records 529  
    printing 531  
    problem diagnostics 24  
    set up tracing 10  
    structure dump contents 522  
CQS setup recommendations 7  
    trace environment - conservative 7  
    trace environment - more aggressive 7  
CRB definition/mapping macro 67  
creating output data sets 10  
creating search arguments 30  
CRTR0XIT trace entry 457  
CSAB definition/mapping macro 67  
CSL  
    problem diagnostics 24  
CSLT (Common Service Layer trace)  
    about 197  
    format 198  
CSV definition/mapping macro 67  
CTB definition/mapping macro 68  
CTL (control) address space  
    online formatted dump 184  
CTM definition/mapping macro 68  
CTT definition/mapping macro 68  
CULE definition/mapping macro 68  
CVB definition/mapping macro 68  
CXB definition/mapping macro 68

## D

data communication (DC)  
    call analyzer (DFSDLA30) 358  
    FMTIMS statement example 159  
    service aid  
        DC trace 327  
        description 325  
        finding the active save set 363  
        IBM 3270 error recovery analysis 363  
        IMS Transaction trace 358  
        IMS-VTAM interface 363  
        message format service module traces 371  
        message format service normal BTAM path 364  
        OTMA dumps 405  
        OTMA log records 405  
        OTMA module-to-code cross reference table 403  
        OTMA trace 400  
        OTMA verb-to-code cross reference table 404  
        receive-any buffer analysis 361  
        terminal communication task trace 325  
data management block (DMB) diagram 96  
data sets  
    allocating 10

data sets (*continued*)  
    output 10  
database  
    *See also* DL/I  
    control block diagram 95  
    diagnosis 25  
    diagnostic techniques 253  
    log analysis 313  
    log record (X'50') DSECT 313  
    Recovery Resource Service 26  
    searching techniques 59  
Database Control (DBCTL)  
    DRA dumps 477  
    dump title format 171  
    IMS traces  
        activating 9  
        DL/I 9  
        Fast Path 9  
        problem, diagnosing 21  
        recovery tokens 478  
        service aids 477  
Database Recovery Control  
    control block diagram 110  
    external trace  
        example 470  
        record format 469  
        using 469  
    FMTIMS statement example 159  
    internal trace  
        example 462  
    RECON data set  
        contents 447  
        diagnostic aid 447  
        service aids 447  
Database Resource Adapter (DRA)  
    Analyzing DRA Problems 478  
    dump 477  
    dump title format 171  
    recovery tokens 478  
    Service Aids 477  
DB2 ESS interface problem, diagnosing 20  
DBCTL (Database Control)  
    DRA dumps 477  
    dump title format 171  
    IMS traces  
        activating 9  
        DL/I 9  
        Fast Path 9  
        problem, diagnosing 21  
        recovery tokens 478  
        service aids 477  
DBPCB definition/mapping macro 68  
DBRC (Database Recovery Control)  
    control block diagram 110  
    external trace  
        example 470  
        record format 469  
        using 469  
    FMTIMS statement example 159  
    internal trace  
        example 462

- DBRC (Database Recovery Control) *(continued)*
  - RECON data set
    - contents 447
    - diagnostic aid 447
    - service aids 447
- DBRC problem, diagnosing 21
- DBRC trace (DSPTRACE)
  - BGNRETRY entry 456
  - CRTR0XIT entry 457
  - DSPCABN0 entry 456
  - DSPCRTR0 entry 457
  - DSPSTACK entry 453, 455
  - DSPSTFRE entry 453, 455
  - DSPSTGET entry 453, 454
  - DSPURI00 entry 457, 459
  - GETFEED entry 460
  - locating 452
  - processing flow 454
  - using 452
- DC (data communication)
  - call analyzer (DFSDL30) 358
  - FMTIMS statement example 159
  - service aid
    - DC trace 327
    - description 325
    - finding the active save set 363
    - IBM 3270 error recovery analysis 363
    - IMS Transaction trace 358
    - IMS-VTAM interface 363
    - message format service module traces 371
    - message format service normal BTAM path 364
    - OTMA dumps 405
    - OTMA log records 405
    - OTMA module-to-code cross reference table 403
    - OTMA trace 400
    - OTMA verb-to-code cross reference table 404
    - receive-any buffer analysis 361
    - terminal communication task trace 325
- DC problem, diagnosing 22
- DC trace
  - diagnosing line problem 334
  - diagnosing terminal problem 334
  - starting 327
  - stopping 328
  - trace output example 340
  - trace record
    - identifiers 330
    - printing 329
    - table of record types and contents 331
- DCB definition/mapping macro 68
- DCB-EXT definition/mapping macro 68
- DDIR definition/mapping macro 68
- DDM (device-dependent module)
  - communication analyzer
    - entry point 325
    - save area 326
    - trace ID 326
    - trace output 327
    - trace record example 327
    - trace record format 327
  - function 433
- deadlock involving non-IRLM resources 57
- deadlock involving only IRLM resources 57
- debugging and diagnostic aids for Spool API
  - debugging tips 409
  - internal trace table 409
- DEDB (data entry database)
  - CI problem
    - CI 0 419
    - CI 1 419
    - common data 422
    - diagnosis aids 419
    - first DOVF CI 420
    - first IOVF CI 420
    - other DOVF CIs 420
    - other IOVF CIs 421
    - other SDEP CI 421
    - RAP CI 420
    - scraps 421
    - type identification 419
- DELETE module
  - DL/I trace, using 300
  - dependency keyword table 545
  - dependency keywords 59
  - dependent region address space (DP)
    - FMTIMS statement example 159
  - developing search arguments 29
- DEVICE BUSY category
  - sense-status message
    - 3270 recovery analysis 363
- DEVICE END category
  - sense-status message
    - 3270 recovery analysis 363
- device-dependent module (DDM)
  - communication analyzer
    - entry point 325
    - save area 326
    - trace ID 326
    - trace output 327
    - trace record example 327
    - trace record format 327
  - function 433
- DFS1269E OTMA failure message 405
- DFS1959E, reason codes 392
- DFS1965 APPC/MVS call failure, reason codes 399
- DFS2712I message
  - using in Fast Path problem analysis 416
- DFS3672I message 373
- DFSAVEC definition/mapping macro 68
- DFSCBT00 pools 91
- DFSCMC00 module, MSC analyzer 442
- DFSCMC10 module
  - abnormal-end appendage 444
  - channel-end appendage 443
  - shutdown appendage 444
- DFSCMC40 module
  - attention DIE routine 442
  - I/O request DIE routine 443
- DFSCMC50 module
  - shutdown processing routine 442
- DFSCNXA0 module
  - error messages 379



- DFSCNXA0 module (*continued*)
  - location codes for error messages 373
  - tracing errors 372
- DFSDDLTO (DL/I test program) 255
- DFSDLA30 (DC call analyzer)
  - tracing using IMS Transaction 358
- DFSDLTRO call image capture trace description 312
- DFSDOPTe definition/mapping macro 68
- DFSDPBFH definition/mapping macro 68
- DFSERA10 (File Select and Formatting Print utility)
  - exit routines 152
  - formatted output example 153
  - function 152
  - printing DC trace records 329
  - unformatted output example 153
- DFSERA20
  - See SNAP
- DFSICIO0 (communication analyzer)
  - DDM
    - entry point 325
    - trace ID 326
  - description 325
  - save area 326
  - trace output 327
  - trace record example 327
  - trace record format 327
- DFSPRPX0 parameter block diagram 81
- DFSQMRQ0 processor module description 341
- DFSSBHD0 utility
  - using with SB IMAGE CAPTURE option 318
- DFSSBWO definition/mapping macro 68
- DFSVTPO0
  - overlay for posting of VTCBs 337
- DFSZD510 control block dump
  - description 319
  - formatted dump example 321
  - unformatted dump example 322
- diagnosing
  - a control or DL/I region loop 19
  - a control region wait or hang 18
  - a CQS-related problem 521
  - a database related problem 25
  - a DB2 ESS interface problem 20
  - a DBCTL-related problem 21
  - a DBRC-related problem 21
  - a DC-related problem 22
  - a Recovery Resource Service related problem 26
  - an APPC-related problem 23
  - an ESAF interface related problem 25
  - an IMS dependent region wait or loop 20
- DIF/MID linkage diagram 112
- dispatcher trace
  - example 211
  - format 203
- DL/I 187
  - See *also* database
  - analyzing problems 261
  - call image capture trace description 312
  - control block
    - description 67
    - diagram 101
- DL/I (*continued*)
  - data record format 113
  - FMTIMS statement example 159
  - online formatted dump
    - data areas dumped 187
  - test program
    - debugging in batch environment 258
    - description 255
    - using 255
  - trace, DL/I
    - buffer handler function codes 296
    - buffer handler module trace IDs 298
    - buffer handler return codes 298
    - DELETE module, using DL/I trace 300
    - description 263
    - JRNAD codes 295
    - output sample 300
    - PSTLRPRM codes 283
    - record format 264
  - trace, other database-related
    - JCB (job control block) 253
    - locating 262
    - PI (program isolation) 311
    - retrieve 301
- DL/I region loop, diagnosing 19
- DL/I test program (DFSDDLTO) 255
- DL/I trace 300
- DMAC definition/mapping macro 68
- DMB definition/mapping macro 68
- DMBSEC definition/mapping macro 68
- DMCB definition/mapping macro 68
- DMHR definition/mapping macro 68
- DOC keyword procedure 35
- documentation management
  - description 11
  - dump preservation 12
  - IMS master console log preservation 12
  - IMS OLDS/SLDS preservation 12
  - JES JOBLOG preservation 11
  - SYS1.LOGREC preservation 12
  - z/OS system console (syslog) preservation 11
- DOF/MOD linkage diagram 111
- DOVF CI
  - diagnosing CI problem in DEDB
    - first CIs 420
    - other CIs 420
- DP (dependent region address space)
  - FMTIMS statement example 159
- DRA (Database Resource Adapter)
  - Analyzing DRA Problems 478
  - dump 477
  - dump title format 171
  - recovery tokens 478
  - Service Aids 477
- DSEB definition/mapping macro 68
- DSECT
  - for database log record (X'50') 313
- DSG definition/mapping macro 68
- DSPCABN0 trace entry 456
- DSPCRTR0 trace entry 457
- DSPSTACK trace entry 453, 455

DSPSTFRE trace entry 453, 455  
 DSPSTGET trace entry 453, 454  
 DSPTRACE (DBRC trace)  
   BGNRETRY entry 456  
   CRTR0XIT entry 457  
   DSPCABN0 entry 456  
   DSPCRTR0 entry 457  
   DSPSTACK entry 453, 455  
   DSPSTFRE entry 453, 455  
   DSPSTGET entry 453, 454  
   DSPURI00 entry 457, 459  
   GETFEED entry 460  
   locating 452  
   processing flow 454  
   using 452  
 DSPURI00 module  
   calling 457  
   entry trace entry 459  
   exit routine trace entry 462  
   GETFEED trace entry 460  
 DSPURI00 trace entry 457  
 DSPWRK1 definition/mapping macro 68  
 dump  
   buffer handler request sequence analysis 260  
   detailed analysis 259  
   DL/I call sequence analysis 260  
   formatted offline  
     See ODF (offline dump formatter)  
   formatted online  
     CTL address space 184  
     description 183  
     DL/I address space 187  
   general analysis 259  
   interactive dump formatter  
     description 178  
     using 179  
   IRLM address space dump  
     See SDUMP  
   save area analysis 260  
 DUMPQ 53

## E

ECB definition/mapping macro 68  
 ECNT definition/mapping macro 68  
 edited command format 177  
 EDSG definition/mapping macro 68  
 EIB definition/mapping macro 69  
 EMHB definition/mapping macro 68  
 EPCB definition/mapping macro 69  
 EPF definition/mapping macro 69  
 EPST definition/mapping macro 69  
 EQEL definition/mapping macro 69  
 error codes  
   0002 406  
   0004 407  
   0006 407  
   0008 407  
   000A 407  
   000C 408  
   000E 408

error location codes 377, 378  
   dynamic logon errors 376  
   error messages issued by DFSCNXA0 373  
   ISC processing 374  
   logging-on device characteristics 378  
   MSC errors 376  
   related to existing ISC session errors 377  
   user-logon-exit processing 377  
 error recovery  
   3270 device  
     log record X'6703' 364  
     sense-status message 363  
   BTAM, IEA000I message 364  
 ESAF Interface  
   diagnosis 25  
 ESCD definition/mapping macro 69  
 ESRB definition/mapping macro 69  
 ESRT definition/mapping macro 69  
 ESS (external subsystem)  
   trace output example 225  
   trace record  
     format 212  
     module ID and subfunction table 213  
     variable section layout 214  
 external SNAP call, control blocks dumped 256  
 external subsystem (ESS)  
   trace output example 225  
   trace record  
     format 212  
     module ID and subfunction table 213  
     variable section layout 214  
 external trace environment 8  
 external trace, DBRC  
   RECON I/O error processing example 471  
   record format 469  
   router processing example 470  
   using 469

## F

FAQE definition/mapping macro 69  
 Fast Path  
   ABENDU1026 analysis 415  
   control block  
     locating in an IMS dump 422  
     detailed control block diagram 98  
   Fast Path Messages, procedure for 482  
   FMTIMS statement example 160  
   general control block diagram 97  
   Last MSC Message, Determining 483  
     Using log records to get last MSC message 483  
   service aid  
     CI contention analysis 422  
     DEDB CI problem assistance aids 419  
     description 415  
     locating control blocks and tables in an IMS  
       dump 422  
   traces 9  
   transaction retry  
     description 418  
     processing flow 418

Fast Path (*continued*)  
 transaction retry (*continued*)  
     system programmer response 418  
 work area  
     locating in an IMS dump 422  
 Fast Path trace  
     description 251  
     record format 251  
     X'9C' trace format 251  
     X'9D' trace format 252  
 FDB definition/mapping macro 69  
 FDT definition/mapping macro 69  
 FEDB definition/mapping macro 69  
 FEIB definition/mapping macro 69  
 File Select and Formatting Print utility (DFSERA10)  
     exit routines 152  
     formatted output example 153  
     function 152  
     printing DC trace records 329  
     unformatted output example 153  
 FMTIMS statement  
     choosing parameters for ODF 156, 157  
     control region example 159  
     DBRC example 159  
     DC example 159  
     DL/I example 159  
     DP example 159  
     Fast Path example 160  
     formatted areas 162  
     LOG example 160  
     options 161  
     sample statements 158  
     syntax restrictions 160  
     table 157  
     VSAM example 160  
 FMTO option  
     specify FMTO option 5  
 FMTO=D parameter value 5  
 formatted dump, offline  
     See ODF (offline dump formatter)  
 FRB definition/mapping macro 69

## G

GB definition/mapping macro 69  
 GBCB definition/mapping macro 69  
 Generalized Sequential Access Method (GSAM)  
     control block dump 319  
     detailed control block diagram 100  
     formatted dump (DFSZD510)  
         description 319  
         example 321  
     general control block diagram 99  
     out-of-space abend 322  
     unformatted dump (DFSZD510), example 322  
 GETFEED trace entry 460  
 GLT definition/mapping macro 69  
 GPT definition/mapping macro 69  
 GQCB definition/mapping macro 69  
 GSAM (Generalized Sequential Access Method)  
     control block dump 319

GSAM (Generalized Sequential Access Method)  
 (*continued*)  
     detailed control block diagram 100  
     formatted dump (DFSZD510)  
         description 319  
         example 321  
     general control block diagram 99  
     out-of-space abend 322  
     unformatted dump (DFSZD510), example 322  
 GTF (Generalized Trace Facility) trace  
     DBRC-related  
         example 470  
         formatting and printing 470  
         record format 469  
         using 469, 470

## H

hang, diagnosing a control region 18  
 hardware I/O category, 3270 error recovery  
     analysis 363  
 HDAM database  
     OSAM ESDS block format 117  
     segment format 115  
     VSAM ESDS block format 117  
 hi-level dump formatting 179  
 HIDAM database  
     OSAM and VSAM ESDS block format 117  
     segment format 115  
     VSAM ESDS block format 117  
 HIDAM index database  
     VSAM LRECL format 117  
 HISAM database  
     block format 115  
     LRECL format 115  
     segment format 114  
 HSAM database  
     block format 113  
     delete byte format 113  
     flag byte format 113  
     segment format 113  
 HSSD definition/mapping macro 69  
 HSSO definition/mapping macro 69  
 HSSP definition/mapping macro 69  
 HSSR definition/mapping macro 69

## I

IBFPRF definition/mapping macro 69  
 IBPOOL definition/mapping macro 69  
 IDC0 trace table entries 380  
     internal trace formats that map entries 380  
 IDSC definition/mapping macro 70  
 IEBGENER 10  
 IEEQE definition/mapping macro 70  
 IMODULE facility  
     CBT pool 583  
     IMS control block/work area, locating  
         using load list 581  
 IMS component identification numbers 31

IMS Connect Dump Formatter  
 about 182

IMS dependent region wait or loop, diagnosing 20

IMS diagnostic aids 382

IMS dump formatter 8

IMS dump, locating  
 Fast Path control block 422  
 Fast Path work area 422

IMS setup recommendations 5  
 external trace environment 7  
 FMTO option 5  
 interactive dump formatter 7  
 SYSDUMP DD 6  
 Table Traces 6

IMS sysplex dump considerations 14  
 sysplex IEADMCxx dump activation 15  
 sysplex IEADMCxx example 14

IMS Transaction trace  
 content 358  
 description 358  
 example 359  
 starting 358

IMS.ACBLIB  
 members layout 65  
 partitioned data set 65

INCORROUT keyword procedure 37

intent conflict 50

interactive dump formatter  
 description 178  
 using 179

interactive problem control system (IPCS)  
 dump formatter 178  
 IRLM configuration 412  
 using with ODF 156

internal resource lock manager (IRLM)  
 keyword procedure 35  
 latch unavailable 58  
 lock request example 109  
 overall control block diagram 108  
 procedure for WAIT state 57  
 service aid  
 description 411  
 dump 412  
 SYS1.LOGREC 412  
 storage manager pool diagram 109

internal trace table  
 description 409

Intersystem Communication (ISC) link  
 starting DC trace 328  
 stopping DC trace 328

intersystem control block diagram 102

INTERVENTION REQUIRED category  
 3270 error recovery analysis 363

IOVF CI  
 diagnosing CI problem in DEDB  
 first CIs 420  
 other CIs 421

IPCS (interactive problem control system)  
 dump formatter 178  
 IRLM configuration 412  
 using with ODF 156

IRLM (internal resource lock manager)  
 keyword procedure 35  
 latch unavailable 58  
 lock request example 109  
 overall control block diagram 108  
 procedure for WAIT state 57  
 service aid  
 description 411  
 dump 412  
 SYS1.LOGREC 412  
 storage manager pool diagram 109

ISC (Intersystem Communication) link  
 starting DC trace 328  
 stopping DC trace 328

ISL definition/mapping macro 70

ISPL definition/mapping macro 70

ITASK ECB posting 211

## J

JCB (job control block) trace  
 content 253  
 function codes 254  
 output sample 254

JCB definition/mapping macro 70

JCL (job control language)  
 printing 6701-MRQE records 346  
 printing QCF SCRAPLOG records 345

job control block (JCB) trace  
 content 253  
 function codes 254  
 output sample 254

job control language (JCL)  
 printing 6701-MRQE records 346  
 printing QCF SCRAPLOG records 345

JRNAD codes 295

## K

KBLA (Knowledge-Based Log Analysis)  
 DL/I trace formatting 264  
 lock trace analysis 411  
 MSC link analysis 27, 435  
 print DC trace records 329  
 saving online log data set input 22

keyword  
 component identification procedure 31  
 definition 29  
 dictionary 541  
 selecting 31  
 type-of-failure 31  
 types used with CHNG and SETO calls 406  
 using dependency keywords 59

keyword procedure  
 ABENDUxxxx 33  
 ABENDxxx 32  
 DOC 35  
 INCORROUT 37  
 IRLM 35  
 MSG 37  
 PERFM 36

keyword procedure (*continued*)  
 SAP analysis 43  
 WAIT/LOOP 40  
 Knowledge-Based Log Analysis (KBLA)  
 DL/I trace formatting 264  
 lock trace analysis 411  
 MSC link analysis 27, 435  
 print DC trace records 329  
 saving online log data set input 22

**L**

latch trace  
 example 246  
 format 242  
 latch manager trace entries 243  
 system locate control function entries 245  
 use manager trace entries 243  
 LCB definition/mapping macro 70  
 LCD definition/mapping macro 70  
 LCDSECT definition/mapping macro 70  
 LCRE definition/mapping macro 70  
 LEV definition/mapping macro 70  
 limits for locking resources 311  
 line problem, diagnosing using DC trace 334  
 link problem  
 diagnosing 435  
 MSS1 and MSS2 record description 440  
 LIPARMS definition/mapping macro 70  
 LLB definition/mapping macro 70  
 lock request not granted 57  
 locking resources, limiting with LOCKMAX 311  
 LOCKMAX parameter 311  
 log  
 FMTIMS statement example 160  
 log analysis, database-related 312  
 log data set, retrieving call image capture data 258  
 log records  
 all records used to analyze IMS problems 127  
 data area format 151  
 description 529  
 for service errors 410  
 log sequence field format 151  
 prefix area format 150  
 printing 531  
 produced by Spool API 409  
 subrecord area format 151  
 table 530  
 type 67D0 138, 409, 410  
 type X'29' 141  
 type X'49' 519  
 type X'50' 313  
 type X'67' 150, 153  
 type X'6703' 364  
 type X'67D0' 138  
 type X'68' 409  
 types 530  
 viewing format 127, 529  
 log router 492  
 log sequence field format 151  
 log subrecord and data area formats 151

log, FMTIMS statement example 156  
 logical LINK  
 starting DC trace 328  
 stopping DC trace 328  
 loop, diagnosing a control or DL/I region 19  
 loop, diagnosing an IMS dependent region 20  
 low-level dump formatting 179  
 LRECL format 115  
 LTB definition/mapping macro 70  
 LU Manager Trace 382  
 LXB definition/mapping macro 70  
 LXB trace  
 DFSCMC00 module, MSC analyzer 442  
 DFSCMC10 module  
 abnormal-end appendage 444  
 channel-end appendage 443  
 shutdown appendage 444  
 DFSCMC40 module  
 attention DIE routine 442  
 I/O request DIE routine 443  
 DFSCMC50 module  
 shutdown processing routine 442  
 example 444  
 using 441

**M**

macros for mapping control blocks 67, 68, 69, 70, 71, 72, 73  
 main storage-to-main-storage  
 access method trace description 435  
 save set trace description 435  
 Manual Intervention for Dump Creation  
 deciding when to dump 13  
 description 12  
 IEADMCxx, MVS SYS1.PARMLIB 14  
 EADMCxx DUMP activation 14  
 EADMCxx example for IMS 14  
 IMS dump techniques 13  
 mapping macros for control blocks 67, 68, 69, 70, 71, 72, 73  
 master trace table Size, z/OS 6  
 Message Format Service (MFS)  
 BTAM error  
 diagnosing 364  
 diagram of normal BTAM path 364  
 diagnosing problems 369  
 module trace  
 CIBSTRAC 371  
 CIBTRACE 372  
 message processing (BUFMSTRA) trace  
 description 435  
 Message Requeuer (MRQ)  
 AIBREASN Codes list 554  
 AIBREASN Codes Set by DFSQMRQ0 547  
 Message Requeuer (MRQ)/Queue Control Facility (QCF) 547  
 AIBREASN codes  
 description 347  
 DFSQMRQ0 processor module 341

- Message Requeuer (MRQ)/Queue Control Facility (QCF) *(continued)*
- JCL
    - printing 6701-MRQE records 346
    - printing QCF SCRAPLOG records 345
  - key fields and offsets of diagnostic records 344
  - key fields in message 345
  - messages successfully requeued 348
  - MRQE diagnostic records 345
  - obtaining additional diagnostics 347
  - sample of successful message requeue 348
  - sample SCRAPLOG record and description 343
- MFS (Message Format Service)
- BTAM error
    - diagnosing 364
    - diagram of normal BTAM path 364
  - diagnosing problems 369
  - module trace
    - CIBSTRAC 371
    - CIBTRACE 372
- MID/DIF linkage diagram 112
- MOD/DOF linkage diagram 111
- module directory, locating 65
- MRMB definition/mapping macro 70
- MRQ (Message Requeuer)/QCF (Queue Control Facility) 547
- AIBREASN codes
    - description 347
  - JCL
    - printing 6701-MRQE records 346
    - printing QCF SCRAPLOG records 345
  - key fields and offsets of diagnostic records 344
  - key fields in message 345
  - messages successfully requeued 348
  - MRQE diagnostic records 345
  - obtaining additional diagnostics 347
  - sample of successful message requeue 348
  - sample SCRAPLOG record and description 343
- MRQE diagnostic records
- control blocks and mapping macros 346
  - description 345
  - sample JCL for printing 346
- MSC (Multiple Systems Coupling)
- abnormal-end appendage 444
  - analyzer trace entry 442
  - attention DIE routine 442
  - BUFMSTRA (message processing) trace,
    - description 435
  - BUFSMVID trace 446
  - channel-end appendage 443
  - channel-to-channel access method trace stack 441
  - communication task trace
    - description 433
    - diagram 434
  - DDM function 433
  - detailed control block diagram 106
  - general control block diagram 105
  - I/O request DIE routine 443
  - main storage-to-main storage access method trace 435
  - main storage-to-main storage save set trace 435
  - MSS1 and MSS2 record description 440
  - service aid 433
  - shutdown appendage 444
  - shutdown processing routine 442
- MVS setup recommendations 3
- N**
- no work to do (wait/loop) 49
  - node trace
    - DC-related problem diagnosis 22
    - turning on 22
    - using to diagnose routing problems 358
    - using with INCORROUT procedure 39
    - using with WAIT/LOOP procedure 40
- O**
- obtaining 6701-MRQB records 347
- ODF (offline dump formatter)
- control blocks, locating 172
  - description 154
  - dump data set input 155
  - executing 155
  - FMTIMS parameter
    - table 157
  - introduction 154, 155
  - output order 173
  - recommendations for using 155

ODF (offline dump formatter) (*continued*)  
 SDUMP input 155  
 title format 170  
 using 155  
 using with IPCS 156  
 offline dump formatter (ODF)  
 control blocks, locating 172  
 description 154  
 dump data set input 155  
 executing 155  
 FMTIMS parameter  
 table 157  
 introduction 154, 155  
 output order 173  
 recommendations for using 155  
 SDUMP input 155  
 title format 170  
 using 155  
 using with IPCS 156  
 offloading trace data set 10  
 online environment  
 call image capture trace 258  
 Online Recovery Manager (ORTT)  
 example 310  
 format 306  
 starting 306  
 Open Transaction Manager Access (OTMA)  
 DFS1269E message 405  
 dumps 405  
 log records 405  
 module-to-cross reference table 403  
 trace  
 description 400  
 format of trace records 400  
 verb-to-code cross reference table 404  
 OSAM (Overflow Sequential Access Method)  
 buffer pool diagram 82  
 DECB with IOB in use 85  
 OSAM and VSAM ESDS block format 117  
 OTMA (Open Transaction Manager Access)  
 DFS1269E message 405  
 dumps 405  
 log records 405  
 module-to-cross reference table 403  
 trace  
 description 400  
 format of trace records 400  
 verb-to-code cross reference table 404  
 out-of-space abend, GSAM 322  
 output data sets, creating 10  
 Overflow Sequential Access Method (OSAM)  
 buffer pool diagram 82  
 DECB with IOB in use 85

## P

PAC definition/mapping macro 70  
 PAPL definition/mapping macro 70  
 PARMLIST definition/mapping macro 70  
 PAT definition/mapping macro 70  
 PATE definition/mapping macro 70

PCA definition/mapping macro 70  
 PCB definition/mapping macro 73  
 PCIB definition/mapping macro 70  
 PCPARMS definition/mapping macro 70  
 PCT definition/mapping macro 70  
 PDA definition/mapping macro 70  
 PDIR definition/mapping macro 71  
 PDL definition/mapping macro 71  
 PEC definition/mapping macro 71  
 PERFM keyword procedure 36  
 PHDAM database  
 segment format 115  
 variable-length segment format 119  
 PHIDAM database  
 segment format 115  
 variable-length segment format 119  
 PI (program isolation)  
 problem analysis 311  
 trace facility 311  
 PNT definition/mapping macro 71  
 POOLHDR definition/mapping macro 71  
 post code list 212  
 posting of ITASK ECBs 211  
 PPRE definition/mapping macro 71  
 PQE definition/mapping macro 71  
 prefix area format for log records 150  
 preparing an APARs 61  
 print utility  
 See DFSERA10 (File Select and Formatting Print utility)  
 processor module for QCF 341  
 program isolation (PI)  
 problem analysis 311  
 trace facility 311  
 program parameters  
 LOCKMAX 311  
 PSB definition/mapping macro 71  
 PSDB definition/mapping macro 71  
 pseudoabend, cause 259  
 PST active 48  
 PST analysis 46  
 PST definition/mapping macro 71  
 PSTLRPRM codes 283  
 PTBWA definition/mapping macro 71  
 PTE definition/mapping macro 71  
 PTK definition/mapping macro 71  
 PTX definition/mapping macro 71  
 PURGE 53  
 PXPparms definition/mapping macro 71

## Q

QCB definition/mapping macro 71  
 QCF (Queue Control Facility)/MRQ (Message Requirer)  
 DFSQMRQ0 processor module 341  
 JCL  
 printing 6701-MRQE records 346  
 printing MRQ SCRAPLOG records 345  
 key fields and offsets of diagnostic records 344  
 key fields in message 345

QCF (Queue Control Facility)/MRQ (Message Requeuer) (*continued*)

- messages successfully requeued 348
- MRQE diagnostic records 345
- obtaining additional diagnostics 347
- sample of successful message requeue 348
- sample SCRAPLOG record and description 343

QCF SCRAPLOG records, sample JCL for printing 345

QEL definition/mapping macro 71

QMBA definition/mapping macro 71

qualifier codes

- ETO parsing errors 379
- screen-attribute errors 379
- VTCB-creation errors 379

Queue Control Facility (QCF)/Message Requeuer (MRQ)

- DFSQMRQ0 processor module 341
- JCL
  - printing 6701-MRQE records 346
  - printing MRQ SCRAPLOG records 345
- key fields and offsets of diagnostic records 344
- key fields in message 345
- messages successfully requeued 348
- MRQE diagnostic records 345
- obtaining additional diagnostics 347
- sample of successful message requeue 348
- sample SCRAPLOG record and description 343

queue manager trace

- description 246
- record format 247

## R

RAP CI

- diagnosing CI problem in DEDB
  - CI format 420

RCPARMS definition/mapping macro 71

RCTE definition/mapping macro 71

RDLWA definition/mapping macro 71

receive-any buffer analysis 361

RECON data set 447

- listing records 447

REG0 trace 325

REPLACE module

- DL/I trace, using 300

request parameter list (RPL) 363

retrieve trace

- ID table 304
- output sample 305
- using 301

RHB definition/mapping macro 71

RHT definition/mapping macro 71

RLB definition/mapping macro 71

RLCBT definition/mapping macro 71

RLMCB definition/mapping macro 71

RLPL definition/mapping macro 71

RLQD definition/mapping macro 71

RPL (request parameter list) 363

RPLI definition/mapping macro 71

RPST definition/mapping macro 72

RRE definition/mapping macro 72

## S

SAP analysis procedure 43

SAP definition/mapping macro 72

save area set

- Fast Path problem analysis
  - example 417
  - finding during DC analysis 363

save area set, abnormal 45

SB (sequential buffering)

- COMPARE option, use in SB 318
- control block diagram 83
- DFSSBHD0 utility
  - using with SB IMAGE CAPTURE option 318
- DL/I trace table entry 316
- SB IMAGE CAPTURE option
  - using with DFSSBHD0 utility 318
- SBESNAP option, activating 318
- SBSNAP option
  - activating 317
  - limiting output 317
  - service aid tool 316
- SBESNAP option, activating 318

SBHE definition/mapping macro 72

SBPARMS definition/mapping macro 72

SBPSS definition/mapping macro 72

SBPST definition/mapping macro 72

SBSCD definition/mapping macro 72

SBSNAP option

- activating 317
- limiting output 317

SBUE definition/mapping macro 72

SBUF definition/mapping macro 72

SCA1 definition/mapping macro 72

SCAR definition/mapping macro 72

SCD definition/mapping macro 72

SCD diagram, online 80

scheduler trace

- example 241
- format 238

SCRAPLOG for QCF

- description/sample record 343
- JCL for printing records 345

SDB definition/mapping macro 72

SDB keyword dictionary 541

SDCB definition/mapping macro 72

SDEP CI

- diagnosing CI problem in DEDB
  - format 421

SDSG definition/mapping macro 72

SDUMP

- IRLM address space dump
  - description 412
  - formatting and printing 412
- ODF 155

SDWA definition/mapping macro 72

search argument

- release level used 60



- search arguments
  - creating 30
  - developing 29
- searching for APARs 60
- searching problem reporting databases 29
- secondary allocation 10
- secondary index database
  - block format 119
  - segment data format 119
  - VSAM LRECL format 118, 119
- segment prefix mapping 116
- selecting keywords 31
- sense-status message 363
- sequential buffering (SB)
  - COMPARE option, use in SB 318
  - control block diagram 83
  - DFSSBHD0 utility
    - using with SB IMAGE CAPTURE option 318
  - DL/I trace table entry 316
  - SB IMAGE CAPTURE option
    - using with DFSSBHD0 utility 318
  - SBESNAP option, activating 318
  - SBSNAP option
    - activating 317
    - limiting output 317
  - service aid tool 316
- service aid
  - DB (database) 253
  - DBRC 447
  - DC 325
  - Fast Path 415
  - IRLM 411
  - MSC 433
  - SYS 127
- service error log records
  - causes 410
  - type 67D0 410
- SETO call
  - Spool API 405
- setting up your system 3
- SGT definition/mapping macro 72
- shared queues interface trace
  - description 250
- SHISAM database
  - block format 115
  - LRECL format 115
  - segment format 114
- SHSAM database
  - block format 113
  - delete byte format 113
  - flag byte format 113
  - segment format 113
- shutdown analysis 53
- shutdown processing 53
- SIDB definition/mapping macro 72
- SIDX definition/mapping macro 72
- SMB definition/mapping macro 72
- SNAP
  - call facility (DFSERA20)
    - description 189
    - output 189
- SNAP (*continued*)
  - COMPARE statement, SNAP call 255
  - control block output 256
  - exceptional condition 256
  - SBESNAP option 318
  - SBSNAP option 317
  - specific call
    - description 257
    - SB COMPARE option 257
    - SBESNAP option 257
    - SBSNAP option 257
- Software Support Facility (SSF)
  - searching 59
- space management module trace IDs 298
- special abend processing
  - Spool API support 409
- specify SYSMDUMP statement 6
- specify SYSUDUMP statement 6
- Spool API
  - CHNG and SETO calls 405
  - debugging tips 409
  - feedback from parsing errors 405
  - interfacing directly to 405
  - log records produced by 409
  - special abend processing 409
- SPQB definition/mapping macro 73
- SQPST definition/mapping macro 73
- SRAN definition/mapping macro 73
- SSF (Software Support Facility)
  - searching 59
- SSIB definition/mapping macro 73
- SSOB definition/mapping macro 73
- SST (subsystem trace)
  - trace output example 225
  - trace record
    - format 212
    - module ID and subfunction table 213
    - variable section layout 214
- SSVP definition/mapping macro 73
- static DB/DC environment 66
- status codes associated with keywords
  - AR 406
  - AS 406
- storage management
  - control block relationships created for MAIN pool 87
  - control block relationships for DFSCBT00 pools 91
  - control block relationships for DFSPPOOL pools 90
  - control block relationships for preallocated storage blocks 88
- storage manager trace 241
- subsystem trace (SST)
  - trace output example 225
  - trace record
    - format 212
    - module ID and subfunction table 213
    - variable section layout 214
- SYS (systems)
  - service aid
    - common trace table interface 190
    - description 127
    - dispatcher trace 203

- SYS (systems) *(continued)*
    - service aid *(continued)*
      - dumps, formatting online 183
      - external subsystem trace 212
      - Fast Path trace 251
      - File Select and Formatting Print utility (DFSERA10) 152
      - ITASK ECB posting 211
      - log record format (type X'29') 141
      - log record format (type X'67') 150
      - log record layout (type X'49') 519
      - log record table 127
      - ODF (offline dump formatter) 154
      - queue manager trace 246
      - scheduler trace 238
      - shared queues interface trace 250
      - Snap call facility 189
  - SYS1.DUMPXX data set
    - IRLM address space dump
      - description 412
      - formatting and printing 412
  - SYS1.DUMPxx data sets 5
  - SYS1.LOGREC record
    - IRLM diagnosis 412
  - SYSMDUMP statement
    - dump preservation 12
    - specify 6
  - system analysis 127
    - See SYS (systems)
  - system post code list 212
  - system service aid
    - See SYS (systems)
  - system set up
    - CQS tracing 10
    - external trace environment 8
    - IMS Control Region EXEC 5
    - IMS dump formatter 8
    - specify SYSMDUMP statement 6
    - specify SYSUDUMP statement 6
    - SYS1.DUMPxx data sets 5
    - writing trace tables 9
    - z/OS master trace table size 6
  - system trace table 6
  - system wait 53
  - SYSUDUMP statement
    - dump preservation 12
    - specify 6
- T**
- TAB definition/mapping macro 73
  - tables, writing trace 9
  - TCT definition/mapping macro 73
  - terminal communication task trace
    - entry point 325
    - save area 326
    - trace ID 326
    - trace output 327
    - trace record example 327
    - trace record format 327
  - terminal problem
    - diagnosing using DC trace 334
  - trace
    - log router 492
  - Trace Entry
    - Fast Path Log Router Interface 483
  - trace table
    - external trace environment
      - starting and stopping 8
    - locating 193
    - sizes
      - z/OS master 6
      - z/OS system 6
  - trace tables, writing 9
  - trace, DL/I
    - use to analyze DL/I call 300
  - traces
    - (ORTT) Online Recovery Manager 306
    - CIBSTRAC 371
    - CIBTRACE 372
    - common trace table interface 190
    - controlling the volume 8
    - CQS 10
    - DBRC 452
    - DBRC external 469
    - DC 327
    - dispatcher 203
    - DL/I 263
    - DL/I call image capture 257
    - external subsystem 212
    - fast path 251
    - Fast Path 9
    - IMS Transaction 358
    - job control block 253
    - LXB 442
    - MSC communication task 433
    - offloading trace data set 10
    - Online Recovery Manager (ORTT) 306
    - OTMA 400
    - program isolation 311
    - queue manager 246
    - retrieve 301
    - scheduler 238
    - shared queues interface 250
    - which to run at all times 9
  - Transaction Manager control block diagram 102
  - transaction retry, Fast Path
    - description 418
    - processing flow 418
    - system programmer response 418
  - type-of-failure keyword 31
- U**
- UEHB definition/mapping macro 73
  - UPAD codes 295
  - UXDT definition/mapping macro 73
  - UXRB control block 73
  - UXRB definition/mapping macro 73

## V

- variable-length segment
  - HDAM format 119
  - HIDAM format 119
  - HISAM format 119
  - PHDAM format 119
  - PHIDAM format 119
- Virtual Storage Access Method (VSAM)
  - FMTIMS statement example 160
  - LRECL format 117
  - PSINDEX 118
  - secondary index 118
- Virtual Telecommunications Access Method (VTAM)
  - request parameter list (RPL) 363
  - RPL (request parameter list) 363
  - terminal problem
    - starting DC trace 328
    - stopping DC trace 328
- VSAM (Virtual Storage Access Method)
  - FMTIMS statement example 160
  - LRECL format 117
  - PSINDEX 118
  - secondary index 118
- VSI definition/mapping macro 73
- VTAM (Virtual Telecommunications Access Method)
  - request parameter list (RPL) 363
  - RPL (request parameter list) 363
  - terminal problem
    - starting DC trace 328
    - stopping DC trace 328
- VTCB
  - load module diagram 103
  - posting of overlays in DFSVTPO0 337

## W

- wait for input 51
- wait, diagnosing a control region 18
- wait, diagnosing an IMS dependent region 20
- wait/hang problem, which dumps are sufficient 41
- WAIT/LOOP procedure 40
- WHB definition/mapping macro 73
- work area
  - CBT pool 583
  - Fast Path
    - locating in an IMS dump 422
    - locating using load list 581
- writing trace tables 9

## X

- X'29' 141
  - log record layout
    - X'2900' 141
    - X'2910' 142
    - X'2920' 143
    - X'2930' 143
    - X'2940' 146
    - X'2950' 147
    - X'2970' 148

- X'29' (continued)
  - log record layout (continued)
    - X'2990' 149
- X'4930' log record
  - layout 519
- X'6701' log records
  - log sequence field 151
  - map 336
  - prefix area 150
  - subrecord and data area formats 151
- X'68' log record 409
- X'D9' trace entry 287
- XCRB definition/mapping macro 73
- XMCA definition/mapping macro 73
- XMCI definition/mapping macro 73
- XRF environment
  - starting DC trace 328
  - stopping DC trace 329

## Z

- z/OS component trace
  - example 413
- z/OS setup recommendations
  - Automatic Dump Data set Allocation 4
  - CHNGDUMP MAXSPACE 4
  - common storage tracker 4
  - system trace table 3
- z/OS storage map diagram 107
- z/OS system trace table 6
- ZIB definition/mapping macro 73







Program Number: 5655-J38

Licensed Materials – Property of IBM  
Printed in USA

LY37-3203-00



Spine information:



IMS

Diagnosis Guide and Reference

Version 9